Dynamic Cross-Modal Prompt Generation for Multimodal Continual Instruction Tuning

Tao Hu[1,2] **Da-Wei Zhou**[1,2(✉)]
[1] School of Artificial Intelligence, Nanjing University
[2] State Key Laboratory for Novel Software Technology, Nanjing University
{hut, zhoudw}@lamda.nju.edu.cn

Abstract

Multimodal Large Language Models (MLLMs) achieve strong performance through instruction tuning, yet real-world deployment often requires continual capability expansion across sequential tasks. In such scenarios, Multimodal Continual Instruction Tuning (MCIT) aims to acquire new capabilities while limiting catastrophic forgetting. Existing methods mainly follow a module-composition paradigm: they maintain task-level prompts or LoRA experts and dynamically route or aggregate a subset of them at inference. However, samples within the same task can still differ substantially in visual scenes, question intents, and reasoning demands. This motivates instance-level adaptation to individual query-image pairs rather than only selecting or combining task-level modules. To this end, we propose DRAPE (Dynamic Cross-Modal Prompt Generation), a prompt-learning framework that synthesizes continuous instance-specific soft prompts for MCIT. Instead of selecting prompts from a fixed pool, DRAPE derives prompt queries from the textual instruction and cross-attends to visual patch features, producing prompts conditioned on query-image pairs that are prepended to the frozen LLM. To mitigate forgetting during sequential updates, DRAPE applies null-space gradient projection to the shared projector and uses CLIP-based prototype routing for task-label-free generator selection at inference. Extensive experiments on MCIT benchmarks show that DRAPE achieves state-of-the-art performance among representative prompt-based and LoRA-based continual-learning baselines. Code is available at: https://github.com/LAMDA-CL/NeurIPS2026-Drape.

1 Introduction

Recent Multimodal Large Language Models (MLLMs) [23, 4] achieve strong performance through large-scale multimodal instruction tuning [55, 39], enabling them to handle diverse vision-language tasks [28, 8]. Yet practical deployment rarely stops at a fixed task mixture: new domains, instruction styles, and capabilities arrive over time, and the model must absorb them without sacrificing what it has already learned. This setting gives rise to Multimodal Continual Instruction Tuning (MCIT) [2], where the central challenge is to extend multimodal capability under sequential updates while limiting catastrophic forgetting [6, 53].

Existing MCIT methods mainly follow a module-composition paradigm. They freeze the pre-trained LLM and vision backbone, maintain lightweight task-specific modules such as prompts or LoRA experts, and dynamically route or combine these modules at inference [1, 2, 50, 9, 7]. Although such routing or retrieval mechanisms make the effective configuration input-dependent, the adaptation remains restricted to selecting or recombining discrete modules learned from previous tasks. In practice, however, samples within the same MCIT task can still differ substantially in visual scenes, question intents, reasoning demands, and instruction–image interactions. These differences require the model to dynamically adapt to individual query–image pairs rather than only selecting or recombining

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

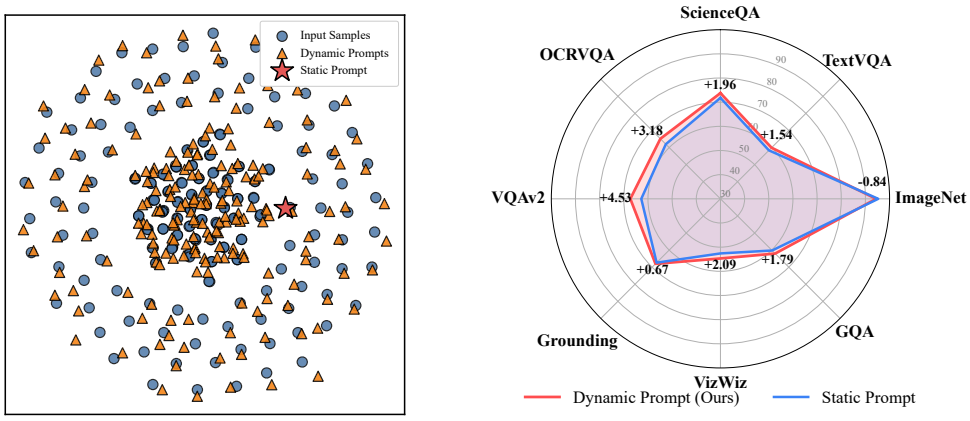


Figure 1: Why instance-specific prompts are needed. **Left:** t-SNE visualization of 200 samples from the VQAv2 test set. For each sample, we average-pool the input embeddings and the corresponding instance-specific soft prompt to obtain visualization features. For the static baseline, a single shared task prompt is average-pooled and plotted for comparison. The instance-specific soft prompts follow the sample distribution more closely, suggesting that they better reflect instance-level variation than static task prompts. **Right:** task-wise performance comparison between instance-specific and static task prompts on CoIN [2]. Instance-specific prompts achieve stronger overall performance, suggesting the benefit of modeling within-task variation rather than using one fixed task-level prompt.

task-level components. This motivates an alternative approach that synthesizes continuous prompts directly from the current multimodal input.

This instance-level variation is further illustrated in Figure 1. In the left panel, the instance-specific soft prompts generated for each sample by our cross-modal prompt generator follow the sample distribution more closely than static task prompts, suggesting that useful adaptation should track instance-level structure rather than collapse all samples in a task into the same prompt. In the right panel, instance-specific prompts achieve stronger overall performance than static task prompts on CoIN [2], further suggesting that within-task multimodal diversity cannot be fully captured by one fixed task-level representation. These observations motivate a continual-learning framework that remains lightweight like prior parameter-efficient methods while allowing the adaptation itself to vary across samples.

To address this issue, we propose DRAPE (Dynamic Cross-Modal Prompt Generation), a prompt-learning framework that generates continuous instance-specific soft prompts for MCIT. Although instance-dependent prompting has been explored in prior language-only or vision-only settings [48, 20, 47, 25], these methods do not directly model query–image conditioned adaptation under sequential multimodal instruction tuning. In contrast, DRAPE synthesizes prompts from both textual instructions and visual patch features: a lightweight cross-modal prompt generator summarizes instruction tokens into prompt queries, cross-attends to visual patch features, and produces prompts that are prepended to the frozen LLM. Rather than selecting or recombining discrete task-level components, the model synthesizes prompts that adapt to the fine-grained multimodal structure of each sample.

To reduce forgetting during sequential updates, we combine this prompt generator with two complementary mechanisms. First, we apply null-space gradient projection to the shared projector so that updates are biased toward directions that minimally interfere with previously observed feature subspaces. Second, we use a CLIP-based prototype router to select the most relevant task-specific generator without task labels at inference. Together, these components form an input-conditioned cross-modal prompt-generation framework for rehearsal-free MCIT, improving adaptation to diverse query–image pairs while limiting catastrophic forgetting throughout training over sequential tasks.

2 Related Work

Multimodal Large Language Models. Multimodal Large Language Models (MLLMs) extend large language models to vision-language tasks by connecting visual encoders with language models through cross-modal alignment modules [32, 23, 4]. Typically, visual features are projected into

the language embedding space and combined with textual instructions, enabling the model to generate responses conditioned on both modalities [23, 4]. With large-scale multimodal instruction tuning [55, 45, 26], MLLMs have achieved strong performance on visual reasoning [17, 52], visual question answering [8], instruction following [58], and multimodal generation [4]. Despite these advances, most existing MLLMs are trained on a fixed mixture of tasks and datasets [54, 2]. As a result, adapting them to newly emerging domains, instruction styles, or capabilities often requires additional fine-tuning, which may interfere with previously learned multimodal knowledge [33, 53]. This static-training assumption motivates studying how MLLMs can be adapted after deployment.

Continual Instruction Tuning. Continual instruction tuning extends standard instruction tuning by enabling foundation models to learn a sequence of tasks while remaining aligned with human instructions [10]. As MLLMs are increasingly deployed in open-world settings, mitigating catastrophic forgetting [53] in Multimodal Continual Instruction Tuning (MCIT) has become an important problem [46, 54]. Existing methods generally follow several directions. Early approaches often adapt continual learning techniques from natural language processing [42, 40, 41], using regularization or replay to preserve past knowledge. Recent parameter-efficient methods, such as MoELoRA [2], ProgLoRA [50], and HiDe-LLaVA [9], keep the MLLM backbone frozen and update only lightweight task-specific modules. More recent studies further improve expert routing and knowledge reuse. SAME [49] stabilizes expert routing and updates to mitigate router and expert drift, while CRAM [38] combines centroid-guided routing with adaptive expert allocation to balance knowledge reuse and parameter efficiency. ProtoAda [34] introduces format-aware task prototypes and geometric consolidation to reduce interference across heterogeneous response formats. ReCAP [14] further leverages external knowledge retrieval to construct instance-specific capability paths from reusable modules. These approaches primarily address continual capability preservation and reuse through expert routing, parameter adaptation, or module composition. In parallel, instance-aware prompting has also been studied in several related settings, including instance-dependent prompt generation for NLP [47], visual instance-aware or input-adaptive prompt tuning [48, 20], compact capsule prompt representations [25], and image-conditioned prompt generation for continual learning [18].

3 Preliminaries

3.1 Multimodal Continual Instruction Tuning

We consider a typical MLLM [24] that processes an image $\mathbf{v}$ and a textual instruction $\mathbf{q}$ into a concatenated multimodal sequence $\mathbf{z} = [\mathbf{w}; \mathbf{u}] \in \mathbb{R}^{(m+s)\times d}$. Here, $\mathbf{w} = \pi(\phi(\mathbf{v}))$ denotes the projected visual features from a vision encoder $\phi(\cdot)$ and a projector $\pi(\cdot)$, while $\mathbf{u} = \psi(\mathbf{q})$ represents the text embeddings. In the continual instruction tuning setting, the model encounters a sequence of tasks $\{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_T\}$ arriving sequentially, where each task $\mathcal{D}_t = \{(\mathbf{v}_i, \mathbf{q}_i, \mathbf{y}_i)\}_{i=1}^{N_t}$ consists of image–instruction–answer triplets, and $\mathbf{y}_i$ denotes the target answer token sequence. Under the strict rehearsal-free constraint (*i.e.*, past data is inaccessible), the goal is to optimize a set of tunable parameters Θ to generalize across all tasks seen so far:

$$\Theta_t^* = \arg\min_{\Theta} \ \mathbb{E}_{(\mathbf{v},\mathbf{q},\mathbf{y})\sim\mathcal{D}_{\le t}} \left[-\sum_{j=1}^{L} \log\, p_\Theta(y_j \mid \mathbf{z},\, \mathbf{y}_{<j}) \right], \tag{1}$$

where $\mathcal{D}_{\le t}$ denotes the joint distribution over all observed tasks, L is the length of the target answer sequence $\mathbf{y}$, and y_j denotes its j-th token. In practice, since training is performed exclusively on $\mathcal{D}_t$, the learning procedure must inherently mitigate catastrophic forgetting.

3.2 Baselines in MCIT

To overcome forgetting while maintaining the pre-trained multimodal knowledge, existing rehearsal-free MCIT methods typically freeze the LLM backbone and rely on two dominant paradigms: Prompt Tuning and Mixture-of-Experts (MoE) with LoRA.

Prompt Tuning: Prompt-based methods [36, 44, 51] adapt the frozen model by prepending learnable continuous vectors (soft prompts) to the input sequence. To handle a continuous sequence of tasks, they typically maintain a dynamic prompt pool $\mathcal{P}$. For a given input, specific prompts $\mathbf{P} \in \mathbb{R}^{L_p\times d}$ are retrieved or matched based on instance features. The augmented input becomes $\mathbf{z}' = [\mathbf{P}; \mathbf{z}]$, and the retrieved prompts, or task-specific prompts when task identities are used, are updated during training.

