BRACE: BENEFIT-AWARE REUSE AND ADAPTIVE CAPACITY EXPANSION FOR MULTIMODAL CONTINUAL INSTRUCTION TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multimodal continual instruction tuning requires models to learn new tasks while retaining existing knowledge. Existing methods primarily rely on parameter-level protection or module-level management, facing two limitations: 1) *Insufficient historical knowledge protection and reuse*, as preserving parameters or modules alone struggles to ensure both stable knowledge retention and accurate selection of beneficial components, potentially causing forgetting, negative transfer, and redundant learning; 2) *Inefficient capacity allocation and utilization*, as residual demand after reuse and component complementarity are insufficiently considered, leading to imbalanced capacity allocation and directional redundancy. To address these limitations, we propose BRACE, which organizes rank-one LoRA components into independently manageable Atoms. The FixedAtom component freezes historical Atoms and task catalogs and enables selective reuse through benefit-aware selection. The AdaAtom component adjusts the number and layer-wise distribution of new Atoms according to residual demand after reuse under a global capacity constraint. An input diversity loss further reduces directional redundancy among new Atoms within each layer. We evaluate BRACE on UCIT and MLLM-DCL using two multimodal backbones. On LLaVA-1.5, BRACE improves mean final performance over the best prior methods by 6.22 and 8.53 percentage points on the two benchmarks, respectively. Compared with BRACE-NA, a variant without AdaAtom or input diversity loss, BRACE reduces cumulative adaptation parameters by 38.3% and 29.2%, respectively, with comparable performance.

1 INTRODUCTION

Multimodal Large Language Models (MLLMs) have demonstrated strong capabilities in visual recognition, visual question answering, and multimodal reasoning through visual instruction tuning (Liu et al., 2023; Dai et al., 2023; Zhu et al., 2024; Chen et al., 2023; Yin et al., 2024). In real-world deployment, models must continually learn new tasks in applications such as remote sensing, medical diagnosis, and autonomous driving while retaining existing knowledge (Lobry et al., 2020; He et al., 2020; Sima et al., 2025; Zhao et al., 2025). Multimodal continual instruction tuning (MCIT) aims to meet this need (He et al., 2026; Chen et al., 2024a; Cao et al., 2024; Liu et al., 2025b). Low-Rank Adaptation (LoRA) enables parameter-efficient adaptation by freezing the base model and learning low-rank updates (Hu et al., 2022). However, continually updating a shared adapter can overwrite historical adaptation directions (Wang et al., 2023; Liang & Li, 2024), whereas maintaining task-specific adapters reduces interference at the cost of storage that grows with the number of tasks (Zhang et al., 2025a; Meng et al., 2026). Balancing historical knowledge retention, cross-task reuse, and controlled parameter growth therefore remains a central challenge in MCIT (Wang et al., 2024; Guo et al., 2025c).

Existing methods mitigate forgetting and control adaptation costs through parameter constraints, modular knowledge organization, and low-rank compression, but two key limitations remain. First, *insufficient historical knowledge protection and reuse*. Many approaches protect historical knowledge by constraining parameter updates or preserving entire modules (Chen et al., 2025; Li et al., 2026; Wang et al., 2023). Different adaptation directions within a LoRA module may have different transfer values for a new task (Zhang et al., 2023; Valipour et al., 2023; Wu et al., 2024).

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

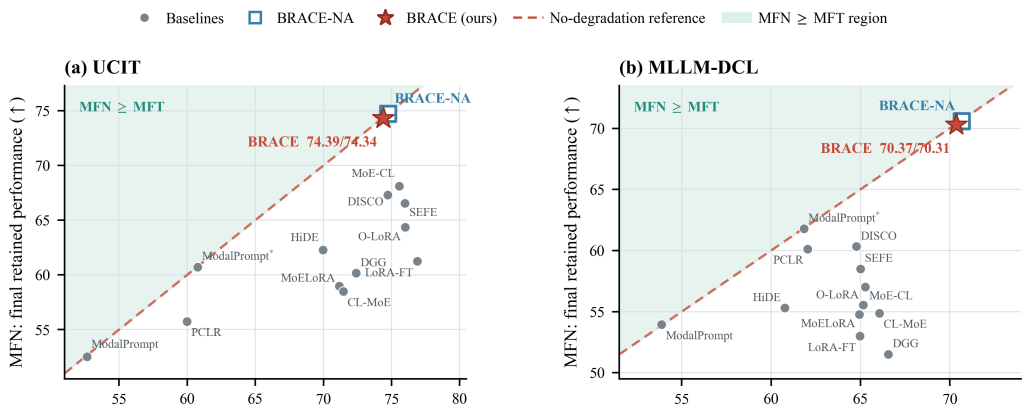


Figure 1: MFT–MFN trade-off on (a) UCIT and (b) MLLM-DCL under the MCITlib protocol (Guo et al., 2025d). Dashed line: MFN = MFT; shaded region: MFN ≥ MFT. Higher is better.

Reusing an entire module can introduce unsuitable directions, whereas excluding it can discard useful components, increasing the risks of negative transfer and redundant learning, respectively. Recent work has further explored fine-grained knowledge organization and reuse through pools of rank-one components and associative memory experts (Meng et al., 2026; Lu et al., 2025). However, finer granularity alone does not ensure that historical components and their reuse configurations remain unchanged, nor does it explicitly assess their contributions to the current task loss. Historical knowledge protection and benefit-aware reuse therefore remain insufficiently coordinated. Second, *inefficient capacity allocation and utilization*. Existing methods primarily control additional parameter counts through expert organization and parameter compression (Chen et al., 2024a; Guo et al., 2025a; Meng et al., 2026), without sufficiently relating new-capacity allocation to historical reuse benefits and layer-wise residual adaptation needs, potentially causing imbalanced capacity allocation. Moreover, limiting capacity alone does not prevent new components from learning similar directions, reducing the effective utilization of limited adaptation resources.

To address these issues, we propose BRACE, an atomized continual adaptation framework for MCIT. Building on the rank-one representation of LoRA updates, BRACE uses independently manageable components, termed *Atoms*, as a unit for historical knowledge protection and reuse, as well as new-capacity allocation and utilization. For historical knowledge protection and reuse, FixedAtom keeps historical Atoms and task catalogs immutable, preserving registered adaptation configurations and preventing subsequent training from directly overwriting existing components. Benefit-aware selection determines which historical Atoms to reuse and how many, integrating historical knowledge protection with selective reuse. For new-capacity allocation and utilization, AdaAtom uses validation gains from reuse and layer-wise historical support to estimate residual adaptation needs, adjusting the number and layer-wise distribution of new Atoms under a global capacity constraint. Meanwhile, an Input Diversity loss suppresses input-direction redundancy among new Atoms within each layer, encouraging complementary features. By combining historical knowledge reuse with adaptive Atom allocation, BRACE builds a reusable Atom pool for subsequent tasks.

We evaluate BRACE on UCIT (Guo et al., 2025a) and MLLM-DCL (Zhao et al., 2025) using LLaVA-1.5 (Liu et al., 2024a) and InternVL (Chen et al., 2024b). As shown in Figures 1(a) and 1(b), on LLaVA-1.5, BRACE improves mean final performance (MFN) over the best prior methods by 6.22 and 8.53 percentage points on the two benchmarks, respectively, with MFN close to mean first-time performance (MFT). Compared with BRACE-NA, a variant without AdaAtom or ID loss, BRACE reduces cumulative adaptation parameters by 38.3% and 29.2%, respectively, with comparable performance. Ablations and controlled comparisons show that BRACE effectively reuses historical knowledge, controls new capacity, and reduces directional redundancy among Atoms.

Our main contributions are as follows: *(i)* We propose BRACE, an atomized continual adaptation framework that treats rank-one LoRA components as manageable units and employs FixedAtom for historical knowledge protection and benefit-aware reuse. *(ii)* We introduce AdaAtom, which adjusts

the number and layer-wise distribution of new Atoms under a global capacity constraint according to residual adaptation demand after reuse, reducing unnecessary capacity growth. *(iii)* We introduce an input diversity loss that suppresses directional redundancy among new Atoms within each layer, improving the complementarity and utilization of limited adaptation capacity.

2 Related Work

Historical Knowledge Protection and Reuse. Parameter constraints and knowledge distillation mitigate forgetting by restricting updates or preserving historical behavior. EWC and InfLoRA constrain parameter updates, while LwF preserves historical capabilities through knowledge distillation (Kirkpatrick et al., 2017; Liang & Li, 2024; Li & Hoiem, 2016). SEFE protects important parameter locations, and DGG constrains current updates using historical gradients (Chen et al., 2025; Li et al., 2026). However, these strategies primarily limit changes to historical knowledge rather than fix all historical adaptation parameters, leaving learned directions susceptible to subsequent updates. Modular approaches reduce interference by preserving and organizing adaptation components. O-LoRA freezes historical low-rank updates and enforces orthogonality between new and historical subspaces (Wang et al., 2023). BranchLoRA, MoELoRA, CL-MoE, ModalPrompt, HiDe-LLaVA, and DISCO organize knowledge through task-specific branches, experts, or modality-specific prompts (Zhang et al., 2025a; Chen et al., 2024a; Huai et al., 2025; Zeng et al., 2025; Guo et al., 2025a;b), while MoE-CL combines shared and task-specific LoRA experts (Kang et al., 2026). However, module-level retrieval cannot distinguish the transfer value of individual directions within a module: reusing an entire module can introduce unsuitable directions, whereas excluding it can discard useful components, causing interference and redundant learning. Recent work explores fine-grained knowledge management. PCLR uses a pool of rank-one components for knowledge compression and integration (Meng et al., 2026), while MoRAM enables component-level reuse through sparse self-activation of rank-one associative memory experts (Lu et al., 2025). Although these methods support fine-grained organization and retrieval, they do not explicitly validate candidate components' transfer benefits through current-task loss changes to guide reuse. BRACE adopts rank-one Atoms as adaptation units and uses FixedAtom to keep historical parameters and task catalogs immutable, preserving registered adaptation configurations. It combines candidate-level benefit estimation with configuration-level validation to determine which historical components to reuse and how many, coordinating historical knowledge protection with effective reuse.