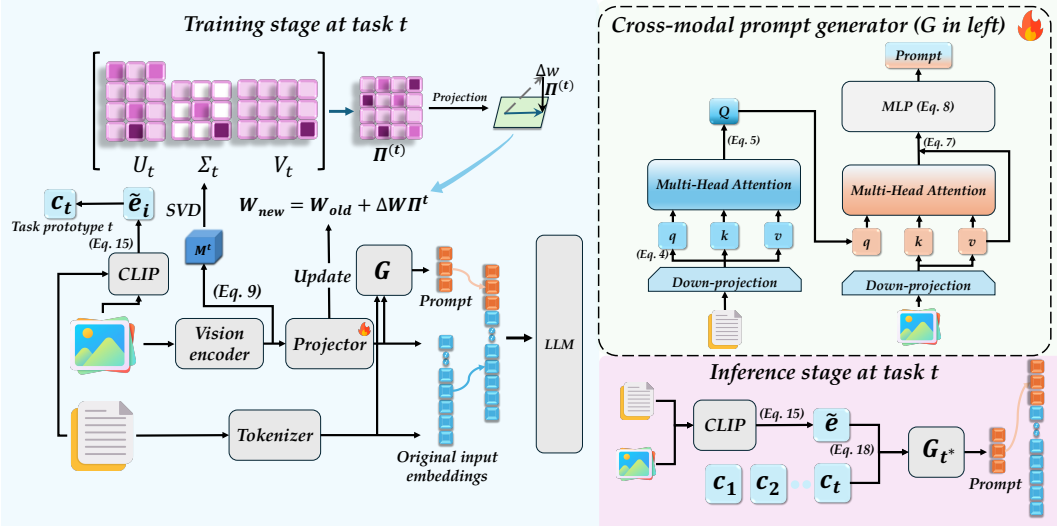


Figure 2: Illustration of DRAPE. **Left**: Training on task t. A task-specific generator synthesizes soft prompts, while the shared visual projector is regularized by projection onto the complement of the retained principal subspace. Feature statistics $\mathbf{M}^{(t)}$ are decomposed via SVD to obtain a projection matrix $\mathbf{\Pi}^{(t)}$ for the next task. After training, a task prototype $\mathbf{c}_t$ is registered in a frozen CLIP embedding space. **Top-Right**: Cross-modal prompt generator. It summarizes instruction tokens into prompt queries and cross-attends to visual features to synthesize instance-specific soft prompts. **Bottom-Right**: Inference without task labels. The router compares the current input against stored prototypes and selects the corresponding generator G_{t^*}.

MoE with LoRA Experts: An alternative paradigm [2] focuses on parameter-efficient fine-tuning via Low-Rank Adaptation (LoRA). To prevent forgetting, these methods allocate task-specific LoRA experts and freeze them after learning each task. During inference, a routing mechanism dynamically blends these historical experts. For a given input, the dynamically assembled parameter Θ_{add} is computed as:

$$\Theta_{add} = \sum_{k=1}^{t} \omega_k \Theta_k, \tag{2}$$

where Θ_k represents the LoRA parameters optimized for task k, and ω_k is the routing weight predicted by a gating network based on current input features.

Discussion. Both paradigms provide practical solutions for MCIT, but they also expose trade-offs. For prompt tuning, the reliance on a discrete prompt pool $\mathcal{P}$ can limit how flexibly the adaptation responds to within-task variation. For MoE with LoRA experts, the model gains stronger task-level specialization, but its behavior is still governed by routing and recombination among stored task-specific components. These approaches can be effective, yet they may be less suitable when subtle multimodal differences call for more input-specific adaptation. This motivates exploring whether MCIT can benefit from dynamically synthesizing prompts for each sample while still preserving historical knowledge through lightweight constraints.

4 DRAPE: Dynamic Cross-Modal Prompt Generation

We propose DRAPE, a prompt-learning framework for MCIT that moves from static component reuse toward input-conditioned prompt synthesis. As illustrated in Figure 2, DRAPE contains three key components. First, a cross-modal prompt generator produces instance-specific soft prompts conditioned on both the visual and textual features of each input. Second, null-space gradient projection constrains updates to the shared visual projector so as to reduce interference with previously observed feature subspaces. Third, a CLIP-guided prototype router selects the most relevant task-specific generator at inference. In the main experiments, the router stores one learned prototype per task, built from fused CLIP text-image features.

4.1 Cross-Modal Prompt Generation

The core of DRAPE is a cross-modal prompt generator G_t that, given an input's projected visual features $\mathbf{w} \in \mathbb{R}^{m\times d}$ and textual instruction embeddings $\mathbf{u} \in \mathbb{R}^{s\times d}$, synthesizes L_p instance-specific soft prompt vectors $\mathbf{P}_i = G_t(\mathbf{w}_i, \mathbf{u}_i) \in \mathbb{R}^{L_p\times d}$. The subscript t indicates that a separate task-specific generator is maintained for each task. During training, the generator masks out image placeholders, padding tokens, and answer positions so that prompt synthesis depends on the non-target instruction context rather than the supervised response. In our default configuration, the generation process uses two stages: instruction-aware query initialization followed by vision-guided prompt synthesis.

Instruction-Aware Query Initialization. The textual instruction encodes the task intent (e.g., "describe this image" vs. "what is the OCR text") and should guide how visual information is extracted. We first project the masked instruction embeddings into a latent space via a linear mapping $f_u : \mathbb{R}^d \to \mathbb{R}^H$:

$$\mathbf{h}^u = f_u(\mathbf{u}) \in \mathbb{R}^{s\times H}, \tag{3}$$

where H is the generator's hidden dimension. In the default 'segment' pooling mode, the instruction sequence is partitioned into L_p roughly equal segments $\{\mathcal{S}_1, \ldots, \mathcal{S}_{L_p}\}$, and a masked average pooling is computed within each segment:

$$\bar{\mathbf{h}}_p = \frac{\sum_{j\in\mathcal{S}_p} \mathbf{h}^u_j \cdot \mathbb{1}[j \text{ is valid}]}{\max\big(|\{j \in \mathcal{S}_p : j \text{ is valid}\}|,\, 1\big)}, \quad p = 1, \ldots, L_p, \tag{4}$$

yielding an initial pooled query matrix $\bar{\mathbf{H}} \in \mathbb{R}^{L_p\times H}$. These pooled summaries are then used as queries in a multi-head attention layer over the full instruction features:

$$\mathbf{Q} = \mathrm{LN}\big(\mathrm{MHA}(\bar{\mathbf{H}},\, \mathbf{h}^u,\, \mathbf{h}^u)\big) \in \mathbb{R}^{L_p\times H}, \tag{5}$$

where MHA denotes multi-head attention with a key padding mask that ignores invalid positions, and LN denotes layer normalization. The resulting $\mathbf{Q}$ contains L_p instruction-aware query vectors that are passed to the visual cross-attention stage.

Vision-Guided Prompt Synthesis. With the instruction-derived queries in hand, we extract task-relevant visual information through cross-modal attention. The projected visual features $\mathbf{w}$ are first mapped to the latent space:

$$\mathbf{h}^v = f_v(\mathbf{w}) \in \mathbb{R}^{m\times H}, \tag{6}$$

where $f_v : \mathbb{R}^d \to \mathbb{R}^H$ is a linear projection. The instruction queries $\mathbf{Q}$ then attend to the visual features via cross-modal attention:

$$\mathbf{R} = \mathrm{LN}(\mathbf{Q} + \mathrm{MHA}(\mathbf{Q},\, \mathbf{h}^v,\, \mathbf{h}^v)) \in \mathbb{R}^{L_p\times H}. \tag{7}$$

A dropout layer is applied to $\mathbf{R}$ for regularization. Finally, a two-layer MLP projects the fused representations back to the LLM's embedding space, producing the instance-specific soft prompts:

$$\mathbf{P}_i = f_{\text{head}}(\mathbf{R}) \in \mathbb{R}^{L_p\times d}, \tag{8}$$

where f_{head} maps $\mathbb{R}^H \to \mathbb{R}^{2H} \to \mathbb{R}^d$ with a GELU activation in between. The instance-specific soft prompts are prepended to the multimodal input sequence, yielding $\mathbf{z}' = [\mathbf{P}_i;\, \mathbf{z}]$, and the LLM generates the response via $p_\theta(\mathbf{y} \mid \mathbf{z}')$.

Discussion. This design choice reflects an asymmetry between modalities: the instruction indicates *which* information is needed, whereas the image provides the content that must be adapted to. By initializing prompt queries from the instruction and refining them through visual cross-attention, DRAPE performs input-conditioned prompt adaptation instead of selecting from a fixed pool of historical components. The implementation also supports mean-pooling, static-prompt, and learnable-query variants for additional analysis beyond the default setting.

4.2 Null-Space Gradient Projection for Projector Preservation

While each task maintains its own generator, the visual projector $\pi(\cdot)$ is shared across all tasks and continuously updated during sequential training. Updates from later tasks may therefore alter the visual-to-language mappings used by earlier tasks and contribute to catastrophic forgetting. To mitigate this effect, we constrain the gradient updates of π using an approximate null-space projection derived from the empirical feature subspaces observed in previous tasks.

Concretely, during training on task $\mathcal{D}_t$, forward hooks attached to each linear layer of π collect the input features $\tilde{\mathbf{v}}$ passing through that layer. The statistics and projections below are maintained per linear layer, with layer indices omitted for clarity. After training completes, these features are merged with the statistics accumulated from previous tasks to form a cumulative second-moment matrix:

$$\mathbf{M}^{(t)} = \frac{N^{(t-1)}\,\mathbf{M}^{(t-1)} + \sum_i \tilde{\mathbf{v}}_i \tilde{\mathbf{v}}_i^\top}{N^{(t)}}\,, \quad N^{(t)} = N^{(t-1)} + N_t\,, \tag{9}$$

where N_t is the number of feature vectors collected from $\mathcal{D}_t$ for the corresponding layer and $\mathbf{M}^{(0)} = \mathbf{0}$. We then decompose $\mathbf{M}^{(t)}$ via singular value decomposition:

$$\mathbf{M}^{(t)} = U\,\Sigma\,V^\top, \quad \Sigma = \mathrm{diag}(\sigma_1 \geq \cdots \geq \sigma_{d_v}). \tag{10}$$

The leading singular directions provide an empirical estimate of the dominant feature subspace used by the tasks observed so far. We identify the effective rank r as the smallest index satisfying:

$$\frac{\sum_{k=1}^{r} \sigma_k}{\sum_{k=1}^{d_v} \sigma_k} \geq \epsilon\,, \tag{11}$$

where ϵ is the energy-retention threshold. To preserve at least one update direction, we enforce $r < d_v$. The directions outside the retained subspace form the complementary basis $V_\perp = V_{[:,\,r+1:d_v]}$, and we construct the projection matrix:

$$\mathbf{\Pi}^{(t)} = V_\perp\,V_\perp^\top \in \mathbb{R}^{d_v \times d_v}. \tag{12}$$

During training on task $\mathcal{D}_{t+1}$, $\mathbf{\Pi}^{(t)}$ is applied to the gradients of each weight matrix in π via registered gradient hooks. For a weight matrix $W \in \mathbb{R}^{d \times d_v}$, the projected gradient is:

$$\nabla W' = \nabla W \mathbf{\Pi}^{(t)} = \nabla W V_\perp V_\perp^\top. \tag{13}$$

This operation restricts the projector update to the complementary subspace of the dominant feature directions accumulated from previous tasks. To see why this helps preserve earlier-task mappings, consider an earlier-task input feature $\tilde{\mathbf{v}}^{\mathrm{old}}$ to the same linear layer. If this feature is well captured by the retained principal subspace, then its component in the complementary subspace is small, i.e., $V_\perp^\top \tilde{\mathbf{v}}^{\mathrm{old}} \approx \mathbf{0}$. For an idealized gradient descent step with learning rate η, let $\Delta\mathbf{o}^{\mathrm{old}}$ denote the first-order change in the layer output for this old feature. Using the projected gradient in Eq. (13), we have:

$$\Delta\mathbf{o}^{\mathrm{old}} = -\eta \nabla W' \tilde{\mathbf{v}}^{\mathrm{old}} = -\eta \nabla W V_\perp \left(V_\perp^\top \tilde{\mathbf{v}}^{\mathrm{old}}\right) \approx \mathbf{0}. \tag{14}$$

Thus, when previous-task features mainly lie in the retained principal subspace, the projected update has limited effect on their projected representations. In this way, the projection helps reduce forgetting in the shared projector while still allowing updates along complementary directions for new tasks.

Discussion. The energy threshold ϵ determines how many dominant feature directions from previous tasks are retained. A larger ϵ preserves more historical feature directions, making subsequent projector updates less likely to overwrite visual-to-language mappings used by earlier tasks. In contrast to replay-based methods, this strategy stores only compact second-moment statistics for the shared projector. Thus, it provides a lightweight complement to the task-specific generators for reducing forgetting without retaining past training samples.

4.3 CLIP-Guided Prototype Routing

Since DRAPE maintains a separate generator G_t for each task, a routing mechanism is required at inference to select the appropriate generator without task labels. We use frozen CLIP embeddings as a shared semantic space for routing. In particular, the visual features are directly reused from the CLIP-based vision tower of LLaVA, so the router does not require loading an additional CLIP encoder, introducing only marginal extra cost for text-side encoding.

Task Prototype Computation. After learning each task, we register one compact prototype in the CLIP space. The routing feature is constructed by concatenating the normalized CLIP text embedding of the instruction and the normalized CLIP image embedding:

$$\tilde{\mathbf{e}}_i = \mathrm{norm}([\xi(\mathbf{q}_i);\, \gamma(\mathbf{v}_i)])\,. \tag{15}$$

During training, fused embeddings from the current task are cached temporarily and used to register the task prototype once training on $\mathcal{D}_t$ is finished. Let $\{\tilde{\mathbf{e}}_i\}_{i=1}^{N_t}$ denote the cached embeddings of task t. We initialize the prototype $\mathbf{c}_t$ by the normalized mean of these embeddings. For $t > 1$, we further refine $\mathbf{c}_t$ with a classification objective that pulls current-task embeddings closer to $\mathbf{c}_t$ than to previously stored prototypes. For a minibatch $\{\tilde{\mathbf{e}}_i\}_{i=1}^{B}$ and earlier prototypes $\{\mathbf{c}_s\}_{s=1}^{t-1}$, the loss is:

$$\mathcal{L}_{\text{cls}} = -\frac{1}{B}\sum_{i=1}^{B}\log\frac{\exp(\cos(\tilde{\mathbf{e}}_i,\mathbf{c}_t)/\tau)}{\exp(\cos(\tilde{\mathbf{e}}_i,\mathbf{c}_t)/\tau)+\sum_{s=1}^{t-1}\exp(\cos(\tilde{\mathbf{e}}_i,\mathbf{c}_s)/\tau)}, \tag{16}$$

where τ is the temperature parameter and $\cos(\cdot,\cdot)$ denotes cosine similarity. For the first task, no previous prototype is available, so Eq. 16 is not applied and $\mathbf{c}_1$ is simply registered as the normalized mean prototype. After registration, the cached instance-level embeddings are discarded. As a result, only one compact prototype is retained for each learned task.

Inference Routing. Given a test sample, we compute its routing feature $\tilde{\mathbf{e}}$ in the same CLIP space and score each task by cosine similarity:

$$s_t = \cos(\tilde{\mathbf{e}}, \mathbf{c}_t). \tag{17}$$

The router selects

$$t^* = \arg\max_{t\in\{1,\dots,T\}} s_t, \tag{18}$$

and activates the corresponding generator G_{t^*} to produce the prompt.

Discussion. The router is lightweight: it stores only one prototype per task and does not rely on replay samples or historical feature banks. Moreover, by reusing the CLIP-based vision tower already available in LLaVA, it avoids introducing an additional CLIP model. Using fused text-image embeddings makes routing depend on both instruction semantics and visual content, while the prototype update keeps each task representation compact yet discriminative.

4.4 Summary of DRAPE

DRAPE addresses multimodal continual instruction tuning through three complementary mechanisms: a cross-modal prompt generator for input-conditioned prompt synthesis, a projection-based constraint that reduces interference in the shared visual projector, and a CLIP-guided prototype router for practical task-specific generator selection without task labels.

Training. When learning task $\mathcal{D}_t$, only the current generator G_t and the shared visual projector π are updated. The vision encoder ϕ, the LLM backbone θ, and all historical generators $\{G_1,\dots,G_{t-1}\}$ remain frozen. The training objective is the standard autoregressive language modeling loss optimized over the tunable parameters $\Theta_t = \{G_t, \pi\}$:

$$\mathcal{L}_t = -\sum_{j=1}^{L}\log\, p_\theta(y_j \mid [\mathbf{P}_i;\, \mathbf{z}],\, \mathbf{y}_{<j})\,, \tag{19}$$

where $\mathbf{P}_i = G_t(\mathbf{w}_i, \mathbf{u}_i)$ is the synthesized prompt. For $t > 1$, the null-space projection in Eq. (13) is applied to the gradients of π via backward hooks. After completing $\mathcal{D}_t$, three post-training steps are executed: (1) the current generator G_t is frozen; (2) the cumulative second-moment matrix $\mathbf{M}^{(t)}$ is updated and the new projection matrix $\mathbf{\Pi}^{(t)}$ is computed for the next task; and (3) the task prototype $\mathbf{c}_t$ is registered in the router from the current task's temporary fused CLIP embeddings, after which those per-instance embeddings are discarded.

Inference. Given a test input $(\mathbf{v}, \mathbf{q})$ without task labels, the router computes the fused CLIP feature $\tilde{\mathbf{e}}$, selects t^* by prototype similarity, and activates G_{t^*} to synthesize $\mathbf{P}_i = G_{t^*}(\mathbf{w}_i, \mathbf{u}_i)$. The frozen LLM then generates the response from $[\mathbf{P}_i;\, \mathbf{z}]$.

5 Experiments

5.1 Implementation Details

Datasets. We conduct experiments on two MCIT benchmarks. The first is CoIN [2], which consists of eight sequential multimodal tasks: ScienceQA [28], TextVQA [35], ImageNet [5], GQA [16],

Table 1: Main results on the CoIN benchmark with LLaVA-v1.5-7B as the backbone (higher is better). The best and second-best values are marked in **bold** and underline, respectively.

Methods	ScienceQA	TextVQA	ImageNet	GQA	VizWiz	Grounding	VQAv2	OCR-VQA	Average
Finetune	26.00	25.38	28.51	33.07	26.52	0.10	40.00	52.92	29.06
CODA-Prompt [36]	58.15	50.16	24.04	54.33	48.94	17.83	55.86	54.42	45.46
DualPrompt [43]	56.40	47.12	34.96	42.03	44.14	12.01	54.43	53.36	43.05
L2P [44]	54.42	46.04	30.36	57.09	42.19	9.38	50.45	54.03	42.99
MoELoRA [2]	47.34	32.91	38.73	37.15	42.48	0.97	42.77	57.50	37.48
Continual LLaVA [1]	58.67	49.99	57.66	**62.53**	42.32	16.25	64.33	**74.91**	53.33
ModalPrompt [51]	68.42	56.40	41.13	61.11	50.13	36.69	**66.90**	59.68	55.06
ProgLoRA [50]	**74.84**	51.83	83.90	49.93	53.87	31.19	62.71	64.44	59.09
DRAPE **(Ours)**	70.67	**59.61**	**94.16**	61.37	**54.43**	**67.92**	66.53	65.11	**67.48**

Table 2: Main results on the UCIT benchmark with LLaVA-v1.5-7B as the backbone (higher is better). The best and second-best values are marked in **bold** and underline, respectively.

Methods	ImgNet-R	ArxivQA	VizWiz	IconQA	CLEVR	Flickr30k	Average
Zero-shot	16.27	53.73	38.39	19.20	20.63	41.88	-
LoRA-FT [13]	58.03	77.63	44.39	67.40	61.77	**58.22**	61.24
O-LoRA [41]	77.50	78.07	44.50	63.13	64.73	58.16	64.35
MoELoRA [2]	70.07	77.70	44.69	50.03	54.03	57.34	58.98
ModalPrompt [51]	51.07	87.27	48.11	39.23	46.57	42.93	52.53
CL-MoE [15]	66.33	77.00	44.78	51.87	53.53	57.42	58.49
HiDe [9]	84.03	90.73	44.43	58.93	41.37	54.25	62.29
SEFE [3]	80.83	78.00	47.01	**69.63**	**65.83**	57.92	66.54
DRAPE **(Ours)**	**85.07**	**91.60**	**55.68**	62.53	65.77	55.82	**69.41**

VizWiz [11], Grounding (RefCOCO) [19, 29], VQAv2 [8], and OCR-VQA [30]. The second is UCIT [9], which contains six sequential tasks: ImageNet-R [12], ArxivQA [21], VizWiz-caption [11], IconQA [27], CLEVR-Math [22], and Flickr30k [31]. Together, these two benchmarks let us evaluate our method in both a widely used MCIT setting and a cleaner setting with reduced data-overlap concerns.

Comparison Methods. We compare DRAPE with classic prompt-based continual learning approaches, including CODA-Prompt [36], DualPrompt [43], and L2P [44], as well as recent MCIT baselines, including MoELoRA [2], Continual LLaVA [1], ModalPrompt [51], and ProgLoRA [50]. For UCIT, we additionally include LoRA-FT [13], O-LoRA [41], CL-MoE [15], HiDe [9], and SEFE [3]. We also report standard sequential fine-tuning as a forgetting-heavy lower bound.

Training Setup. We adopt LLaVA-v1.5-7B [23] as the backbone MLLM, with CLIP-ViT-L/14-336 [32] as the vision encoder and a two-layer MLP with GELU activation as the visual projector. We conduct all experiments using Prism [37]. For the proposed generator, we set the hidden dimension to $H = 512$ and the prompt length to $L_p = 10$. The null-space threshold is $\epsilon = 0.99$. Each task is trained for one epoch with per-device batch size 4 and gradient accumulation steps 2. We use cosine learning-rate decay with warmup ratio 0.03, setting the peak learning rates to 2×10^{-4} for the generator and 2×10^{-5} for the projector. All experiments are conducted on 4 NVIDIA RTX 4090 GPUs with DeepSpeed ZeRO-2 and `bf16` precision.

Evaluation Metrics. Following [57, 56, 2], we denote by $\mathcal{A}_{s,t}$ the performance on task s after training up to task t, with T total tasks. The primary metric is the final average accuracy, defined as $\bar{\mathcal{A}} = \frac{1}{T}\sum_{s=1}^{T} \mathcal{A}_{s,T}$. We also report Backward Transfer (B) and Mean Accuracy (M), with their detailed formulations provided in Appendix A.

5.2 Benchmark Comparison

Table 1 reports the benchmark comparison on CoIN, where DRAPE achieves the highest average accuracy among all compared methods (**67.48**), outperforming ProgLoRA (59.09) by 8.39 points and obtaining the best results on four tasks: TextVQA, ImageNet, VizWiz, and Grounding. Table 2

Table 3: Ablation study on the CoIN benchmark. Δ indicates the average accuracy change relative to the full model. The best results are shown in **bold**.

Variants	ScienceQA	TextVQA	ImageNet	GQA	VizWiz	Grounding	VQAv2	OCR-VQA	Average	Δ
DRAPE (Full)	**70.67**	**59.61**	**94.16**	61.37	**54.43**	**67.92**	66.53	65.11	**67.48**	-
w/o Cross-Modal Attn.	69.39	59.44	94.14	61.11	47.77	67.87	66.55	64.22	66.31	-1.17
w/o Null-Space Proj.	69.94	55.78	83.62	**61.48**	49.83	65.43	**67.43**	**65.38**	64.86	-2.62

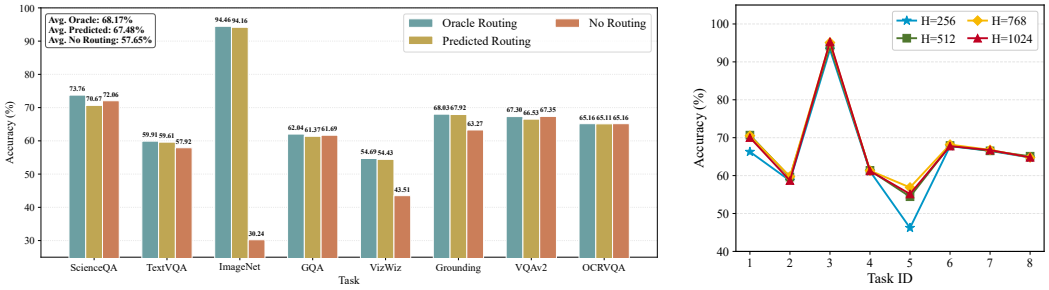


Figure 3: Routing ablation and generator hidden-dimension sensitivity on the CoIN benchmark. Left: comparison among oracle routing, learned prototype routing, and a no-routing variant that always uses the last-task generator. Right: per-task final accuracy under different generator hidden dimensions H.

further shows that this advantage generalizes to UCIT, where DRAPE again achieves the best average performance (**69.41**), surpassing the strongest baseline SEFE (66.54) by 2.87 points and ranking first on ImgNet-R, ArxivQA, and VizWiz. These results show that the gains of DRAPE are robust across both the standard CoIN benchmark and the cleaner UCIT setting. Stage-wise forgetting analysis using Backward Transfer and Mean Accuracy is provided in Appendix C.