Adaptation Capacity Allocation and Utilization. Existing methods manage continual adaptation costs through budget control, module organization, and parameter compression (Wang et al., 2022; Guo et al., 2025c). MoELoRA and CL-MoE allocate adaptation capacity through expert sharing and routing (Chen et al., 2024a; Huai et al., 2025). HiDe-LLaVA organizes shared and task-specific components according to layer-wise adaptation characteristics (Guo et al., 2025a), while PCLR controls storage growth by compressing and consolidating a low-rank parameter pool (Meng et al., 2026). However, controlling capacity does not ensure its effective utilization. On the one hand, fixed or predefined allocations insufficiently account for layer-wise differences in historical knowledge support and residual adaptation demand, potentially oversupplying well-supported layers while undersupplying those with limited support. On the other hand, constraining parameter counts or module sizes alone does not ensure component complementarity: multiple low-rank components may still learn similar directions, reducing the effective utilization of limited adaptation resources. BRACE jointly improves capacity allocation and utilization through AdaAtom and the Input Diversity loss. For allocation, AdaAtom adjusts the number and layer-wise distribution of new Atoms under a global capacity constraint according to each layer's residual adaptation demand after historical knowledge reuse, aligning capacity with adaptation needs. For utilization, the Input Diversity loss suppresses input-direction redundancy among new Atoms within each layer, encouraging complementary features and improving the efficiency of the limited additional capacity.

3 Methodology

3.1 Overview

We propose BRACE, an atomized continual adaptation framework for multimodal continual instruction tuning that combines benefit-aware historical reuse with residual-guided capacity allocation. As

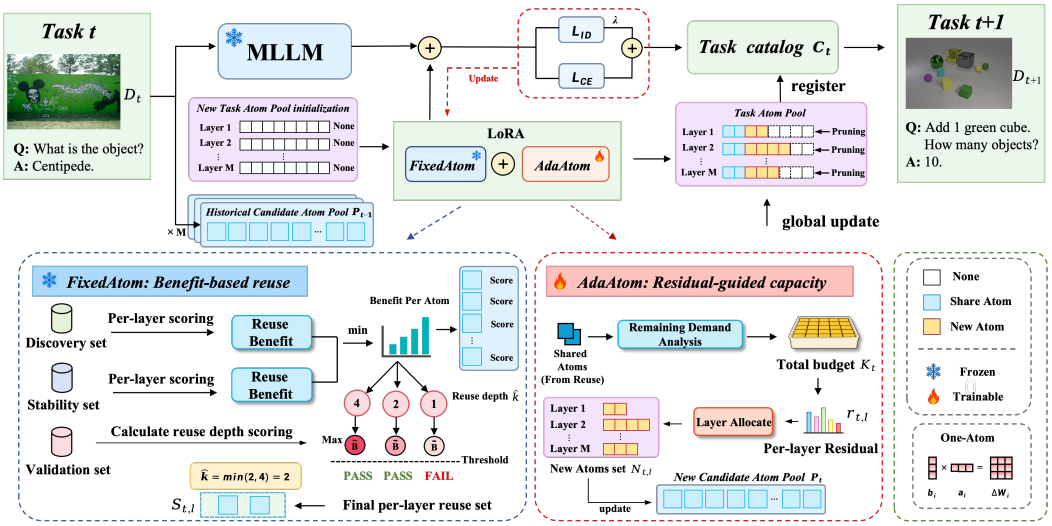


Figure 2: **Overview of BRACE.** FixedAtom selects frozen historical Atoms using benefit estimates and held-out validation. AdaAtom allocates new capacity from reuse evidence. Only new Atoms are trained with the task and ID losses, then frozen and registered for future reuse.

shown in Figure 2, given task-t data $\mathcal{D}_t$, BRACE constructs layer-wise LoRA branches by reusing Atoms from the historical pool $\mathcal{P}_{t-1}$ and allocating new ones. Each Atom is a rank-one LoRA component. FixedAtom evaluates historical Atoms on image-disjoint discovery and stability sets, ranks them by their standardized Atom-level removal benefits, and uses an independent validation set to determine the reuse depth $\hat{k}$ and layer-wise reuse sets $\mathcal{S}_{t,\ell}$. Historical Atoms and registered task catalogs remain frozen throughout this process. AdaAtom then uses the normalized validation reuse gain e_t, i.e., the clipped relative validation loss reduction to characterize residual adaptation demand, determines the total active budget K_t, and allocates new Atoms $\mathcal{N}_{t,\ell}$ according to layer-wise residual scores $r_{t,\ell}$. During training, only the new Atoms are optimized using the task and Input Diversity losses, while the backbone and historical Atoms remain frozen. Afterward, the new Atoms are frozen and added to the historical pool, and the resulting adaptation path is registered as task catalog $\mathcal{C}_t$ for subsequent reuse.

3.2 FixedAtom: Benefit-Aware Reuse

Given the task-t dataset $\mathcal{D}_t$ and the frozen historical Atom pool $\mathcal{P}_{<t}^{(\ell)}$ at each layer, FixedAtom selects reusable historical Atoms while preserving existing parameters and task catalogs. An Atom is a rank-one LoRA component $b_u^{(\ell)} a_u^{(\ell)\top} \in \mathbb{R}^{d_\ell \times k_\ell}$, and the layer-$\ell$ update is:

$$\Delta W_t^{(\ell)} = s \sum_{u=1}^{k_{t,\ell}} b_u^{(\ell)} \left(a_u^{(\ell)}\right)^\top. \tag{1}$$

where s is the scaling factor, $k_{t,\ell}$ is the number of active Atoms at layer ℓ for task t, and $\Delta W_t^{(\ell)}$ is the low-rank adaptation to the frozen pretrained weight. This representation enables each historical Atom to be independently scored, selected, and referenced.

Historical Atom benefit estimation. To estimate the utility of historical Atoms for the current task, FixedAtom partitions $\mathcal{D}_t$ into image-disjoint discovery and stability sets, $\mathcal{D}_t^{\text{dis}}$ and $\mathcal{D}_t^{\text{sta}}$. For a candidate Atom u, its Atom-level removal benefit is defined by the loss change caused by its removal:

$$\beta_u(x) = \mathcal{L}(x; g_u = 0) - \mathcal{L}(x; g_u = 1) \approx - \left. \frac{\partial \mathcal{L}(x; g_u)}{\partial g_u} \right|_{g_u = 1}. \tag{2}$$

where x is a current-task sample, and $g_u = 1$ and $g_u = 0$ denote retaining and removing Atom u, respectively. The approximation follows from a first-order Taylor expansion at $g_u = 1$. One forward

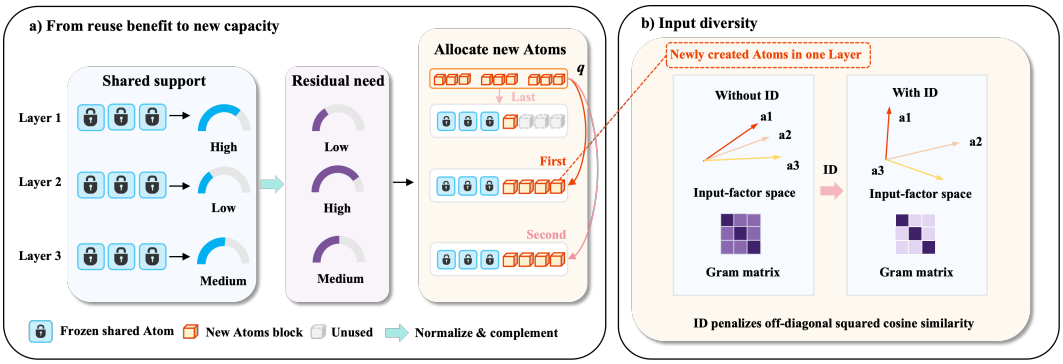


Figure 3: **Illustration of AdaAtom and the Input Diversity (ID) loss. (a)** AdaAtom allocates new Atoms according to task-level residual demand and layer-wise historical-support score. **(b)** ID reduces directional redundancy among new Atoms within each layer.

and backward pass produces the gate gradients, and hence the first-order removal benefit estimate, for all the candidate Atoms. The derivation and the relationship between the benefit and loss are provided in Appendix E and Appendix A.3.

Stable benefit ranking. To reduce finite-sample ranking variation, FixedAtom ranks historical Atoms within each layer using smaller standardized benefit score from discovery and stability sets:

$$\mathcal{S}_{t,\ell} = \mathrm{TopK}\Big(\mathcal{P}^{(\ell)}_{<t}, \min\big(z^{\mathrm{dis}}_u, z^{\mathrm{sta}}_u\big), k\Big), \tag{3}$$

where z^{dis}_u and z^{sta}_u are the two standardized benefits, and TopK selects the k highest-ranked Atoms. Their computation is detailed in Appendix A.3.

Reuse depth selection. To determine how much historical knowledge can be reused for the current task, FixedAtom ranks historical Atoms by their current-task Benefits and validates candidate reuse configurations on an independent held-out set, we select the reuse depth on an independent validation set $\mathcal{D}^{\mathrm{val}}_t$. For each $k \in \mathcal{K}$, all layers activate their top-k Atoms, and the resulting configuration is evaluated against zero reuse. Using its mean configuration Benefit $\bar{B}_t(k)$, we define:

$$k^\star = \arg\max_{k\in\mathcal{K}} \bar{B}_t(k), \qquad \tau_t = \bar{B}_t(k^\star) - \mathrm{SE}_t(k^\star), \tag{4}$$

where $\mathcal{K}$ is the candidate depth set, $k^\star$ is the depth with the highest mean configuration benefit, and $\mathrm{SE}_t(k^\star)$ is its standard error. We then select:

$$\hat{k} = \min\left\{k \in \mathcal{K} : \bar{B}_t(k) > 0,\ \bar{B}_t(k) \geq \tau_t\right\}. \tag{5}$$

where $\hat{k}$ is the smallest depth with a positive mean configuration benefit that reaches τ_t. If no candidate qualifies, we set $\hat{k} = k^\star$. The validation benefit of the selected configuration is passed to AdaAtom for capacity allocation. The mean configuration benefit and standard error computation are detailed in Appendix A.3.

3.3 AdaAtom: Adaptive Residual Allocation

Given the historical reuse sets $\mathcal{S}_{t,\ell}$ selected by FixedAtom, AdaAtom determines the remaining capacity before training. As illustrated in Figure 3(a), the validation reuse gain e_t determines the task-level active budget K_t, while the layer-wise residual scores $r_{t,\ell}$ distribute the remaining capacity according to residual adaptation demand.

Task-level active budget. Using the relative validation reuse gain $e_t \in [0, 1]$, AdaAtom computes:

$$K_t = \mathrm{clip}_{[F_t, K_{\mathrm{cap}}]}\left[F_t + q\,\mathrm{round}\left(\frac{(1-e_t)(K_{\mathrm{cap}} - F_t)}{q}\right)\right]. \tag{6}$$

where K_t is the total active budget containing both reused and new Atoms, F_t is the sum of the layer-wise capacity floors, K_{cap} is the global active-capacity cap, and q is the allocation quantum. A larger

e_t contracts the budget toward F_t, whereas a smaller e_t permits greater expansion. The computation of the relative validation reuse gain e_t and capacity configuration are detailed in Appendix A.4.

Layer-wise residual allocation. Given K_t, AdaAtom prioritizes layers with weaker historical-support score. The support and residual scores of layer ℓ are defined as:

$$b_{t,\ell} = \sum_{u \in \mathcal{P}_{<t}^{(\ell)}} \max\left(\overline{\widetilde{\beta}}_u^{\mathrm{dis}}, 0\right), \qquad r_{t,\ell} = 1 - \frac{b_{t,\ell}}{\max_j b_{t,j}}. \tag{7}$$

where $\overline{\widetilde{\beta}}_u^{\mathrm{dis}}$ is the mean discovery-set benefit of candidate Atom u, $b_{t,\ell}$ measures the historical-support score of layer ℓ, and a larger $r_{t,\ell}$ indicates greater residual demand. If all support scores are zero, all residual scores are set to one. The relationships among Atom-level benefits, layer-wise historical support, and the relative validation reuse gain are detailed in Appendix A.4.

Starting from the layer-wise capacity floors, AdaAtom allocates the remaining capacity according to the relative residual scores. At each step, it compares all unsaturated layers and assigns one quantum to the layer with the largest residual score; a layer may receive multiple quanta until reaching its cap, after which allocation proceeds to the next highest-scoring layer :

$$\ell^\dagger = \arg\max_{\ell: k_{t,\ell} < R} r_{t,\ell}, \qquad k_{t,\ell^\dagger} \leftarrow k_{t,\ell^\dagger} + q. \tag{8}$$

where R is the per-layer active-capacity cap and $k_{t,\ell}$ is the total active count at layer ℓ, including frozen reused Atoms and trainable new Atoms. The settings of R and q are provided in Appendix A.4. Allocation stops when the total active count reaches K_t, yielding:

$$m_{t,\ell} = k_{t,\ell} - |\mathcal{S}_{t,\ell}|. \tag{9}$$

where $m_{t,\ell} = |\mathcal{N}_{t,\ell}|$ is the number of trainable new Atoms at layer ℓ. These new Atoms and the frozen reuse set $\mathcal{S}_{t,\ell}$ jointly form the current adaptation path.

Task-catalog construction and protection. The reuse set $\mathcal{S}_{t,\ell}$ and the new Atom set $\mathcal{N}_{t,\ell}$ jointly define the adaptation path at each module. During training, the backbone and historical Atoms remain frozen, and only new Atoms are optimized. After training, the new Atoms are frozen and added to the historical pool, while the task catalog is registered as

$$\mathcal{C}_t = \{\mathcal{S}_{t,\ell} \cup \mathcal{N}_{t,\ell}\}_{\ell=1}^{L}, \tag{10}$$

where L is the number of adapted modules and $\mathcal{C}_t$ records their active Atom references. The first task uses the full per-module budget without historical reuse. Subsequent tasks append new Atoms and catalogs without modifying historical parameters or references, protecting existing adaptation paths while avoiding copies of shared Atoms. The task-ID-free inference procedure constructs task prototypes and selects the corresponding catalog from the input, as described in Appendix A.7.

3.4 Input Diversity

Although AdaAtom adapts the amount and placement of new Atoms, it does not prevent them from learning similar directions within a layer. As illustrated in Figure 3(b), the Input Diversity (ID) loss penalizes their off-diagonal squared cosine similarities, encouraging complementary directions and more effective capacity utilization:

$$\mathcal{L}_{\mathrm{ID}} = \frac{1}{|\mathcal{M}_t|} \sum_{\ell \in \mathcal{M}_t} \frac{1}{m_{t,\ell}(m_{t,\ell}-1)} \sum_{i \neq j} \cos^2\left(a_i^{(\ell)}, a_j^{(\ell)}\right). \tag{11}$$

where $\mathcal{M}_t$ contains layers with at least two new Atoms, $m_{t,\ell} = |\mathcal{N}_{t,\ell}|$, and i and j index distinct new Atoms within the same layer. We set $\mathcal{L}_{\mathrm{ID}} = 0$ when $\mathcal{M}_t$ is empty. Squared cosine similarity penalizes parallel and antiparallel directions equally, while pair-wise normalization and layer-wise averaging balance contributions across layers. Historical Atoms are excluded from this loss. Implementation details are provided in Appendix A.5.

Joint optimization and output. Given resulting adaptation path, we optimize the new Atoms using:

$$\mathcal{L} = \mathcal{L}_{\mathrm{CE}} + \lambda_{\mathrm{ID}} \mathcal{L}_{\mathrm{ID}}. \tag{12}$$

where $\mathcal{L}_{\mathrm{CE}}$ is the cross-entropy loss over current-task answer tokens, as detailed in Appendix A.5, and λ_{ID} controls the strength of the ID regularization. During training, the backbone and reused historical Atoms remain frozen, and only the new Atoms are updated. After training, the new Atoms are frozen, added to historical pool, and registered in the current task catalog for subsequent reuse.

Table 1: Comparison of different methods on the MLLM-DCL benchmark based on LLaVA-1.5. The best and second-best results are highlighted in **bold** and underline, respectively.

Method	Venue	RS	Med	AD	Sci	Fin	MFT (↑)	MFN (↑)	MAA (↑)	BWT (↑)
Zero-shot	-	32.29	28.28	15.59	35.55	62.56	34.85	-	-	-
Individual	-	78.15	58.20	52.77	49.32	88.02	65.29	-	-	-
LoRA-FT	ICLR-22	69.65	41.59	25.43	40.88	87.45	64.98	53.00	61.13	-14.97
O-LoRA	EMNLP-23	74.64	44.42	30.02	41.47	87.15	65.16	55.54	62.12	-12.03
MoELoRA	NeurIPS-24	77.54	41.85	27.62	40.13	86.75	64.94	54.78	61.76	-12.71
ModalPrompt	EMNLP-25	53.63	45.68	40.77	41.81	87.82	53.87	53.94	49.67	**0.09**
ModalPrompt*	EMNLP-25	78.67	51.37	47.80	43.29	87.78	61.84	61.78	64.05	-0.07
CL-MoE	CVPR-25	71.34	46.84	26.33	41.17	88.74	66.06	54.88	61.79	-13.97
HiDE	ACL-25	74.31	48.95	33.21	38.54	81.55	60.77	55.31	60.68	-6.82
SEFE	ICML-25	77.26	50.37	37.21	40.87	86.82	65.01	58.51	63.63	-8.13
DISCO	ICCV-25	76.49	44.48	44.84	46.61	89.22	64.78	60.33	63.93	-5.57
DGG	CVPR-26	57.76	42.24	30.27	37.53	89.79	66.57	51.52	57.92	-18.81
MoE-CL	WWW-26	70.69	43.11	39.36	42.24	89.73	65.27	57.03	63.39	-10.31
PCLR	ICLR-26	74.66	49.10	38.82	46.51	91.55	62.04	60.13	58.88	-2.39
BRACE-NA	-	**81.42**	**67.27**	55.13	**57.73**	91.41	**70.66**	**70.59**	**72.01**	-0.09
BRACE	-	81.25	65.94	**55.22**	57.51	**91.61**	70.37	70.31	71.52	-0.08

Table 2: Comparison of different methods on the UCIT benchmark based on LLaVA-1.5. The best and second-best results are highlighted in **bold** and underline, respectively.