5.3 Further Analysis

Ablation Study. To evaluate the contribution of each core component in DRAPE, we conduct controlled ablations in Table 3. Removing the vision cross-attention stage ("w/o Cross-Modal Attn.") reduces the average accuracy from 67.48 to 66.31, a drop of 1.17 points, with the clearest degradation appearing on visually demanding tasks such as VizWiz and OCR-VQA. Disabling null-space gradient projection ("w/o Null-Space Proj.") causes a larger drop to 64.86, *i.e.*, 2.62 points below the full model, with especially visible declines on ImageNet, TextVQA, VizWiz, and Grounding. This pattern is consistent with the intended role of the projector constraint: it does not eliminate forgetting entirely, but it helps reduce cross-task interference during sequential updates.

Routing Behavior and Generator Hidden Dimension. Figure 3 presents two additional analyses. The left panel shows that our learned prototype router achieves an average accuracy of 67.48, closely approaching the oracle upper bound (68.17) and vastly outperforming the no-routing baseline (57.65). This advantage is most prominent on ScienceQA and ImageNet, where the router effectively prevents the severe task mismatches seen in the baseline. To further illustrate this, we provide case studies in Appendix H, showing that the advantage of DRAPE is especially clear on samples requiring fine-grained instance-level adaptation. The right panel evaluates the generator's hidden dimension (H). While $H = 256$ underperforms (especially on VizWiz), increasing H from 512 to 768 or 1024 yields only marginal, plateauing improvements. Overall, the prototype router proves highly effective, and a moderate capacity suffices; thus, we adopt $H = 512$ as our default setting.

Visualization. Figure 4 visualizes prompt-to-image attention maps on OCR-VQA examples, where *Baseline* denotes the static task-level prompt variant. By generating prompts conditioned on both textual instructions and visual features, DRAPE attends more selectively to query-relevant textual regions. For the year-recognition example, DRAPE focuses on the lower-left text containing *"2016 Calendar"*; for the book-type example, it attends to *"Engagement Calendar 2016"*, which supports the answer *"Calendars"*. Compared with the more diffuse baseline attention, these results suggest



Figure 4: Prompt-to-image attention visualizations on OCR-VQA examples. For each example, **Column 1** shows the original image, **Column 2** shows the baseline attention map, and **Column 3** shows the attention map produced by DRAPE. By generating prompts conditioned on both textual instructions and visual features, DRAPE attends more selectively to query-relevant textual evidence, such as the printed year in the left example and the calendar-related text in the right example.

that cross-modal prompt generation helps DRAPE capture visual evidence required by the current instruction. Additional qualitative case studies are provided in Appendix H.

6 Conclusion

In this paper, we study multimodal continual instruction tuning with a particular focus on how models adapt to diverse samples encountered over a task stream. We find that continual adaptation should operate not only across tasks, but also across distinct samples within the same task, since different query-image pairs may require substantially different grounding, reasoning, or text-understanding behaviors. To address this challenge, we propose DRAPE, a framework that replaces static task-level adaptation with instance-specific prompt generation conditioned on the current multimodal input. Extensive experiments on the CoIN and UCIT benchmarks demonstrate that DRAPE achieves strong performance across diverse multimodal tasks and maintains competitive continual learning behavior.

Limitations. While DRAPE is developed and validated in the prompt-tuning setting considered in this paper, we have not yet extended the same instance-specific adaptation idea to other lightweight adaptation paradigms such as LoRA. Exploring how instance-specific generation can be integrated with LoRA-style updates is an interesting direction for future work.

Acknowledgments

This work was supported in part by the Basic Research Program of Jiangsu (BK20251251), NSFC (62506160), JSTJ-2025-147, Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (No. JYB2025XDXM118), the 111 Center (No. B26023), and the Collaborative Innovation Center of Novel Software Technology and Industrialization. The authors declare no competing interests.

References

[1] Meng Cao, Yuyang Liu, Yingfei Liu, Tiancai Wang, Jiahua Dong, Henghui Ding, Xiangyu Zhang, Ian Reid, and Xiaodan Liang. Continual llava: Continual instruction tuning in large vision-language models. *arXiv preprint arXiv:2411.02564*, 2024.

[2] Cheng Chen, Junchen Zhu, Xu Luo, Heng T Shen, Jingkuan Song, and Lianli Gao. Coin: A benchmark of continual instruction tuning for multimodel large language models. *Advances in neural information processing systems*, 37:57817–57840, 2024.

[3] Jinpeng Chen, Runmin Cong, Yuzhi Zhao, Hongzheng Yang, Guangneng Hu, Horace Ip, and Sam Kwong. SEFE: Superficial and essential forgetting eliminator for multimodal continual instruction tuning. In *Proceedings of the 42nd International Conference on Machine Learning*, pages 7982–8001. PMLR, 2025.

[4] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. *Advances in neural information processing systems*, 36:49250–49267, 2023.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[6] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4): 128–135, 1999.

[7] Chendi Ge, Xin Wang, Zeyang Zhang, Hong Chen, Jiapei Fan, Longtao Huang, Hui Xue, and Wenwu Zhu. Dynamic mixture of curriculum LoRA experts for continual multimodal instruction tuning. In *Proceedings of the 42nd International Conference on Machine Learning*, pages 19011–19033. PMLR, 2025.

[8] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 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*, pages 6904–6913, 2017.

[9] Haiyang Guo, Fanhu Zeng, Ziwei Xiang, Fei Zhu, Da-Han Wang, Xu-Yao Zhang, and Cheng-Lin Liu. Hide-llava: Hierarchical decoupling for continual instruction tuning of multimodal large language model. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13572–13586, 2025.

[10] Haiyang Guo, Fanhu Zeng, Fei Zhu, Jiayi Wang, Xukai Wang, Jingang Zhou, Hongbo Zhao, Wenzhuo Liu, Shijie Ma, Da-Han Wang, et al. Continual learning for generative ai: From llms to mllms and beyond. *arXiv preprint arXiv:2506.13045*, 2025.

[11] Danna Gurari, Qing Li, Abigale J Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P Bigham. Vizwiz grand challenge: Answering visual questions from blind people. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3608–3617, 2018.

[12] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8340–8349, 2021.

[13] 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 *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[14] Tao Hu, Zhinuo Zhou, Xialiang Tong, De-Chuan Zhan, and Da-Wei Zhou. Recap: Retrieval-guided capability reuse for multimodal continual instruction tuning. *arXiv preprint arXiv:2609.37889*, 2026.

[15] Tianyu Huai, Jie Zhou, Xingjiao Wu, Qin Chen, Qingchun Bai, Ze Zhou, and Liang He. Cl-moe: Enhancing multimodal large language model with dual momentum mixture-of-experts for continual visual question answering. In *Proceedings of the computer vision and pattern recognition conference*, pages 19608–19617, 2025.

[16] Drew A Hudson and Christopher D Manning. 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*, pages 6700–6709, 2019.

[17] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910, 2017.

[18] Dahuin Jung, Dongyoon Han, Jihwan Bang, and Hwanjun Song. Generating instance-level prompts for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11847–11857, 2023.

[19] Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. Referitgame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 787–798, 2014.

[20] Minh Le, Anh Nguyen, Huy Nguyen, Chau Nguyen, Anh T Tran, and Nhat Ho. Revisit visual prompt tuning: The expressiveness of prompt experts. In *International Conference on Learning Representations*, pages 107257–107304, 2026.

[21] Lei Li, Yuqi Wang, Runxin Xu, Peiyi Wang, Xiachong Feng, Lingpeng Kong, and Qi Liu. Multimodal arxiv: A dataset for improving scientific comprehension of large vision-language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14369–14387, 2024.

[22] Adam Dahlgren Lindström and Savitha Sam Abraham. Clevr-math: A dataset for compositional language, visual and mathematical reasoning. In *Proceedings of the 16th International Workshop on Neural-Symbolic Learning and Reasoning as part of the 2nd International Joint Conference on Learning & Reasoning (IJCLR 2022), Cumberland Lodge, Windsor Great Park, UK, September 28-30, 2022*, pages 155–170. CEUR-WS.org, 2022.

[23] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems*, pages 34892–34916. Curran Associates, Inc., 2023.

[24] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26296–26306, 2024.

[25] Yiyang Liu, James Liang, Heng Fan, Wenhao Yang, Yiming Cui, Xiaotian Han, Lifu Huangg, Dongfang Liu, Qifan Wang, and Cheng Han. All you need is one: Capsule prompt tuning with a single vector. In *Advances in Neural Information Processing Systems*, pages 88139–88166. Curran Associates, Inc., 2025.

[26] Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. The flan collection: Designing data and methods for effective instruction tuning. In *International conference on machine learning*, pages 22631–22648. PMLR, 2023.

[27] Pan Lu, Liang Qiu, Jiaqi Chen, Tanglin Xia, Yizhou Zhao, Wei Zhang, Zhou Yu, Xiaodan Liang, and Song-Chun Zhu. Iconqa: A new benchmark for abstract diagram understanding and visual language reasoning. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021.

[28] Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521, 2022.

[29] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 11–20, 2016.

[30] Anand Mishra, Shashank Shekhar, Ajeet Kumar Singh, and Anirban Chakraborty. Ocr-vqa: Visual question answering by reading text in images. In *2019 international conference on document analysis and recognition (ICDAR)*, pages 947–952. IEEE, 2019.

[31] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649, 2015.

[32] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[33] Guangyuan Shi, Jiaxin Chen, Wenlong Zhang, Li-Ming Zhan, and Xiao-Ming Wu. Overcoming catastrophic forgetting in incremental few-shot learning by finding flat minima. *Advances in neural information processing systems*, 34:6747–6761, 2021.

[34] Yu-Cheng Shi, Zhen-Hao Xie, Jun-Tao Tang, and Da-Wei Zhou. Protoada: Prototype-guided adaptive adapter expansion and geometric consolidation for multimodal continual instruction tuning. *arXiv preprint arXiv:2606.02576*, 2026.

[35] Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8317–8326, 2019.

[36] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11909–11919, 2023.

[37] Jun-Tao Tang, Yu-Cheng Shi, Zhen-Hao Xie, and Da-Wei Zhou. Prism: A plug-in reproducible infrastructure for scalable multimodal continual instruction tuning. *arXiv preprint arXiv:2605.26110*, 2026.

[38] Jun-Tao Tang, Zhen-Hao Xie, Yu-Cheng Shi, and Da-Wei Zhou. Cram: Centroid-routing and adaptive moe for multimodal continual instruction tuning. *arXiv preprint arXiv:2606.02502*, 2026.

[39] Shengbang Tong, David Fan, Jiachen Li, Yunyang Xiong, Xinlei Chen, Koustuv Sinha, Michael Rabbat, Yann LeCun, Saining Xie, and Zhuang Liu. Metamorph: Multimodal understanding and generation via instruction tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17001–17012, 2025.

[40] Mingyang Wang, Heike Adel, Lukas Lange, Jannik Strötgen, and Hinrich Schütze. Rehearsal-free modular and compositional continual learning for language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 469–480, 2024.

[41] Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuan-Jing Huang. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10658–10671, 2023.

[42] Yifan Wang, Yafei Liu, Chufan Shi, Haoling Li, Chen Chen, Haonan Lu, and Yujiu Yang. Inscl: A data-efficient continual learning paradigm for fine-tuning large language models with instructions. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 663–677, 2024.

[43] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision*, pages 631–648. Springer, 2022.

[44] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 139–149, 2022.

[45] Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[46] Tongtong Wu, Linhao Luo, Yuan-Fang Li, Shirui Pan, Thuy-Trang Vu, and Gholamreza Haffari. Continual learning for large language models: A survey. *arXiv preprint arXiv:2402.01364*, 2024.

[47] Zhuofeng Wu, Sinong Wang, Jiatao Gu, Rui Hou, Yuxiao Dong, VG Vinod Vydiswaran, and Hao Ma. Idpg: An instance-dependent prompt generation method. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5507–5521, 2022.

[48] Xi Xiao, Yunbei Zhang, Xingjian Li, Tianyang Wang, Xiao Wang, Yuxiang Wei, Jihun Hamm, and Min Xu. Visual instance-aware prompt tuning. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 2880–2889, 2025.

[49] Zhen-Hao Xie, Jun-Tao Tang, Yu-Cheng Shi, Han-Jia Ye, De-Chuan Zhan, and Da-Wei Zhou. SAME: Stabilized mixture-of-experts for multimodal continual instruction tuning. In *Proceedings of the 43rd International Conference on Machine Learning*, pages 139659–139683. PMLR, 2026.

[50] Yahan Yu, Duzhen Zhang, Yong Ren, Xuanle Zhao, Xiuyi Chen, and Chenhui Chu. Progressive lora for multimodal continual instruction tuning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 2779–2796, 2025.

[51] Fanhu Zeng, Fei Zhu, Haiyang Guo, Xu-Yao Zhang, and Cheng-Lin Liu. ModalPrompt: Towards efficient multimodal continual instruction tuning with dual-modality guided prompt. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 12126–12141, Suzhou, China, 2025. Association for Computational Linguistics.