Method	Venue	ImgNet-R	ArxivQA	VizWiz	IconQA	CLEVR	Flickr30k	MFT (↑)	MFN (↑)	MAA (↑)	BWT (↑)
Zero-shot	-	16.27	53.73	38.39	19.20	20.63	41.88	31.68	-	-	-
Individual	-	91.67	90.83	57.87	78.43	76.63	61.72	76.19	-	-	-
LoRA-FT	ICLR-22	58.03	77.63	44.39	67.40	61.77	**58.22**	**76.89**	61.24	76.55	-18.78
O-LoRA	EMNLP-23	77.50	78.07	44.50	63.13	64.73	58.16	76.01	64.35	78.02	-13.99
MoELoRA	NeurIPS-24	70.07	77.70	44.69	50.03	54.03	57.34	71.17	58.98	75.08	-14.63
ModalPrompt	EMNLP-25	51.07	87.27	48.11	39.23	46.57	42.93	52.65	52.53	57.67	-0.15
ModalPrompt*	EMNLP-25	74.43	92.00	55.92	44.27	53.97	43.67	60.76	60.71	70.57	-0.05
CL-MoE	CVPR-25	66.33	77.00	44.78	51.87	53.53	57.42	71.46	58.49	74.19	-15.56
HiDE	ACL-25	84.03	90.73	44.43	58.93	41.37	54.25	69.96	62.29	77.32	-9.20
SEFE	ICML-25	80.83	78.00	47.01	69.63	65.83	57.92	75.98	66.54	78.76	-11.33
DISCO	ICCV-25	87.70	94.93	46.71	69.50	49.13	55.93	74.71	67.31	81.00	-8.87
DGG	CVPR-26	86.33	76.57	43.20	57.37	42.33	55.17	72.40	60.16	76.66	-14.68
MoE-CL	WWW-26	84.67	94.17	46.04	71.20	56.37	56.27	75.56	68.12	80.75	-8.93
PCLR	ICLR-26	72.40	83.07	36.76	57.73	31.13	53.43	59.99	55.75	70.45	-5.09
BRACE-NA	-	89.57	**95.03**	**62.06**	**72.07**	**71.60**	57.96	74.74	**74.72**	**82.79**	**-0.02**
BRACE	-	**89.67**	**95.03**	61.78	70.30	71.37	57.89	74.39	74.34	82.62	-0.06

4 EXPERIMENTS

4.1 EXPERIMENTAL SETTING

We evaluate BRACE on UCIT (Guo et al., 2025a) and MLLM-DCL (Zhao et al., 2025) using LLaVA-1.5-7B (Liu et al., 2024a) and InternVL-Chat-7B (Chen et al., 2024b), with additional MLLM-ACL experiments in the appendix. BRACE uses no historical data replay and is evaluated without ground-truth task identities at inference. We denote the variant without AdaAtom or ID loss but retaining FixedAtom as BRACE-NA (Non-Adaptive). We report MFT, MFN, MAA, and BWT, alongside cumulative new Atom counts, adaptation parameter counts, and FP16 storage costs. We set the ID loss weight to 0.2 based on ablations. Dataset details, baselines, metric definitions, and full configurations are provided in the appendix.

4.2 MAIN EXPERIMENT

Results on MLLM-DCL. To assess continual adaptation across domains, we compare BRACE with existing methods on MLLM-DCL, as shown in Table 1. BRACE achieves MFT, MFN, and MAA scores of 70.37, 70.31, and 71.52, exceeding the best prior results by 3.80, 8.53, and 7.47 percentage points, respectively. It outperforms prior methods in final performance across all five domains, with a BWT of -0.08. Its MFN is only 0.28 percentage points below BRACE-NA (70.59), indicating that adaptive capacity allocation combined with ID loss maintains similar final performance. These results demonstrate effective new-domain learning with limited average forgetting.

Results on UCIT. Table 2 further demonstrates BRACE's effectiveness across heterogeneous tasks: its MFN of 74.34 and MAA of 82.62 exceed the best prior results by 6.22 and 1.62 percentage points, respectively. Despite lower MFT than LoRA-FT, BRACE achieves an MFN 13.10 percentage points

Table 3: Historical Atom selection and reuse depth on UCIT.

Task	k	Top-k	Random-k	Bottom-k
2	8	**0.099**	0.022	−0.093
3	4	**0.072**	−0.040	−0.176
4	8	**0.359**	0.229	0.016
5	2	**0.021**	−0.026	−0.155
6	8	**0.771**	0.647	0.305

(a) Atom selection

Task	k	Chosen	$k = 8$	All	Atoms (C/A)
2	8	**0.099**	**0.099**	−0.076	1,792 / 3,584
3	4	**0.072**	0.027	−0.115	896 / 2,544
4	8	**0.359**	**0.359**	0.289	1,792 / 3,805
5	2	0.021	**0.031**	−1.028	448 / 3,421
6	8	**0.771**	**0.771**	0.723	1,792 / 5,048

(b) Reuse depth

Table 4: Incremental ablation of AdaAtom and ID loss under the task-free protocol.

Dataset	Variant	MFT ↑	MFN ↑	MAA ↑	BWT ↑	Cum. Atoms ↓	Mean $\lvert\cos\rvert$ ↓
UCIT	w/o AdaAtom & ID	**74.74**	**74.72**	**82.79**	**−0.02**	15,680 (–)	0.0381
	w/o ID	73.84	73.69	82.25	−0.19	11,144 (−28.9%)	0.0539
	BRACE	74.39	74.34	82.62	−0.06	**9,576** (−38.9%)	**0.0229**
MLLM-DCL	w/o AdaAtom & ID	**70.66**	**70.59**	**72.01**	−0.09	28,672 (–)	0.0825
	w/o ID	70.31	70.25	71.65	**−0.06**	20,560 (−28.3%)	0.0959
	BRACE	70.37	70.31	71.52	−0.08	**20,256** (−29.4%)	**0.0592**

higher and improves BWT from -18.78 to -0.06. This demonstrates that BRACE can effectively retain the capabilities learned from previous tasks throughout the task sequence.

4.3 Analysis of Benefit-Aware Reuse

Benefit-aware selection. To determine whether benefit-aware ranking can identify useful historical Atoms rather than selecting them arbitrarily, we compare Top-k, Random-k, and Bottom-k on UCIT. We use the same checkpoint, candidate pool, and per-module reuse budget for all three strategies, and evaluate them on 96 samples whose images are disjoint from those used for ranking and depth selection. Top-k achieves the highest mean held-out configuration benefit on all five tasks. In particular, it yields positive mean configuration benefits of 0.072 and 0.021 on Tasks 3 and 5, whereas Random-k yields -0.040 and -0.026. These results indicate that benefit-aware ranking identifies more useful Atoms under a matched reuse budget.

Reuse depth selection. To analyze whether task-dependent depth selection can avoid unnecessary reuse while retaining transfer benefits, we compare the validation-selected depth (Chosen), fixed-depth reuse with $k = 8$, and reuse of all candidates (All) on the same independent evaluation set. The selected configuration achieves a higher benefit than All with fewer Atoms on every task. On Task 3, the selected $k = 4$ outperforms $k = 8$ (0.072 vs. 0.027). On Task 5, Chosen obtains 0.021 with only 448 Atoms, whereas All obtains -1.028 with $3{,}421$ Atoms; although $k = 8$ achieves a slightly higher mean configuration benefit of 0.031, it requires four times as many Atoms.

4.4 Ablation Studies

Component analysis. To disentangle the contributions of AdaAtom and ID loss, we introduce them incrementally starting from BRACE-NA. As shown in Table 4, adding AdaAtom alone reduces cumulative newly created Atoms by 28.9% and 28.3% on UCIT and MLLM-DCL, respectively, with MFN decreases of 1.03 and 0.34 percentage points. Subsequently adding ID loss further reduces Atom counts by 14.1% and 1.5%, improves MFN by 0.65 and 0.06 percentage points, and lowers input-direction similarity by 57.5% and 38.3%. These results indicate that AdaAtom accounts for most of the capacity reduction, while ID loss reduces directional redundancy and improves the trade-off between capacity and final performance under adaptive allocation. Relative to BRACE-NA, complete BRACE reduces cumulative newly created Atoms by 38.9% and 29.4%, with MFN decreases of only 0.38 and 0.28 percentage points.

ID loss weight. To examine the effect of regularization strength, we compare $\lambda \in \{0.1, 0.2, 0.3\}$. As shown in Table 5, increasing λ consistently reduces input-direction similarity on both bench-

Table 5: Sensitivity to the ID loss weight λ on UCIT and MLLM-DCL.

Dataset	λ	MFT ↑	MFN ↑	MAA ↑	BWT ↑	Cum. Atoms ↓	$\lvert\cos\rvert$ ↓
UCIT	0.1	73.72	73.58	82.21	−0.16	10696	0.0306
	0.2	**74.39**	**74.34**	**82.62**	−0.06	**9576**	0.0229
	0.3	73.77	73.93	82.27	**+0.18**	11148	**0.0183**
DCL	0.1	70.19	70.16	71.42	−0.04	**19920**	0.0730
	0.2	**70.37**	**70.31**	71.52	−0.08	20256	0.0592
	0.3	70.21	70.19	**71.53**	**−0.02**	20200	**0.0513**

Table 6: Final performance and adaptation storage under the task-free protocol.

Dataset	Method	MFN ↑	Cum. Atoms ↓	parameters(M) ↓	FP16 storage(MB) ↓
UCIT	O-LoRA	64.35	21,504	239.86	479.7
	MoE-CL	68.12	10,752	163.45	326.9
	PCLR	55.75	16,272	191.70	383.4
	BRACE-NA	**74.72**	15,680	174.90	349.8
	BRACE	74.34	**9,576**	**107.84**	**215.7**
MLLM-DCL	O-LoRA	55.54	35,840	399.77	799.5
	MoE-CL	57.03	**18,432**	279.31	558.6
	PCLR	60.13	25,312	297.58	595.2
	BRACE-NA	**70.59**	28,672	319.82	639.6
	BRACE	70.31	20,256	**226.38**	**452.8**

marks, but neither performance nor Atom counts change monotonically. Among the tested settings, $\lambda = 0.2$ achieves the highest MFT and MFN on both benchmarks and requires the fewest newly created Atoms on UCIT. We therefore adopt $\lambda = 0.2$. These results indicate that stronger directional diversity regularization does not necessarily improve task performance.

4.5 Storage Efficiency

To evaluate final performance and storage costs under different capacity strategies, we compare BRACE-NA, BRACE, and three prior methods on UCIT and MLLM-DCL. Table 6 reports MFN, cumulative Atom counts, adaptation parameters, and FP16 storage. BRACE-NA, which retains FixedAtom and provides relatively ample new capacity, achieves the highest MFN among all compared methods, reaching 74.72 and 70.59 on UCIT and MLLM-DCL, respectively. BRACE achieves MFN scores of 74.34 and 70.31, outperforming every listed non-BRACE baseline. On UCIT, it surpasses MoE-CL (68.12 MFN, 326.9 MB), while on MLLM-DCL it surpasses PCLR (60.13 MFN, 595.2 MB). BRACE also requires only 215.7 MB and 452.8 MB of FP16 storage on the two benchmarks, respectively. Although BRACE-NA achieves slightly higher MFN, it incurs substantially higher storage costs. These results show that BRACE achieves a favorable performance-storage trade-off by preserving performance close to BRACE-NA with lower storage overhead.

5 Conclusion

We presented BRACE, an atomized continual adaptation framework for multimodal continual instruction tuning. By organizing rank-one LoRA components into independently manageable Atoms, BRACE combines immutable historical knowledge protection and benefit-aware reuse through FixedAtom with adaptive capacity allocation through AdaAtom. An Input Diversity loss further reduces directional redundancy among newly created Atoms. Experiments demonstrate that BRACE improves final performance while maintaining strong new-task acquisition. Compared with its base variant, BRACE reduces cumulative adaptation parameters by 38.3% on UCIT and 29.2% on MLLM-DCL with only small decreases in final performance. These results highlight component-level knowledge management as an effective approach to balancing learning, retention, and parameter growth in continual multimodal adaptation.

AI USE STATEMENT

Generative AI tools were used to assist with the literature review, improve English writing, and polish grammar. All AI-assisted content was reviewed and verified by the authors, who made all final research decisions and take full responsibility for the final manuscript and its results.

REPRODUCIBILITY STATEMENT

We describe BRACE in the main text and provide additional implementation details, dataset descriptions, evaluation protocols, and hyperparameter settings in the appendix. We will release the code after publication to facilitate reproducibility.

REFERENCES

Manoj Acharya, Kushal Kafle, and Christopher Kanan. TallyQA: Answering complex counting questions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 8076–8084, 2019. doi: 10.1609/aaai.v33i01.33018076.

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. URL https://arxiv.org/abs/2411.02564.

Shuaichen Chang, David Palzer, Jialin Li, Eric Fosler-Lussier, and Ningchuan Xiao. MapQA: A dataset for question answering on choropleth maps. *arXiv preprint arXiv:2211.08545*, 2022.

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. In *Advances in Neural Information Processing Systems*, volume 37, pp. 57817–57840, 2024a. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/6a45500d9eda640deed90d8a62742be5-Abstract-Datasets_and_Benchmarks_Track.html.

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*, volume 267 of *Proceedings of Machine Learning Research*, pp. 7982–8001. PMLR, 2025. URL https://proceedings.mlr.press/v267/chen25n.html.

Jun Chen, Deyao Zhu, Xiaoqian Shen, Xiang Li, Zechun Liu, Pengchuan Zhang, Raghuraman Krishnamoorthi, Vikas Chandra, Yunyang Xiong, and Mohamed Elhoseiny. MiniGPT-v2: Large language model as a unified interface for vision-language multi-task learning. *arXiv preprint arXiv:2310.09478*, 2023. URL https://arxiv.org/abs/2310.09478.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, Bin Li, Ping Luo, Tong Lu, Yu Qiao, and Jifeng Dai. InternVL: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24185–24198, 2024b. URL https://arxiv.org/abs/2312.14238.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. InstructBLIP: Towards general-purpose vision-language models with instruction tuning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 49250–49267, 2023. URL https://arxiv.org/abs/2305.06500.

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)*, pp. 13572–13586. Association for Computational Linguistics, 2025a. doi: 10.18653/v1/2025.acl-long.666. URL https://aclanthology.org/2025.acl-long.666/.

Haiyang Guo, Fanhu Zeng, Fei Zhu, Wenzhuo Liu, Da-Han Wang, Jian Xu, Xu-Yao Zhang, and Cheng-Lin Liu. Federated continual instruction tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1325–1335, 2025b. URL https://arxiv.org/abs/2503.12897.

Haiyang Guo, Fanhu Zeng, Fei Zhu, Jiayi Wang, Xukai Wang, Jingang Zhou, Hongbo Zhao, Wenzhuo Liu, Shijie Ma, Da-Han Wang, Xu-Yao Zhang, and Cheng-Lin Liu. Continual learning for generative AI: From LLMs to MLLMs and beyond. *arXiv preprint arXiv:2506.13045*, 2025c. URL https://arxiv.org/abs/2506.13045.

Haiyang Guo, Fei Zhu, Hongbo Zhao, Fanhu Zeng, Wenzhuo Liu, Shijie Ma, Da-Han Wang, and Xu-Yao Zhang. MCITlib: Multimodal continual instruction tuning library and benchmark. *arXiv preprint arXiv:2508.07307*, 2025d. URL https://arxiv.org/abs/2508.07307.

Ziyu Guo, Renrui Zhang, Hao Chen, Jialin Gao, Dongzhi Jiang, Jiaze Wang, and Pheng-Ann Heng. SciVerse: Unveiling the knowledge comprehension and visual reasoning of LMMs on multimodal scientific problems. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 19683–19704. Association for Computational Linguistics, 2025e.

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*, pp. 3608–3617, 2018.

Jinghan He, Haiyun Guo, Kuan Zhu, Ming Tang, and Jinqiao Wang. Continual instruction tuning for large multimodal models. *IEEE Transactions on Image Processing*, 35:2699–2713, 2026. doi: 10.1109/TIP.2026.3671616.

Xuehai He, Yichen Zhang, Luntian Mou, Eric Xing, and Pengtao Xie. PathVQA: 30000+ questions for medical visual question answering. *arXiv preprint arXiv:2003.10286*, 2020. URL https://arxiv.org/abs/2003.10286.

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 *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8320–8329, 2021.

Yu-Chung Hsiao, Fedir Zubach, Gilles Baechler, Srinivas Sunkara, Victor Carbune, Jason Lin, Maria Wang, Yun Zhu, and Jindong Chen. ScreenQA: Large-scale question-answer pairs over mobile app screenshots. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 9427–9452, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.477. URL https://aclanthology.org/2025.naacl-long.477/.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

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 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19608–19617, 2025. URL https://arxiv.org/abs/2503.00413.

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*, pp. 2901–2910, 2017.

Jiazheng Kang, Le Huang, Cheng Hou, Zhe Zhao, ZhenXiang Yan, and Ting Bai. Self-evolving llms via continual instruction tuning. In *Proceedings of the ACM Web Conference 2026*, pp. 7724–7732, 2026.

Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images. In *European Conference on Computer Vision*, pp. 235–251. Springer, 2016.

Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4999–5007, 2017.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114. URL https://doi.org/10.1073/pnas.1611835114.

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)*, pp. 14369–14387. Association for Computational Linguistics, 2024a.

Songze Li, Mingyu Gao, Tonghua Su, Xu-Yao Zhang, and Zhongjie Wang. Multimodal continual instruction tuning with dynamic gradient guidance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. URL https://arxiv.org/abs/2511.15164.

Zhang Li, Biao Yang, Qiang Liu, Zhiyin Ma, Shuo Zhang, Jingxu Yang, Yabo Sun, Yuliang Liu, and Xiang Bai. Monkey: Image resolution and text label are important things for large multimodal models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024b.

Zhizhong Li and Derek Hoiem. Learning without forgetting. In *Computer Vision – ECCV 2016*, pp. 614–629. Springer, 2016. doi: 10.1007/978-3-319-46493-0_37. URL https://doi.org/10.1007/978-3-319-46493-0_37.

Yan-Shuo Liang and Wu-Jun Li. InfLoRA: Interference-free low-rank adaptation for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23638–23647, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Liang_InfLoRA_Interference-Free_Low-Rank_Adaptation_for_Continual_Learning_CVPR_2024_paper.html.

Adam Dahlgren Lindström and Savitha Sam Abraham. CLEVR-Math: A dataset for compositional language, visual, and mathematical reasoning. *arXiv preprint arXiv:2208.05358*, 2022.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 34892–34916, 2023. URL https://arxiv.org/abs/2304.08485.