[52] Aimen Zerroug, Mohit Vaishnav, Julien Colin, Sebastian Musslick, and Thomas Serre. A benchmark for compositional visual reasoning. *Advances in neural information processing systems*, 35:29776–29788, 2022.

[53] Yuexiang Zhai, Shengbang Tong, Xiao Li, Mu Cai, Qing Qu, Yong Jae Lee, and Yi Ma. Investigating the catastrophic forgetting in multimodal large language models. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*, 2023.

[54] Duzhen Zhang, Yahan Yu, Jiahua Dong, Chenxing Li, Dan Su, Chenhui Chu, and Dong Yu. MM-LLMs: Recent advances in MultiModal large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 12401–12430, Bangkok, Thailand, 2024. Association for Computational Linguistics.

[55] Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Guoyin Wang, and Fei Wu. Instruction tuning for large language models: A survey. *ACM Comput. Surv.*, 58(7), 2026.

[56] Da-Wei Zhou, Hai-Long Sun, Jingyi Ning, Han-Jia Ye, and De-Chuan Zhan. Continual learning with pre-trained models: a survey. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 8363–8371, 2024.

[57] Da-Wei Zhou, Qi-Wei Wang, Zhi-Hong Qi, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Class-incremental learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):9851–9873, 2024.

[58] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

Appendix

In this appendix, we provide additional details and supplementary analyses for DRAPE, including evaluation metrics, null-space projection analysis, additional benchmark results, sensitivity analysis, efficiency comparison, routing diagnostics, qualitative case studies, descriptions of compared methods, detailed result matrices, and broader-impact discussion.

Section A provides the definitions of the continual evaluation metrics used in our experiments, including backward transfer and mean accuracy.

Section B analyzes why null-space gradient projection can reduce interference with previous tasks.

Section C reports stage-wise continual performance on CoIN and UCIT, analyzing forgetting behavior throughout sequential training.

Section D reports additional CoIN benchmark comparisons with recent MCIT baselines.

Section E analyzes the sensitivity of different methods to the number of prompts or LoRA experts.

Section F analyzes the training efficiency, storage overhead, and inference cost of DRAPE compared with representative baselines.

Section G presents routing diagnostics, including the row-normalized routing confusion matrix.

Section H provides qualitative case studies on GQA, VQAv2, and OCR-VQA to illustrate instance-specific prompt behavior.

Section I summarizes the compared baseline methods.

Section J summarizes the training and inference procedures of DRAPE.

Section K reports the detailed continual-learning result matrices on CoIN and UCIT.

Section L discusses the broader impacts.

A Details of Evaluation Metrics

We provide the detailed definitions of the continual evaluation metrics used in our experiments. Let $\mathcal{A}_{s,t}$ denote the performance on task s after training up to task t, where T is the total number of tasks.

Backward Transfer (BWT). BWT measures the performance drop on previously learned tasks and quantifies catastrophic forgetting at incremental stage t, defined as $B_t = \frac{1}{t-1}\sum_{s=1}^{t-1}(\mathcal{A}_{s,s} - \mathcal{A}_{s,t})$ for $t = 2, \ldots, T$. We adopt a forgetting-oriented sign convention for backward transfer, where lower values indicate less forgetting and negative values indicate positive backward transfer. A lower BWT indicates better resistance to catastrophic forgetting.

Mean Accuracy (MA). MA evaluates the average performance over all learned tasks at incremental stage t, defined as $M_t = \frac{1}{t}\sum_{s=1}^{t}\mathcal{A}_{s,t}$ for $t = 1, \ldots, T$. A higher MA indicates better continual learning ability throughout the task sequence.

For stage-wise continual evaluation, we report both metrics across incremental stages and further average them over $t = 2, \ldots, T$.

B Analysis of Null-Space Gradient Projection

We provide a first-order analysis to illustrate why the null-space gradient projection can reduce interference on previously learned tasks. We follow the notation in Section 4.2 and make the layer index explicit: $\mathbf{M}_\ell^{(t)}$, $V_{\ell,\perp}$, and $\mathbf{\Pi}_\ell^{(t)}$ denote the second-moment matrix, complementary basis, and projection matrix for projector layer ℓ, respectively. The same argument applies independently to each linear layer of the shared projector $\pi(\cdot)$.

Let $\tilde{\mathbf{v}}^{(\ell)} \in \mathbb{R}^{d_\ell}$ denote the input feature to layer ℓ. After observing tasks up to $\mathcal{D}_t$, the empirical second-moment matrix maintained for this layer is

$$\mathbf{M}_\ell^{(t)} = \hat{\mathbb{E}}_{\le t}\left[\tilde{\mathbf{v}}^{(\ell)}\tilde{\mathbf{v}}^{(\ell)\top}\right], \tag{20}$$

where $\hat{\mathbb{E}}_{\le t}$ denotes the empirical average over features collected from tasks up to t. Its singular value decomposition is

$$\mathbf{M}_\ell^{(t)} = U_\ell \Sigma_\ell V_\ell^\top, \quad \Sigma_\ell = \mathrm{diag}(\sigma_{\ell,1}, \ldots, \sigma_{\ell,d_\ell}), \tag{21}$$

with $\sigma_{\ell,1} \ge \cdots \ge \sigma_{\ell,d_\ell} \ge 0$. Following the energy criterion in Section 4.2, we split the right singular vectors into the retained principal subspace and its complement:

$$V_\ell = [V_{\ell,\parallel}, V_{\ell,\perp}], \tag{22}$$

where $V_{\ell,\parallel}$ contains the top r_ℓ directions and $V_{\ell,\perp}$ contains the remaining directions. Let $r_{\ell,\epsilon}$ denote the smallest rank satisfying the energy-retention criterion. In practice, we set $r_\ell = \min(r_{\ell,\epsilon}, d_\ell - 1)$ to preserve at least one update direction. With $\epsilon = 0.99$ and $d_\ell \ge 100$ in our setting, this cap does not affect the energy-retention criterion. Therefore, we have

$$\frac{\sum_{k=1}^{r_\ell} \sigma_{\ell,k}}{\sum_{k=1}^{d_\ell} \sigma_{\ell,k}} \ge \epsilon, \tag{23}$$

which implies

$$\sum_{k=r_\ell+1}^{d_\ell} \sigma_{\ell,k} \le (1-\epsilon) \sum_{k=1}^{d_\ell} \sigma_{\ell,k}. \tag{24}$$

Therefore, the empirical energy of previous-task features in the complementary subspace is bounded by

$$\hat{\mathbb{E}}_{\le t}\left[\|V_{\ell,\perp}^\top \tilde{\mathbf{v}}^{(\ell)}\|_2^2\right] = \mathrm{tr}\left(V_{\ell,\perp}^\top \mathbf{M}_\ell^{(t)} V_{\ell,\perp}\right) = \sum_{k=r_\ell+1}^{d_\ell} \sigma_{\ell,k} \le (1-\epsilon)\mathrm{tr}(\mathbf{M}_\ell^{(t)}). \tag{25}$$

In practice, we set $\epsilon = 0.99$, which retains most of the empirical feature energy from previous tasks. As a result, previous-task features are expected to have only small components in the complementary subspace. Intuitively, for an earlier-task feature $\tilde{\mathbf{v}}_{\mathrm{old}}^{(\ell)}$, this means

$$V_{\ell,\perp}^\top \tilde{\mathbf{v}}_{\mathrm{old}}^{(\ell)} \approx \mathbf{0}. \tag{26}$$

Now consider the weight matrix W_ℓ of layer ℓ. During training on the next task, the gradient is projected as

$$\nabla W_\ell' = \nabla W_\ell \mathbf{\Pi}_\ell^{(t)} = \nabla W_\ell V_{\ell,\perp} V_{\ell,\perp}^\top. \tag{27}$$

For an earlier-task feature $\tilde{\mathbf{v}}_{\mathrm{old}}^{(\ell)}$, the first-order change in the layer output induced by this projected update is

$$\Delta \mathbf{o}_{\mathrm{old}}^{(\ell)} = -\eta \nabla W_\ell V_{\ell,\perp} V_{\ell,\perp}^\top \tilde{\mathbf{v}}_{\mathrm{old}}^{(\ell)}, \tag{28}$$

where η is the learning rate. Since the projected update acts only through the complementary component of the old feature, we can rewrite it as

$$\Delta \mathbf{o}_{\mathrm{old}}^{(\ell)} = -\eta \nabla W_\ell V_{\ell,\perp} \left(V_{\ell,\perp}^\top \tilde{\mathbf{v}}_{\mathrm{old}}^{(\ell)}\right) \approx \mathbf{0}. \tag{29}$$

Equivalently, its magnitude can be bounded as

$$\|\Delta \mathbf{o}_{\mathrm{old}}^{(\ell)}\|_2 \le \eta \|\nabla W_\ell\|_2 \left\|V_{\ell,\perp}^\top \tilde{\mathbf{v}}_{\mathrm{old}}^{(\ell)}\right\|_2. \tag{30}$$

Thus, when earlier-task features are well captured by the retained principal subspace, their components in $V_{\ell,\perp}$ are small, and the projected update has a limited first-order effect on their layer outputs.

In the ideal case where $\tilde{\mathbf{v}}_{\mathrm{old}}^{(\ell)} \in \mathrm{span}(V_{\ell,\parallel})$, we have

$$V_{\ell,\perp}^\top \tilde{\mathbf{v}}_{\mathrm{old}}^{(\ell)} = \mathbf{0}, \tag{31}$$

and therefore the projected update does not change the layer output for that feature to first order. Overall, this analysis shows why null-space projection helps reduce first-order interference on earlier-task representations while preserving update directions for new-task adaptation.

Method	Immediate Avg. Accuracy	Final Avg. Accuracy
w/o Null-Space Projection	69.01	64.86
DRAPE	68.05	67.48

Table 4: Stability–plasticity analysis of null-space projection on the CoIN benchmark. Immediate accuracy is measured after learning each task, while final accuracy is measured after completing the entire task sequence.

B.1 Stability–Plasticity Analysis of Null-Space Projection

Null-space gradient projection (NSP) is designed to preserve historical multimodal representations by restricting updates along important feature directions. However, such constraints may potentially reduce the model's plasticity when learning new tasks. To analyze the trade-off between stability and plasticity, we compare DRAPE with a variant without NSP in terms of immediate learning performance and final performance after completing the whole task sequence. Specifically, the *Immediate Avg. Accuracy* denotes the average accuracy measured immediately after learning each task, which reflects the model's ability to acquire new knowledge. The *Final Avg. Accuracy* denotes the average accuracy measured after completing the entire task sequence, which reflects the retention of previously learned knowledge.

As shown in Table 4, removing NSP improves the immediate average accuracy from 68.05% to 69.01%, indicating that NSP introduces only a minor restriction on learning new tasks. However, without NSP, the final average accuracy decreases from 67.48% to 64.86%, demonstrating increased interference from subsequent task updates. These results show that NSP achieves a favorable stability–plasticity trade-off by preserving sufficient adaptation capacity for new tasks while improving knowledge retention throughout continual learning.

C Continual Learning Performance Across Tasks

We report stage-wise continual metrics on CoIN and UCIT to further analyze forgetting behavior during sequential training. Table 5 shows the results on CoIN. Compared with previous methods, DRAPE achieves the lowest average backward-transfer value and the highest average mean accuracy among the compared methods, with $B = \mathbf{1.50}$ and $M = \mathbf{68.92}$. Relative to the strongest baseline ModalPrompt, this corresponds to 2.22 points lower backward transfer and 12.82 points higher mean accuracy. These results indicate that DRAPE improves forgetting resistance while maintaining strong overall continual performance throughout the task stream.

Table 6 further reports the stage-wise continual metrics on UCIT. DRAPE again achieves the lowest average backward transfer, with $B = \mathbf{0.04}$, slightly outperforming ModalPrompt (0.06) and showing larger improvements over the remaining baselines. Its B_t values remain close to zero across incremental stages and even become negative at early stages, indicating limited forgetting and slight positive backward transfer. Meanwhile, DRAPE obtains an average mean accuracy of **<u>76.31</u>**, which is comparable to the best-performing baseline SEFE (**76.35**) and higher than the other compared methods. These results further show that DRAPE preserves previously acquired knowledge effectively while retaining competitive adaptation to newly introduced tasks.