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*, pp. 26296–26306, 2024a. URL https://openaccess.thecvf.com/content/CVPR2024/html/Liu_Improved_Baselines_with_Visual_Instruction_Tuning_CVPR_2024_paper.html.

Junpeng Liu, Tianyue Ou, Yifan Song, Yuxiao Qu, Wai Lam, Chenyan Xiong, Wenhu Chen, Graham Neubig, and Xiang Yue. Harnessing webpage uis for text-rich visual understanding. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 17971–18005, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/2da53cd1abdae59150e35f4693834f32-Paper-Conference.pdf.

Wenzhuo Liu, Fei Zhu, Haiyang Guo, Longhui Wei, and Cheng-Lin Liu. LLaVA-c: Continual improved visual instruction tuning. *arXiv preprint arXiv:2506.08666*, 2025b. URL https://arxiv.org/abs/2506.08666.

Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xu-Cheng Yin, Cheng-Lin Liu, Lianwen Jin, and Xiang Bai. OCRBench: On the hidden mystery of OCR in large multimodal models. *Science China Information Sciences*, 67(12), 2024b. doi: 10.1007/s11432-024-4235-6.

Sylvain Lobry, Diego Marcos, Jesse Murray, and Devis Tuia. RSVQA: Visual question answering for remote sensing data. *IEEE Transactions on Geoscience and Remote Sensing*, 58(12): 8555–8566, 2020. doi: 10.1109/TGRS.2020.2988782. URL https://doi.org/10.1109/TGRS.2020.2988782.

Haodong Lu, Chongyang Zhao, Minhui Xue, Lina Yao, Kristen Moore, and Dong Gong. Little by little: Continual learning via incremental mixture of rank-1 associative memory experts. *arXiv preprint arXiv:2506.21035*, 2025.

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 J. Vanschoren and S. Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/d3d9446802a44259755d38e6d163e820-Paper-round2.pdf.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. MathVista: Evaluating mathematical reasoning of foundation models in visual contexts. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.02255.

Weicheng Meng, Jingyang Qiao, Shaohui Liu, Zhizhong Zhang, and Yuan Xie. PCLR: Progressively compressed LoRA for multimodal continual instruction tuning. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=WdP1NVSzsz.

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*, pp. 2641–2649, 2015.

Wenhao Shi, Zhiqiang Hu, Yi Bin, Junhua Liu, Yang Yang, See-Kiong Ng, Lidong Bing, and Roy Ka-Wei Lee. Math-LLaVA: Bootstrapping mathematical reasoning for multimodal large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 4663–4680, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.268. URL https://aclanthology.org/2024.findings-emnlp.268/.

Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Jens Beißwenger, Ping Luo, Andreas Geiger, and Hongyang Li. DriveLM: Driving with graph visual question answering. In *Computer Vision – ECCV 2024*, pp. 256–274. Springer Nature Switzerland, 2025. doi: 10.1007/978-3-031-72943-0_15. URL https://doi.org/10.1007/978-3-031-72943-0_15.

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, Austin Wang, Rob Fergus, Yann LeCun, and Saining Xie. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 87310–87356. Curran Associates, Inc., 2024. doi: 10.52202/079017-2771. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/9ee3a664ccfeabc0da16ac6f1f1cfe59-Paper-Conference.pdf.

Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. DyLoRA: Parameter-efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 3274–3287. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.eacl-main.239. URL https://aclanthology.org/2023.eacl-main.239/.

Bryan Wang, Gang Li, Xin Zhou, Zhourong Chen, Tovi Grossman, and Yang Li. Screen2Words: Automatic mobile UI summarization with multimodal learning. In *Proceedings of the 34th Annual ACM Symposium on User Interface Software and Technology*, pp. 498–510, 2021.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(8):5362–5383, 2024. URL https://arxiv.org/abs/2302.00487.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 10658–10671. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-emnlp.715. URL https://aclanthology.org/2023.findings-emnlp.715/.

Zifeng Wang, Zheng Zhan, Yifan Gong, Geng Yuan, Wei Niu, Tong Jian, Bin Ren, Stratis Ioannidis, Yanzhi Wang, and Jennifer Dy. SparCL: Sparse continual learning on the edge. In *Advances in Neural Information Processing Systems*, volume 35, pp. 20366–20380, 2022. doi: 10.52202/068431-1481. URL https://doi.org/10.52202/068431-1481.

Taiqiang Wu, Jiahao Wang, Zhe Zhao, and Ngai Wong. Mixture-of-subspaces in low-rank adaptation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 7880–7899. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.450. URL https://aclanthology.org/2024.emnlp-main.450/.

Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A survey on multimodal large language models. *National Science Review*, 11(12):nwae403, 2024. doi: 10.1093/nsr/nwae403. URL https://doi.org/10.1093/nsr/nwae403.

Kaining Ying, Fanqing Meng, Jin Wang, Zhiqian Li, Han Lin, Yue Yang, Hao Zhang, Wenbo Zhang, Yuqi Lin, Shuo Liu, Jiayi Lei, Quanfeng Lu, Runjian Chen, Peng Xu, Renrui Zhang, Haozhe Zhang, Peng Gao, Yali Wang, Yu Qiao, Ping Luo, Kaipeng Zhang, and Wenqi Shao. MMT-bench: A comprehensive multimodal benchmark for evaluating large vision-language models towards multitask AGI. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 57116–57198. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/ying24a.html.

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*, pp. 12126–12141. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.609. URL https://aclanthology.org/2025.emnlp-main.609/.

Duzhen Zhang, Yong Ren, Zhong-Zhi Li, Yahan Yu, Jiahua Dong, Chenxing Li, Zhilong Ji, and Jinfeng Bai. Enhancing multimodal continual instruction tuning with BranchLoRA. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5743–5756. Association for Computational Linguistics, 2025a. doi: 10.18653/v1/2025.acl-long.287. URL https://aclanthology.org/2025.acl-long.287/.

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

Renrui Zhang, Xinyu Wei, Dongzhi Jiang, Ziyu Guo, Yichi Zhang, Chengzhuo Tong, Jiaming Liu, Aojun Zhou, Shanghang Zhang, Gao Peng, and Hongsheng Li. Mavis: Mathematical visual instruction tuning with an automatic data engine. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 87955–87989, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/db36dcad6baee298a34ffca324b84b09-Paper-Conference.pdf.

Hongbo Zhao, Fei Zhu, Haiyang Guo, Meng Wang, Rundong Wang, Gaofeng Meng, and Zhaoxiang Zhang. MLLM-CL: Continual learning for multimodal large language models. *arXiv preprint arXiv:2506.05453*, 2025. URL https://arxiv.org/abs/2506.05453.

Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. MiniGPT-4: Enhancing vision-language understanding with advanced large language models. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2304.10592.

BRACE: Benefit-Aware Reuse and Adaptive Capacity Expansion for Multimodal Continual Instruction Tuning

Contents:

A Complete Method Details **17**
- A.1 Overview of the BRACE Pipeline 17
- A.2 Atom Parameterization and Module Assembly 19
- A.3 Benefit Estimation and Reuse Selection 19
- A.4 Budget Construction and Residual Allocation 20
- A.5 Input Diversity and Optimization Details 21
- A.6 Catalog Registration and Storage 22
- A.7 Task-ID-Free Catalog Retrieval 22

B Training and Implementation Details **23**

C Code, Datasets and Metrics **23**
- C.1 Code 23
- C.2 Datasets 23
- C.3 Evaluation Metrics 26
- C.4 Details 28

D Additional Experiments and Analysis **28**
- D.1 Historical Reuse and Adaptive Capacity Allocation 28
- D.2 Capacity Efficiency and Input-Direction Redundancy 29
- D.3 Evaluation Across Backbones and Capabilities 31
- D.4 Qualitative Analysis 32
- D.5 Robustness to Optimization and Task Order 33

E Theoretical Analysis **34**

F Limitations and Future Directions **36**

A Complete Method Details

A.1 Overview of the BRACE Pipeline

BRACE combines benefit-aware historical reuse with adaptive capacity expansion for sequential task learning. Algorithm 1 summarizes training: FixedAtom selects frozen historical Atoms, AdaAtom allocates residual capacity across modules, and input diversity regularizes the newly trained components. The new Atoms are then frozen and registered with a task catalog, preserving existing references while expanding the pool available to later tasks. Algorithm 2 describes task-ID-free inference: a calibrated prototype router selects a catalog from the input, and only its referenced Atoms are activated for prediction. Table 7 summarizes the principal symbols.

Algorithm 1: Benefit-aware Reuse and Adaptive Capacity Expansion (BRACE)

Require: Tasks $\{\mathcal{D}_t\}_{t=1}^{T}$; frozen backbone Θ_0; budgets R, $R_{\min}$, q, ρ; depths $\mathcal{K}$; scale s; ID weight λ_{ID}
Ensure : Atom pool $\mathcal{P}$; catalogs $\{\mathcal{C}_t\}$; routing prototypes $\{\mathcal{Q}_t\}$ and calibration states

$\mathcal{P} \leftarrow \varnothing$
for $t = 1, \ldots, T$ **do**
  1. FixedAtom.
  if $t = 1$ **then**
    $\mathcal{S}_{t,\ell} \leftarrow \varnothing, \quad k_{t,\ell} \leftarrow R \quad (\forall \ell)$
  else
    Form image-disjoint $\mathcal{D}_t^{\mathrm{dis}}$, $\mathcal{D}_t^{\mathrm{sta}}$, $\mathcal{D}_t^{\mathrm{val}}$ and eligible pools $\mathcal{P}_{<t}^{(\ell)} \subseteq \mathcal{P}$
    Estimate the first-order removal Benefit $\widetilde{\beta}_u = -\,\partial\mathcal{L}/\partial g_u|_{g_u=1}$ and standardized scores $z_u^{\mathrm{dis}}, z_u^{\mathrm{sta}}$ (Eqs. 2 and 15)
    foreach $k \in \mathcal{K}$ **do**
      $\mathcal{S}_{t,\ell}^{(k)} \leftarrow \mathrm{TopK}(\mathcal{P}_{<t}^{(\ell)}, \min(z_u^{\mathrm{dis}}, z_u^{\mathrm{sta}}), k)$ for all ℓ (Eq. 3)
      $\mathcal{S}_t^{(k)} \leftarrow \{\mathcal{S}_{t,\ell}^{(k)}\}_{\ell=1}^{L}$
      Evaluate the complete configuration against zero reuse and compute $\overline{B}_t(k), \mathrm{SE}_t(k)$ (Eqs. 16 and 17)
    end
    $k^\star \leftarrow \arg\max_{k\in\mathcal{K}} \overline{B}_t(k)$, $\tau_t \leftarrow \overline{B}_t(k^\star) - \mathrm{SE}_t(k^\star)$
    $\hat{k} \leftarrow \min\{k \in \mathcal{K} : \overline{B}_t(k) > 0,\ \overline{B}_t(k) \geq \tau_t\}$; fallback: $\hat{k} \leftarrow k^\star$
    $\mathcal{S}_{t,\ell} \leftarrow \mathcal{S}_{t,\ell}^{(\hat{k})}$ for all ℓ
  2. AdaAtom.
  Compute e_t, $f_{t,\ell}$, F_t, K_{cap} (Eqs. 19, 20, and 21)
  $k_{t,\ell} \leftarrow f_{t,\ell}$ for all ℓ
  $K_t \leftarrow \mathrm{clip}_{[F_t, K_{\mathrm{cap}}]}\left[F_t + q\,\mathrm{round}\left(\frac{(1-e_t)(K_{\mathrm{cap}}-F_t)}{q}\right)\right]$ (Eq. 6)
  Compute $b_{t,\ell}, r_{t,\ell}$ (Eq. 7); if all $b_{t,\ell} = 0$, set $r_{t,\ell} = 1$
  while $\sum_\ell k_{t,\ell} < K_t$ **do**
    $\ell^\dagger \leftarrow \arg\max_{\ell: k_{t,\ell} < R} r_{t,\ell}$; ties: smaller $k_{t,\ell}$, then fixed module order
    $k_{t,\ell^\dagger} \leftarrow k_{t,\ell^\dagger} + q$ (Eq. 8)
  end
end
  3. Adaptation and registration.
  Create $m_{t,\ell} = k_{t,\ell} - |\mathcal{S}_{t,\ell}|$ new Atoms $\mathcal{N}_{t,\ell}$ (Eq. 9) and $\mathcal{N}_t = \{\mathcal{N}_{t,\ell}\}_{\ell=1}^{L}$
  Keep Θ_0 and all historical Atoms frozen; assemble $\Delta W_t^{(\ell)}$ using Eq. 14; optimize only $\mathcal{N}_t$ with $\mathcal{L}_{\mathrm{CE}} + \lambda_{\mathrm{ID}}\mathcal{L}_{\mathrm{ID}}$ (Eq. 12)
  Freeze $\mathcal{N}_t$ and update $\mathcal{P} \leftarrow \mathcal{P} \cup \bigcup_{\ell=1}^{L} \mathcal{N}_{t,\ell}$
  Register $\mathcal{C}_t \leftarrow \{\mathcal{S}_{t,\ell} \cup \mathcal{N}_{t,\ell}\}_{\ell=1}^{L}$ (Eq. 10); fit $\mathcal{Q}_t$ and store $\{d_{t,j}^{\mathrm{cal}}\}_{j=1}^{n_t}$ using disjoint calibration images (Eq. 25)
end
return $\mathcal{P}$, $\{\mathcal{C}_t, \mathcal{Q}_t, \{d_{t,j}^{\mathrm{cal}}\}_{j=1}^{n_t}\}_{t=1}^{T}$

Algorithm 2: Task-ID-Free Inference

Require: Input (I, x); frozen backbone and encoders; registered catalogs, prototypes, and calibration states
Ensure : Prediction $\hat{y}$ without an external task identifier

$z \leftarrow \mathrm{Normalize}(\mathrm{Normalize}(f_{\mathrm{img}}(I)) \oplus \mathrm{Normalize}(f_{\mathrm{text}}(x)))$ (Eq. 25)
foreach *registered task* i **do**
  $d_i(z) \leftarrow 1 - \max_{p\in\mathcal{Q}_i} z^\top p$ (Eq. 26)
  $\pi_i(z) \leftarrow \dfrac{1 + \sum_{j=1}^{n_i} \mathbb{I}[d_{i,j}^{\mathrm{cal}} \geq d_i(z)]}{n_i + 1}$ (Eq. 27)
end
$\hat{t} \leftarrow \arg\max_i \pi_i(z)$; ties: smaller $d_i(z)$, then registration order
Activate only the ordered references in $\mathcal{C}_{\hat{t}}$, assemble their LoRA update using Eq. 14, and return
$\hat{y} \leftarrow f(I, x; \Theta_0, \mathcal{C}_{\hat{t}})$
return $\hat{y}$

Table 7: Notation for BRACE. Each layer denotes an adapted linear module.

Symbol	Description
t	Current task index.
T	Number of tasks in the stream.
ℓ	Index of an adapted module.
L	Number of adapted modules.
$\mathcal{D}_t$	Training data for task t, partitioned into discovery, stability, and validation subsets.
Θ_0	Frozen pretrained backbone.
k_ℓ	Input dimension of module ℓ.
d_ℓ	Output dimension of module ℓ.
s	Fixed Atom scaling coefficient.
$a_u^{(\ell)}$	Input-side factor of Atom u in module ℓ.
$b_u^{(\ell)}$	Output-side factor of Atom u in module ℓ.
$\mathcal{P}$	Stored global Atom pool.
$\mathcal{P}_{<t}^{(\ell)}$	Eligible historical candidates at module ℓ before task t.
$\mathcal{S}_{t,\ell}$	Historical Atoms selected for reuse at module ℓ.
$\mathcal{S}_t^{(k)}$	Complete multi-module reuse configuration at depth k.
$\mathcal{N}_{t,\ell}$	Newly created Atoms at module ℓ.
$\mathcal{C}_t$	Active Atom-reference catalog registered for task t.
$\mathcal{R}_{t,\ell}$	Ordered per-module reference list used for assembly.
$A_{\mathcal{C}_t}^{(\ell)}$	Assembled LoRA input-factor matrix.
$B_{\mathcal{C}_t}^{(\ell)}$	Assembled LoRA output-factor matrix.
$\Delta W_t^{(\ell)}$	LoRA update assembled for task t at module ℓ.
$\beta_u(x)$	Exact loss increase caused by removing Atom u.
$\widetilde{\beta}_u(x)$	First-order estimate of the removal benefit.
g_u	Differentiable gate for Atom u.
$z_u^{(v)}$	Standardized benefit on probe subset v.
$\mathcal{K}$	Candidate reuse depths.
$k^\star$	Depth with the largest mean validation configuration benefit.
$\hat{k}$	Selected common reuse depth.
$B_i(k)$	validation configuration benefit of depth k on sample i.
$\overline{B}_t(k)$	Mean validation configuration benefit for task t at depth k.
$\mathrm{SE}_t(k)$	Paired standard error of the mean configuration benefit.
τ_t	One-standard-error selection threshold.
e_t	Clipped relative validation loss reduction from reuse.
R	Maximum active Atom count per module.
$R_{\min}$	Minimum active Atom count per module.
q	Allocation quantum.
ρ	Global capacity-cap ratio.
$f_{t,\ell}$	Reuse-preserving active floor at module ℓ.
F_t	Sum of the module-wise active floors.
K_{cap}	Effective global active-capacity cap.
K_t	Task-level budget of shared and new Atoms.
$b_{t,\ell}$	Historical-support score of module ℓ.
$r_{t,\ell}$	Relative residual-demand score of module ℓ.
$k_{t,\ell}$	Final active Atom count at module ℓ.
$m_{t,\ell}$	Number of newly created Atoms at module ℓ.
$\mathcal{M}_t$	Modules containing at least two new Atoms.
$\mathcal{L}$	Joint training loss.
$\mathcal{L}_{\mathrm{CE}}$	Answer-token cross-entropy loss.
$\mathcal{L}_{\mathrm{ID}}$	Input Diversity loss.
λ_{ID}	Weight of the Input Diversity loss.
P_t^{new}	Atom parameters first created for task t.
$P_{\le T}^{\mathrm{unique}}$	Cumulative unique Atom parameters through task T.
$z(I,x)$	Fused image–instruction routing feature.
$\mathcal{Q}_t$	Unit-norm routing prototypes for task t.
$d_t(z)$	Distance from query feature z to task t.
$d_{t,j}^{\mathrm{cal}}$	Calibration distance for example j of task t.
n_t	Number of calibration examples for task t.
$\pi_t(z)$	Calibrated typicality score for task t.
$\hat{t}$	Selected task-catalog index.

A.2 Atom Parameterization and Module Assembly

Atom representation. For an adapted linear layer ℓ with input dimension k_ℓ and output dimension d_ℓ, each Atom u consists of a paired input factor $a_u^{(\ell)} \in \mathbb{R}^{k_\ell}$ and output factor $b_u^{(\ell)} \in \mathbb{R}^{d_\ell}$. Its contribution to the weight update is the rank-one term $s\, b_u^{(\ell)} a_u^{(\ell)\top}$. BRACE therefore selects and reuses individual factor pairs from historical tasks, rather than entire task-specific LoRA adapters.