Method	TextVQA		ImageNet		GQA		VizWiz		Grounding		VQAv2		OCR-VQA		*Average*	
	$B_2\downarrow$	$M_2\uparrow$	$B_3\downarrow$	$M_3\uparrow$	$B_4\downarrow$	$M_4\uparrow$	$B_5\downarrow$	$M_5\uparrow$	$B_6\downarrow$	$M_6\uparrow$	$B_7\downarrow$	$M_7\uparrow$	$B_8\downarrow$	$M_8\uparrow$	$B\downarrow$	$M\uparrow$
Finetune	44.30	44.14	65.53	32.52	64.42	22.75	51.98	25.55	67.08	5.71	43.89	23.65	35.61	29.06	53.26	26.20
CODA-Prompt	11.54	57.88	27.70	34.05	14.38	43.44	12.78	42.76	12.72	39.28	14.05	39.42	7.27	45.46	14.34	43.18
MoELoRA	41.31	43.13	52.47	34.08	32.76	41.71	33.81	37.71	41.41	25.59	30.80	34.34	26.12	37.48	36.95	36.29
ModalPrompt	6.55	64.50	4.40	56.34	3.16	57.63	4.51	54.15	3.98	50.96	2.02	54.07	1.41	55.06	3.72	56.10
ProgLoRA	17.21	59.92	6.78	73.60	17.60	60.71	12.27	61.35	18.31	50.60	9.56	57.66	7.53	59.09	12.75	60.42
DRAPE (Ours)	**3.37**	**65.11**	**2.48**	**74.37**	**1.32**	**71.51**	**1.07**	**68.07**	**0.83**	**68.08**	**0.78**	**67.80**	**0.66**	**67.48**	**1.50**	**68.92**

Table 5: Continual performance metrics at each incremental stage on the CoIN benchmark. B_t and M_t stand for *Backward Transfer* and *Mean Accuracy* at incremental stage t.

Method	ArxivQA		VizWiz		IconQA		CLEVR		Flickr30k		*Average*	
	$B_2 \downarrow$	$M_2 \uparrow$	$B_3 \downarrow$	$M_3 \uparrow$	$B_4 \downarrow$	$M_4 \uparrow$	$B_5 \downarrow$	$M_5 \uparrow$	$B_6 \downarrow$	$M_6 \uparrow$	$B \downarrow$	$M \uparrow$
LoRA-FT	1.37	90.83	8.81	75.77	13.55	71.04	15.28	68.40	18.78	61.24	11.56	73.46
O-LoRA	1.33	**92.40**	6.47	**78.26**	11.18	72.70	13.20	69.02	13.99	64.35	9.23	75.35
MoELoRA	1.16	91.44	5.85	77.52	11.78	65.66	10.31	65.68	14.63	58.98	8.74	71.85
ModalPrompt	0.07	69.20	0.00	62.20	0.04	56.45	0.04	54.57	**0.15**	52.53	0.06	58.99
CL-MoE	3.90	89.84	8.40	75.65	13.04	64.74	11.39	65.15	15.56	58.49	10.46	70.77
HiDe	2.04	90.55	1.53	78.18	4.80	74.08	6.44	67.94	9.20	62.29	4.80	74.61
SEFE	0.57	90.73	4.35	78.19	8.51	**74.17**	9.30	72.15	11.33	66.54	6.81	**76.35**
DRAPE (Ours)	**-0.14**	88.58	**-0.02**	77.59	**0.02**	73.82	**0.17**	**72.16**	0.16	**69.41**	**0.04**	76.31

Table 6: Continual performance metrics at each incremental stage on the UCIT benchmark. B_t and M_t denote the backward transfer and mean accuracy at incremental stage t.

Method	ScienceQA	TextVQA	ImageNet	GQA	VizWiz	Grounding	VQAv2	OCR-VQA	Average
ProgLoRA [50]	74.84	51.83	83.90	49.93	53.87	31.19	62.71	64.44	59.09
SEFE [3]	75.35	58.66	83.10	54.25	48.85	16.75	65.35	66.25	58.57
HiDe [9]	73.20	56.92	69.28	61.33	50.76	59.18	67.12	64.76	62.82
O-LoRA [41]	86.63	48.78	12.38	61.67	46.68	66.33	67.05	55.58	56.65
CL-MoE [15]	81.37	51.48	22.95	59.99	39.01	55.26	67.60	63.70	55.17
DRAPE (Ours)	70.67	59.61	94.16	61.37	54.43	67.92	66.53	65.11	**67.48**

Table 7: Additional comparison on the CoIN benchmark with LLaVA-v1.5-7B as the backbone. All task-wise scores correspond to the final accuracy after completing the entire task sequence. The best average accuracy is highlighted in bold.

D Additional CoIN Baseline Comparison

To provide a more comprehensive comparison on the CoIN benchmark, we include additional results of recent MCIT baselines under the CoIN setting. All task-wise scores correspond to the final accuracy after completing the entire task sequence. As shown in Table 7, DRAPE achieves the best average performance among the compared methods.

E Prompt/LoRA Number Sensitivity

We further analyze how the number of prompts or LoRA experts affects continual performance on the CoIN benchmark. Figure 5 illustrates the final average accuracy and the last-task accuracy when varying the number of prompts for each task and the number of MoE LoRA experts. Specifically, the x-axis values $\{2, 5, 10\}$ denote the number of prompts used for each task in prompt-based methods and the number of LoRA experts in MoELoRA. Generally, increasing the number of prompts or LoRA experts brings modest improvements, and the gains gradually saturate once the number becomes moderately large. For baselines like MoELoRA, CODA-Prompt, and ModalPrompt, increasing the corresponding number brings steady enhancements, whereas DRAPE remains relatively stable across the tested settings. Considering the trade-off between effectiveness and efficiency, we set the default number of prompts to $L_p = 10$, which achieves strong performance within the evaluated prompt budget.

F Efficiency Comparison

We analyze the efficiency of DRAPE from three aspects: training cost, storage overhead, and inference efficiency.

Training and storage overhead. Table 8 compares the training time and active trainable parameters with representative prompt-based and LoRA-based continual learning methods. Here, "Trainable Param" refers to the parameters updated during a single task update, including the current task-specific generator and the shared multimodal projector. DRAPE updates 32M parameters per task and achieves the lowest training time among all compared methods. For long-term sequential learning, DRAPE only stores one frozen task-specific generator for each learned task, which occupies approximately

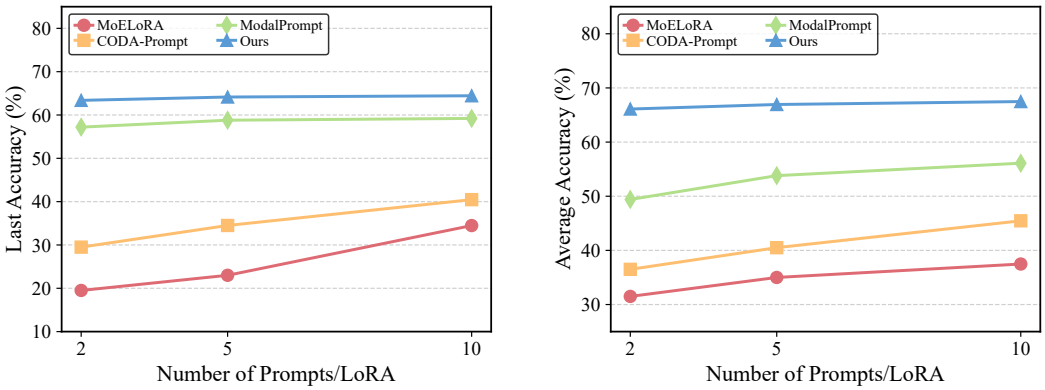


Figure 5: Impact of prompt and LoRA expert numbers on the CoIN benchmark. We vary the number of prompts for each task in prompt-based methods and the number of MoE LoRA experts in MoELoRA. Left: last-task accuracy. Right: final average accuracy.

Method	Training (Hour)	Trainable Param
MoELoRA	7.27	360M
CODA-Prompt	3.43	165M
HiDe	3.93	247M
ModalPrompt	2.54	20M
DRAPE (Ours)	1.59	32M

Table 8: Training efficiency comparison on CoIN.

Method	Prompt Construction (ms)	Latency (ms/sample)	Throughput (Tokens/s)	Memory (GB)
Zero-shot	–	182.01	33.22	14.35
SEFE	–	117.16	25.65	14.36
HiDe	–	622.78	3.21	15.09
Static Prompt	0.40	195.49	30.55	14.94
DRAPE (Ours)	2.18	197.99	30.21	15.24

Table 9: Inference overhead comparison on CoIN.

21.04 MB in our implementation. Therefore, even after learning 50 tasks, the additional storage requirement is about 1 GB, which is around one tenth of the full LLaVA backbone whose weights alone require more than 10 GB. These results demonstrate that DRAPE remains lightweight for continual task expansion.

Inference overhead. Since DRAPE generates instance-specific prompts conditioned on each image-instruction pair, we further evaluate the additional inference cost introduced by dynamic prompt generation. The prompt generator is executed only once before autoregressive decoding, after which DRAPE follows the standard autoregressive inference procedure with KV caching. Table 9 reports the prompt construction time, end-to-end latency, token throughput, and peak memory usage. To isolate the overhead introduced by instance-specific prompt generation, we include a static task-level prompt variant that uses the same backbone and training protocol but replaces input-conditioned prompt generation with learnable fixed task-level prompts. We also compare with representative MCIT baselines, including SEFE and HiDe. Compared with static prompting, DRAPE introduces only 1.78 ms additional prompt construction time and increases the end-to-end latency by 2.50 ms per sample. The token throughput remains nearly unchanged, decreasing from 30.55 to 30.21 tokens/s, while the peak memory increases by only 0.30 GB. Compared with existing MCIT baselines, DRAPE achieves a favorable efficiency-performance trade-off by providing finer-grained multimodal adaptation with limited additional computational and memory overhead.

G Routing Diagnostics

Figure 6 complements the routing ablation by showing where the learned router departs from oracle task selection. The dominant diagonal entries indicate that the router usually activates the correct task generator, which helps explain why learned routing remains close to the oracle-routing upper bound and far outperforms the no-routing variant in the main results. The largest off-diagonal errors reveal two main patterns. First, GQA and VQAv2 are strongly confused with each other, with GQA samples often routed to VQAv2 and VQAv2 samples often routed to GQA. Second, ScienceQA has a more dispersed routing pattern, with a noticeable fraction of samples routed to OCR-VQA and

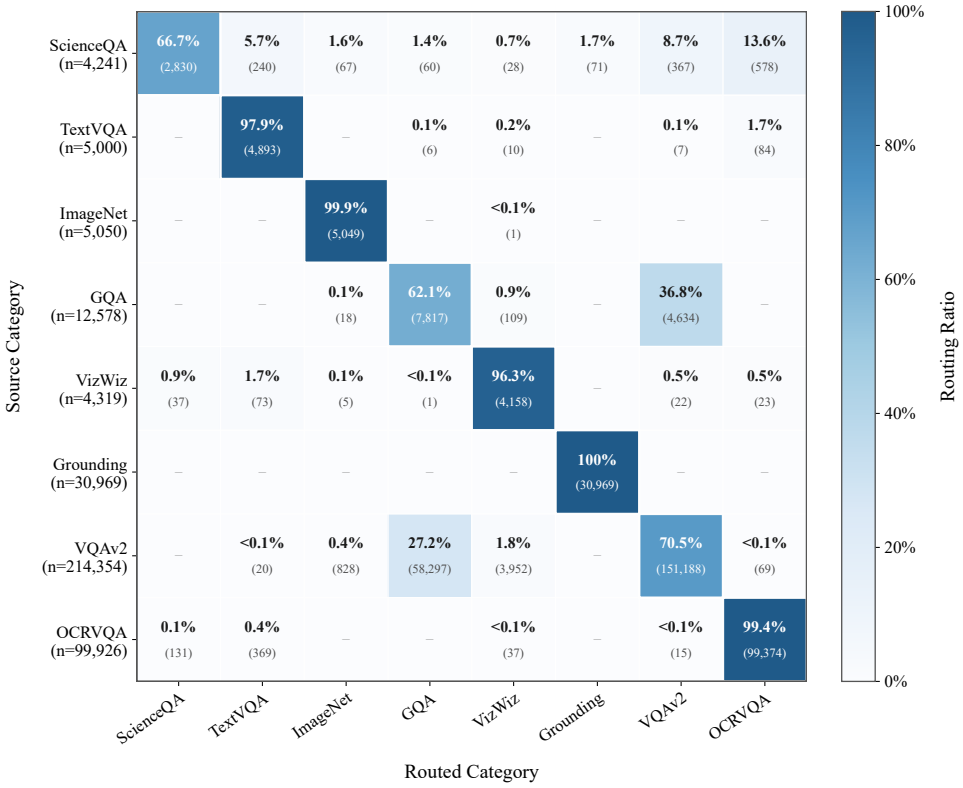


Figure 6: Row-normalized routing confusion matrix on the final task. Each row corresponds to a ground-truth task and sums to 100%. Diagonal entries indicate routing accuracy, while off-diagonal entries reveal the dominant misrouting patterns. The largest off-diagonal entries occur between GQA and VQAv2, and from ScienceQA to OCR-VQA and VQAv2.

VQAv2. These errors do not necessarily lead to proportional drops in final task performance, since the misrouted generators often correspond to semantically related visual-question-answering or text-reading behaviors. This helps explain why the learned router remains close to oracle routing despite imperfect task-level routing accuracy. Because task sizes differ substantially, the row-normalized view is more informative than raw confusion counts for cross-task comparison.

H Case Studies

To further examine the qualitative behavior of DRAPE beyond aggregate accuracy, we present representative case studies on GQA, VQAv2, and OCR-VQA. In all figures, *Baseline* denotes the static task-level prompt variant, while *Drape* denotes our instance-specific prompt variant. Each example reports the prediction of the two variants together with the ground-truth answer. The left example in each figure corresponds to a relatively simple case where both methods succeed, while the middle and right examples illustrate more challenging cases in which the static prompt fails but DRAPE produces the correct answer. These examples provide qualitative evidence that a single shared task-level prompt can be sufficient for standard inputs, but may become less reliable when the required adaptation depends on fine-grained query–image interactions.

GQA. Figure 7 shows three examples from GQA. In the left example, both variants correctly answer the query *"What device is sitting next to the mouse pad?"* with "Keyboard," suggesting that both prompt strategies can handle a direct object-grounding case. The middle example requires more precise spatial grounding: for the query *"Where is the skinny person standing?"*, the static prompt predicts the coarse spatial answer "Left," whereas DRAPE correctly identifies the supporting object, "Table." In the right example, the query *"Does the calf have brown color and large size?"* requires compositional attribute verification. The static variant incorrectly predicts "Yes," while DRAPE

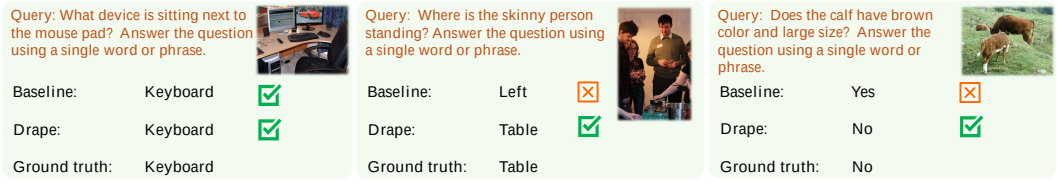


Figure 7: Case studies on GQA. The left example is a relatively simple case where both variants are correct. The middle and right examples require finer spatial grounding and compositional attribute verification, respectively. In these more challenging cases, the static task-level prompt fails, whereas the instance-specific prompt generated by DRAPE produces the correct answer.

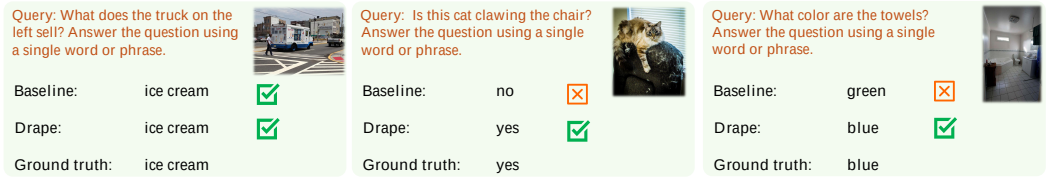


Figure 8: Case studies on VQAv2. The left example is a relatively simple case where both variants succeed. The middle and right examples require fine-grained local action recognition and attribute recognition, respectively. In these cases, the static task-level prompt produces incorrect predictions, whereas DRAPE remains aligned with the visual evidence relevant to the current query.

correctly answers "No." These cases suggest that instance-specific prompt generation can better accommodate shifts from simple grounding to finer spatial reasoning and compositional attribute judgment within the same task.

VQAv2. Figure 8 presents examples from VQAv2. In the left example, both variants correctly answer *"What does the truck on the left sell?"* with "ice cream," indicating that a static task-level prompt can be adequate for a straightforward open-domain VQA query. The middle example asks *"Is this cat clawing the chair?"*, which depends on fine-grained local action recognition. The static prompt predicts "no," whereas DRAPE correctly predicts "yes." In the right example, the query *"What color are the towels?"* requires local attribute recognition; the static prompt answers "green," while DRAPE correctly answers "blue." These examples indicate that the benefit of instance-specific prompting is not limited to relational reasoning, but also extends to subtle local visual evidence in general VQA.

OCR-VQA: same image, different queries. Figure 9 fixes the image and varies the query on the same book cover. This setting isolates the effect of query-dependent adaptation, since the visual input remains unchanged while the required reasoning behavior changes. In the left example, both variants correctly identify the author as "Edward De Bono." In the middle example, the query shifts to title extraction. The static prompt predicts "Unanswerable," whereas DRAPE recovers the full title *"Creativity Workout: 62 Exercises to Unlock Your Most Creative Ideas."* In the right example, the query asks for the book genre. The static prompt predicts the incorrect category *"Arts & Photography,"* while DRAPE correctly answers *"Health, Fitness & Dieting."* These results suggest that even for a fixed image, different textual queries may require different prompt behaviors, and that instance-specific prompt generation can better adapt to such query-dependent semantic shifts.

OCR-VQA: same query, different images. Figure 10 provides the complementary analysis by fixing the query type and varying the image. All examples ask for the book title. In the left example, both variants correctly recover the title from a relatively simple cover. In the middle example, the static prompt predicts the incomplete title *"2016 Cat Naps Daily Calendar,"* while DRAPE correctly recovers *"Cat Naps 2016 Boxed/Daily Calendar."* In the right example, the static prompt predicts "Unanswerable," whereas DRAPE correctly identifies *"OMG Facts 2016 Boxed/Daily Calendar."* This comparison suggests that even when the instruction form is fixed, the appropriate adaptation still depends on the specific visual layout and textual content of each image.

Overall observation. Across GQA, VQAv2, and OCR-VQA, we observe a consistent qualitative pattern. Static task-level prompts are often sufficient for relatively simple or canonical samples, but they can become brittle when the task requires fine-grained grounding, local visual recognition, compositional attribute verification, or OCR-style text extraction. By generating prompts conditioned

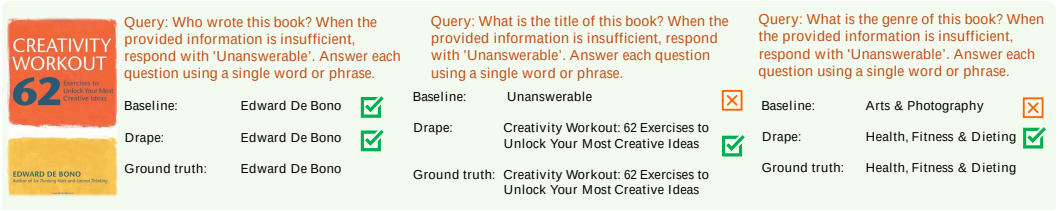


Figure 9: OCR-VQA case study with a fixed image and different queries. Although the visual input is unchanged, the required behavior varies across queries, ranging from author recognition to title extraction and genre identification. The static task-level prompt handles the easier query but becomes less reliable on the more demanding ones, whereas DRAPE adapts more effectively to the query-specific requirements.

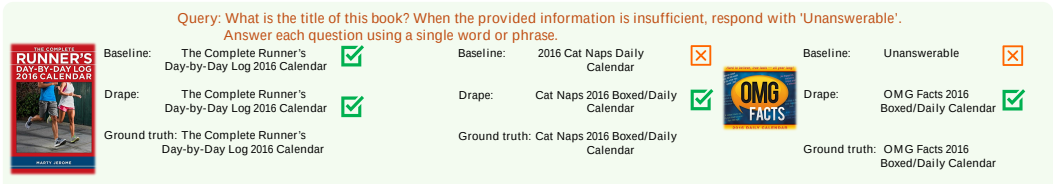


Figure 10: OCR-VQA case study with a fixed query type and different images. Both variants correctly recover the title in a relatively simple example, while the more challenging covers require finer OCR-style extraction. In these harder cases, the static task-level prompt produces an incomplete answer or falls back to "Unanswerable," whereas DRAPE recovers the correct full title.

on the current query–image pair, DRAPE provides a more flexible adaptation mechanism that better matches sample-specific multimodal demands. These case studies complement the quantitative results by illustrating how instance-specific prompt generation can improve within-task adaptation in multimodal continual instruction tuning.

I Comparing Methods

In this section, we provide brief descriptions of the methods compared in the paper.

Finetune sequentially fine-tunes the model on each incoming task without an explicit mechanism to preserve previous knowledge.

MoELoRA maintains multiple task-related LoRA experts together with a gating function, so that only part of the additional parameters are activated for each input. In this way, it aims to preserve previously acquired knowledge while adapting to the current sample through expert selection.

L2P maintains a prompt pool in memory space and retrieves relevant prompts for each input, thereby balancing task-invariant and task-specific knowledge through prompt selection.

DualPrompt introduces two types of prompts, namely general prompts and expert prompts, to model task-invariant and task-specific knowledge, respectively. These prompts are attached to different transformer layers to facilitate continual adaptation and knowledge retention.

CODA-Prompt learns a set of input-conditioned prompts for rehearsal-free continual learning and dynamically composes them during inference.

ModalPrompt extends prompt-based continual learning to the multimodal setting by introducing learnable prompts for vision-language adaptation while keeping most backbone parameters frozen.

ProgLoRA incrementally expands task-specific LoRA parameters across tasks and preserves previous knowledge by freezing historical modules. We follow the standard configuration in its released implementation and use the default LoRA hyperparameters.

LoRA-FT performs sequential low-rank adaptation without an explicit forgetting-mitigation mechanism.

Algorithm 1 Training of DRAPE for Multimodal Continual Instruction Tuning

Require: Task stream $\{\mathcal{D}_t\}_{t=1}^{T}$, model components $\phi, \psi, \theta, \pi, \{G_t\}_{t=1}^{T}$, CLIP encoders ξ, γ, energy threshold ϵ
Ensure: Trained projector π, frozen generators $\{G_t\}_{t=1}^{T}$, task prototypes $\{\mathbf{c}_t\}_{t=1}^{T}$

Initialize cumulative feature statistics $\mathbf{M}_\ell^{(0)} = \mathbf{0}$ and counts $N_\ell^{(0)} = 0$ for each projector layer ℓ
for $t = 1$ to T **do**
  Activate G_t and freeze historical generators
  for each mini-batch $\mathcal{B} \subset \mathcal{D}_t$ **do**
    Compute $\mathbf{w}_i = \pi(\phi(\mathbf{v}_i))$, $\mathbf{u}_i = \psi(\mathbf{q}_i)$, and $\mathbf{P}_i = G_t(\mathbf{w}_i, \mathbf{u}_i)$
    Accumulate projector-layer feature statistics and cache CLIP routing features $\tilde{\mathbf{e}}_i$
    Compute autoregressive loss $\mathcal{L}_t$ and backpropagate gradients
    if $t > 1$ **then**
      Project gradients of π: $\nabla W_\ell \leftarrow \nabla W_\ell \mathbf{\Pi}_\ell^{(t-1)}$
    end if
    Update G_t and π
  end for
  Freeze G_t and update cumulative statistics $\mathbf{M}_\ell^{(t)}$ and $N_\ell^{(t)}$
  Compute $\mathbf{\Pi}_\ell^{(t)}$ from $\mathbf{M}_\ell^{(t)}$ using SVD and threshold ϵ
  Initialize $\mathbf{c}_t$ as the normalized mean of cached CLIP features
  if $t > 1$ **then**
    Refine $\mathbf{c}_t$ using Eq. (16)
  end if
  Register $\mathbf{c}_t$ and discard cached CLIP features
end for

Algorithm 2 Inference of DRAPE without Task Labels

Require: Test sample $(\mathbf{v}, \mathbf{q})$, model components ϕ, ψ, θ, π, CLIP encoders ξ, γ, generators $\{G_t\}_{t=1}^{T}$, task prototypes $\{\mathbf{c}_t\}_{t=1}^{T}$
Ensure: Predicted answer $\hat{\mathbf{y}}$

Compute routing feature $\tilde{\mathbf{e}} = \mathrm{norm}([\xi(\mathbf{q}); \gamma(\mathbf{v})])$
Select generator $t^* = \arg\max_{t \in \{1,\ldots,T\}} \cos(\tilde{\mathbf{e}}, \mathbf{c}_t)$
Compute visual features $\mathbf{w} = \pi(\phi(\mathbf{v}))$ and text embeddings $\mathbf{u} = \psi(\mathbf{q})$
Generate instance-specific prompt $\mathbf{P} = G_{t^*}(\mathbf{w}, \mathbf{u})$
Generate prediction $\hat{\mathbf{y}}$ using θ conditioned on $[\mathbf{P}; \mathbf{w}; \mathbf{u}]$

O-LoRA reduces interference across tasks by imposing orthogonality constraints on LoRA updates. It aims to preserve previously learned knowledge while maintaining adaptation ability on new tasks.

CL-MoE is an expert-based continual tuning method that improves adaptation through task-specialized modules and expert routing. It dynamically selects or combines experts according to the current input.

HiDe is motivated by CKA similarity analysis, which suggests that top and lower transformer layers exhibit different representation patterns in continual learning. It hierarchically decouples adaptation by using task-specific LoRA expansion with dual-modality anchor matching in higher layers, while fusing LoRAs in lower layers to preserve shared knowledge without router training.

SEFE addresses forgetting through answer-style diversification and RegLoRA. RegLoRA regularizes the top-$M\%$ elements of LoRA update matrices to preserve critical historical knowledge during continual adaptation.

J Pseudocode of DRAPE

We summarize the training and inference procedures of DRAPE in Algorithms 1 and 2. During sequential training, DRAPE learns a task-specific prompt generator while preserving previously learned generators. The shared multimodal projector is updated using layer-wise null-space gradient

Table 10: Detailed continual instruction tuning results matrices for different methods on the CoIN benchmark.

Finetune	ScienceQA	TextVQA	ImageNet	GQA	VizWiz	Grounding	VQAv2	OCR-VQA
ScienceQA	82.45							
TextVQA	38.15	50.14						
ImageNet	0.96	0.58	96.03					
GQA	13.91	15.78	5.67	55.65				
VizWiz	8.46	25.17	4.60	38.12	51.42			
Grounding	0.00	0.00	0.00	0.27	0.00	34.00		
VQAv2	9.10	27.58	6.62	43.92	19.10	0.03	59.17	
OCR-VQA	26.00	25.38	28.51	33.07	26.52	0.10	40.00	52.92

L2P	ScienceQA	TextVQA	ImageNet	GQA	VizWiz	Grounding	VQAv2	OCR-VQA
ScienceQA	72.83							
TextVQA	68.07	57.16						
ImageNet	32.05	26.73	39.43					
GQA	47.53	46.02	18.03	60.47				
VizWiz	65.94	37.68	1.72	56.29	47.90			
Grounding	5.74	42.96	33.92	39.44	39.64	1.87		
VQAv2	32.57	48.65	7.41	47.32	34.52	8.17	59.40	
OCR-VQA	54.42	46.04	30.36	57.09	42.19	9.38	50.45	54.03

CODA-Prompt	ScienceQA	TextVQA	ImageNet	GQA	VizWiz	Grounding	VQAv2	OCR-VQA
ScienceQA	70.26							
TextVQA	58.72	57.05						
ImageNet	36.96	34.95	30.26					
GQA	50.78	53.52	10.12	59.35				
VizWiz	55.37	47.21	6.78	56.43	48.01			
Grounding	33.56	47.67	32.07	44.62	43.39	34.42		
VQAv2	48.34	49.54	20.72	47.72	35.73	13.03	60.87	
OCR-VQA	58.15	50.16	24.04	54.33	48.94	17.83	55.86	54.42

ModalPrompt	ScienceQA	TextVQA	ImageNet	GQA	VizWiz	Grounding	VQAv2	OCR-VQA
ScienceQA	77.05							
TextVQA	70.50	58.50						
ImageNet	68.57	58.18	42.26					
GQA	68.82	56.08	43.43	62.17				
VizWiz	67.48	55.05	37.60	61.81	48.81			
Grounding	66.58	55.68	35.92	61.95	48.74	36.88		
VQAv2	68.12	56.43	40.22	60.92	51.19	36.63	64.99	
OCR-VQA	68.42	56.40	41.13	61.11	50.13	36.69	66.90	59.68

MoELoRA	ScienceQA	TextVQA	ImageNet	GQA	VizWiz	Grounding	VQAv2	OCR-VQA
ScienceQA	75.78							
TextVQA	34.47	51.80						
ImageNet	22.61	0.04	79.60					
GQA	32.37	34.04	42.48	57.95				
VizWiz	45.32	38.13	2.63	43.80	58.70			
Grounding	58.76	9.08	5.64	31.87	11.45	36.77		
VQAv2	33.01	48.42	10.61	49.78	32.23	1.75	64.58	
OCR-VQA	47.34	32.91	38.73	37.15	42.48	0.97	42.77	57.50

Dualprompt	ScienceQA	TextVQA	ImageNet	GQA	VizWiz	Grounding	VQAv2	OCR-VQA
ScienceQA	67.16							
TextVQA	52.20	53.12						
ImageNet	28.49	24.77	46.40					
GQA	49.70	47.94	12.06	55.10				
VizWiz	57.88	51.17	21.34	48.03	51.62			
Grounding	18.27	39.64	29.77	44.06	35.97	30.82		
VQAv2	36.77	49.85	22.48	27.96	41.08	13.51	61.27	
OCR-VQA	56.40	47.12	34.96	42.03	44.14	12.01	54.43	53.36

ProgLoRA	ScienceQA	TextVQA	ImageNet	GQA	VizWiz	Grounding	VQAv2	OCR-VQA
ScienceQA	76.27							
TextVQA	59.06	60.78						
ImageNet	70.92	52.57	97.32					
GQA	51.56	50.33	79.68	61.27				
VizWiz	65.62	50.79	81.17	48.99	60.16			
Grounding	40.08	47.53	77.75	50.61	48.30	39.35		
VQAv2	76.90	56.79	73.90	53.64	40.63	35.96	65.83	
OCR-VQA	74.84	51.83	83.90	49.93	53.87	31.19	62.71	64.44

DRAPE	ScienceQA	TextVQA	ImageNet	GQA	VizWiz	Grounding	VQAv2	OCR-VQA
ScienceQA	73.66							
TextVQA	70.29	59.93						
ImageNet	69.21	59.43	94.46					
GQA	69.87	59.71	94.50	61.96				
VizWiz	69.79	59.54	94.53	61.88	54.60			
Grounding	70.01	59.51	94.46	61.93	54.55	68.03		
VQAv2	70.57	59.49	94.32	61.19	54.32	68.07	66.65	
OCR-VQA	70.67	59.61	94.16	61.37	54.43	67.92	66.53	65.11

Table 11: Detailed continual instruction tuning results matrices for different methods on the UCIT benchmark.

LoRA-FT	ImgNet-R	ArxivQA	VizWiz	IconQA	CLEVR	Flickr30k
ImgNet-R	92.00					
ArxivQA	90.63	91.03				
VizWiz	74.00	91.40	61.90			
IconQA	72.73	78.53	53.03	79.87		
CLEVR	68.97	77.60	49.43	67.67	78.33	
Flickr30k	58.03	77.63	44.39	67.40	61.77	58.22

MoELoRA	ImgNet-R	ArxivQA	VizWiz	IconQA	CLEVR	Flickr30k
ImgNet-R	91.23					
ArxivQA	90.07	92.80				
VizWiz	80.30	92.03	60.22			
IconQA	80.53	77.20	51.19	53.73		
CLEVR	79.37	77.40	48.68	51.30	71.67	
Flickr30k	70.07	77.70	44.69	50.03	54.03	57.34

CL-MoE	ImgNet-R	ArxivQA	VizWiz	IconQA	CLEVR	Flickr30k
ImgNet-R	91.30					
ArxivQA	87.40	92.27				
VizWiz	76.23	90.53	60.18			
IconQA	77.33	76.20	51.09	54.33		
CLEVR	75.73	75.70	48.27	52.80	73.23	
Flickr30k	66.33	77.00	44.78	51.87	53.53	57.42

SEFE	ImgNet-R	ArxivQA	VizWiz	IconQA	CLEVR	Flickr30k
ImgNet-R	90.80					
ArxivQA	90.23	91.23				
VizWiz	83.60	89.73	61.24			
IconQA	84.17	77.20	56.36	78.93		
CLEVR	82.90	77.13	53.54	71.43	75.73	
Flickr30k	80.83	78.00	47.01	69.63	65.83	57.92

O-LoRA	ImgNet-R	ArxivQA	VizWiz	IconQA	CLEVR	Flickr30k
ImgNet-R	91.40					
ArxivQA	90.07	94.73				
VizWiz	81.50	91.70	61.59			
IconQA	82.40	77.73	54.06	76.63		
CLEVR	81.07	77.87	50.62	62.00	73.53	
Flickr30k	77.50	78.07	44.50	63.13	64.73	58.16

ModalPrompt	ImgNet-R	ArxivQA	VizWiz	IconQA	CLEVR	Flickr30k
ImgNet-R	51.10					
ArxivQA	51.03	87.37				
VizWiz	51.10	87.37	48.14			
IconQA	51.20	87.27	48.03	39.30		
CLEVR	51.17	87.23	47.96	39.40	47.07	
Flickr30k	51.07	87.27	48.11	39.23	46.57	42.93

HiDe	ImgNet-R	ArxivQA	VizWiz	IconQA	CLEVR	Flickr30k
ImgNet-R	90.87					
ArxivQA	88.83	92.27				
VizWiz	88.17	91.90	54.46			
IconQA	87.00	89.70	46.51	73.13		
CLEVR	86.23	90.07	44.28	64.37	54.77	
Flickr30k	84.03	90.73	44.43	58.93	41.37	54.25

DRAPE	ImgNet-R	ArxivQA	VizWiz	IconQA	CLEVR	Flickr30k
ImgNet-R	85.13					
ArxivQA	85.27	91.90				
VizWiz	85.20	91.87	55.70			
IconQA	85.27	91.63	55.77	62.63		
CLEVR	85.23	91.57	55.62	62.27	66.10	
Flickr30k	85.07	91.60	55.68	62.53	65.77	55.82

projection to reduce interference with earlier tasks. After each task, the cumulative feature statistics and projection matrices are updated, and a compact CLIP-space prototype is registered for task-free routing. During inference, the prototype router selects the most relevant frozen generator based on the current image-instruction pair, enabling instance-specific prompt generation without requiring task labels.

K Detailed Result Matrices

In Tables 10–11, we report the final accuracy matrices of the compared methods on the CoIN and UCIT benchmarks. Results may vary across hardware and software environments.

L Broader Impacts

This work studies rehearsal-free and parameter-efficient multimodal continual instruction tuning, which may reduce the computational and storage costs of adapting MLLMs to evolving tasks while helping preserve previously learned capabilities. Such improvements can make continual updates more practical for research and resource-constrained applications. The broader risks are mainly those associated with MLLMs and continual adaptation in general: if newly introduced data contains biases, unsafe content, or systematic errors, the adapted model may inherit these issues, and more efficient adaptation could also facilitate unintended specialization. This work does not introduce new datasets, release high-risk models, or directly address deployment safety; practical use should therefore include careful data curation, safety evaluation, and post-update monitoring.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction state the main contributions and scope, and the claims are supported by the method and experimental results.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We included it in the paper.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

Justification: The assumptions and derivation for the first-order analysis of null-space gradient projection are provided in the appendix.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: The paper provides all the necessary details to reproduce the results.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: The source code is publicly available at: `https://github.com/LAMDA-CL/NeurIPS2026-Drape`. Detailed algorithmic descriptions are provided in the appendix.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: The experiment details are available in the paper.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: The paper does not report error bars, but follows standard MCIT evaluation protocols on fixed benchmarks and shows consistent gains across multiple baselines. Repeated full MCIT runs are computationally expensive.

   Guidelines:

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

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: The paper reports the computational resources used for the experiments.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conforms to the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The paper discusses potential positive impacts and general risks of multimodal continual adaptation in appendix.

    Guidelines:

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

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: No high-risk data or models are released.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: All the datasets used are publicly available, discussed, and properly cited in Section 5.

    Guidelines:

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

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

13. **New assets**

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

    Answer: [N/A]

    Justification: The paper does not release new datasets, models, or benchmarks.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or new human-subject studies.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve human-subject research.

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs

Guidelines:

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