Task-specific assembly. For task t, the active Atom set in module ℓ combines the selected historical Atoms $\mathcal{S}_{t,\ell}$ with the newly created Atoms $\mathcal{N}_{t,\ell}$. Let $\mathcal{R}_{t,\ell} = (u_1, \ldots, u_{k_{t,\ell}})$ be an ordered list of these references, where $k_{t,\ell} = |\mathcal{S}_{t,\ell}| + |\mathcal{N}_{t,\ell}|$. The effective LoRA matrices are

$$A_{\mathcal{C}_t}^{(\ell)} = \begin{bmatrix} a_{u_1}^{(\ell)\top} \\ \vdots \\ a_{u_{k_{t,\ell}}}^{(\ell)\top} \end{bmatrix}, \qquad B_{\mathcal{C}_t}^{(\ell)} = \begin{bmatrix} b_{u_1}^{(\ell)} & \cdots & b_{u_{k_{t,\ell}}}^{(\ell)} \end{bmatrix}. \tag{13}$$

Their shapes are $k_{t,\ell} \times k_\ell$ and $d_\ell \times k_{t,\ell}$, respectively. The resulting update is

$$\begin{aligned} \Delta W_t^{(\ell)} &= s B_{\mathcal{C}_t}^{(\ell)} A_{\mathcal{C}_t}^{(\ell)} \\ &= \underbrace{s \sum_{u \in \mathcal{S}_{t,\ell}} b_u^{(\ell)} a_u^{(\ell)\top}}_{\text{frozen historical update}} + \underbrace{s \sum_{u \in \mathcal{N}_{t,\ell}} b_u^{(\ell)} a_u^{(\ell)\top}}_{\text{trainable residual update}}. \end{aligned} \tag{14}$$

This assembly specifies the factors used in the current computation. Historical Atoms remain stored under their originating tasks and are referenced by subsequent catalogs, without duplicating their persistent parameter storage.

Initialization and optimization. New input factors are randomly initialized, and new output factors are initialized to zero. The residual update is therefore initially zero, so training starts from the selected historical update. During task learning, only the newly created factor pairs are optimized; the base model and all historical factors remain frozen. After training, the new Atoms are frozen and their references are registered together with the selected historical references in the task catalog. The registered scale s remains fixed and is not recomputed from the number of Atoms activated by a later catalog.

A.3 Benefit Estimation and Reuse Selection

Probe construction. For a task with historical Atoms, the implementation scans the current training records in their stored order, keeps the first record for each image, and assigns consecutive unique images to discovery, stability, and validation subsets. Their sizes are configurable; an insufficient number of unique images raises an explicit error. These subsets are selection holdouts rather than test data, and their examples may remain available to the subsequent task-training stage. The selected indices and image digests are stored to make the split reproducible.

Gate-based candidate scoring. The per-sample removal benefit is defined by Eq. 2 and is computed before any new Atom is trained.A positive removal benefit means that removing the Atom increases the current-task loss, indicating that retaining the Atom is beneficial. During a probe pass, every candidate historical Atom in the layer-wise pool is active with gate one. Its scalar response is multiplied by a differentiable scalar gate, and one backward pass returns the derivatives for all gates. The backbone and every historical Atom remain frozen throughout this operation. The resulting first-order scores are therefore conditional on the jointly active candidate pool: they estimate the loss change from removing one component while the other candidates stay in place, rather than the gain from independently adding that component to an empty adapter.

To place the noisy probe estimates on a comparable scale and make the ranking conservative across the discovery and stability splits, we use the following standardized score for subset s:

$$z_u^{(s)} = \frac{\bar{\beta}_u^{(s)} \sqrt{n_s}}{\max(\hat{\sigma}_u^{(s)}, \epsilon)}, \tag{15}$$

where n_s is the number of probes in subset s, $\widetilde{\overline{\beta}}_u^{(s)}$ is the sample-mean removal benefit for Atom u, $\widehat{\sigma}_u^{(s)}$ is its sample standard deviation computed with denominator $n_s - 1$, and $\epsilon > 0$ is a numerical floor. Thus, the denominator is the standard error after clamping the sample standard deviation, rather than the raw standard deviation. Eq. 3 ranks each Atom by the smaller score from the discovery and stability subsets, so a candidate that is strong on only one subset is not promoted. Ties are resolved by the smaller mean configuration benefit and then by Atom identifier.

Configuration-level validation. The ranking above produces candidate identities, but the reuse depth must be validated as a complete multi-layer configuration. For each candidate depth $k \in \mathcal{K}$, every adapted layer activates its own top-k candidates. For validation example i, the configuration-level benefit is defined as the loss reduction

$$B_i(k) = \mathcal{L}_i(\varnothing) - \mathcal{L}_i(\mathcal{S}_t^{(k)}), \tag{16}$$

where $\mathcal{L}_i(\varnothing)$ is the loss with all historical updates disabled and $\mathcal{L}_i(\mathcal{S}_t^{(k)})$ is the loss under the complete top-k configuration. Both losses use the same frozen model, preprocessing, and validation example. We then summarize the paired Benefits and their sampling uncertainty with

$$\overline{B}(k) = \frac{1}{n}\sum_i B_i(k), \qquad \mathrm{SE}(k) = \sqrt{\frac{\sum_i (B_i(k) - \overline{B}(k))^2}{n(n-1)}}. \tag{17}$$

where n is the number of validation examples, $\overline{B}(k)$ is the mean configuration benefit, and $\mathrm{SE}(k)$ is its paired standard error. The best depth and the one-standard-error threshold are defined by the reuse-depth rule in the main method; no second threshold equation is introduced here. The rule chooses the smallest depth with a positive configuration benefit within one standard error of the best mean. If no depth is eligible, it falls back to the depth with the largest mean configuration benefit. The shared depth is common across layers, whereas the Atom identities remain layer specific.

The normalized validation reuse gain e_t passed to AdaAtom is measured from the complete selected configuration, not obtained by summing per-Atom gate scores. This distinction retains cross-layer interactions in the budget signal.

Optional paired diagnostic. As an optional diagnostic, the implementation can test whether the paired configuration benefits are systematically positive under random sign flips. With M random sign assignments, the Monte Carlo tail estimate is

$$p_{\mathrm{flip}} = \frac{1 + \sum_{b=1}^{M} \mathbb{I}[\overline{B}^{(b)} \geq \overline{B}]}{M+1}, \tag{18}$$

where $\overline{B}^{(b)}$ is the mean after independently flipping the signs of the paired differences in random assignment b, and $\overline{B}$ is the observed mean. This diagnostic is separate from the depth rule, requires a sign-symmetry null, and is not used to select Atoms or capacity.

A.4 Budget Construction and Residual Allocation

Connecting reuse evidence to the task budget. AdaAtom receives the validation losses of the selected reuse configuration. Let $\overline{\mathcal{L}}_{\mathrm{base}}$ be the mean loss with zero historical reuse and $\overline{\mathcal{L}}_{\mathrm{shared}}$ the mean loss with the selected shared configuration. To convert this complete-configuration evidence into a task-level residual-capacity signal, we define the clipped relative validation loss reduction e_t

$$e_t = \mathrm{clip}_{[0,1]}\left(\frac{\overline{\mathcal{L}}_{\mathrm{base}} - \overline{\mathcal{L}}_{\mathrm{shared}}}{\overline{\mathcal{L}}_{\mathrm{base}}}\right). \tag{19}$$

where $\mathrm{clip}_{[0,1]}$ restricts the ratio to $[0, 1]$, $\overline{\mathcal{L}}_{\mathrm{base}}$ is the mean loss with zero historical reuse, and $\overline{\mathcal{L}}_{\mathrm{shared}}$ is the mean loss with the selected configuration. Both losses use the same validation examples. Thus, e_t is a normalized validation reuse gain rather than a literal fraction of the task's knowledge: a larger value contracts the active budget toward its floor, while a smaller value leaves more capacity for residual learning.

Floors and effective cap. Let R denote the maximum number of active Atoms allowed in one adapted module. Thus, R is the per-module rank budget and is fixed before training; it is chosen

to match the rank budget of the corresponding fixed-rank baseline, rather than learned from the current task. Let $R_{\min}$ denote the minimum active count per module and let q denote the allocation quantum. The implementation uses four coarse quanta by default, $q = \max(1, \lfloor R/4 \rfloor)$, and requires q to divide R and $R_{\min}$ to be a multiple of q. In the reported protocols, $R = 16$ for UCIT and $R = 32$ for the rank-32 MLLM-DCL setting; $R_{\min} = 8$, giving $q = 4$ and $q = 8$, respectively. These values preserve a nontrivial residual-learning floor while keeping the allocation search at a small number of discrete capacity levels. To retain every selected reference while respecting the quantum, the floor for module ℓ is

$$f_{t,\ell} = \max\left(R_{\min}, q\left\lceil \frac{|\mathcal{S}_{t,\ell}|}{q} \right\rceil\right), \qquad F_t = \sum_{\ell} f_{t,\ell}. \tag{20}$$

The preceding floor is module-wise. To impose a feasible global upper bound on the total number of active Atom slots, we define

$$K_{\text{cap}} = \max\left\{F_t,\ \min\left[LR,\ q\ \text{round}\left(\frac{\rho LR}{q}\right)\right]\right\}. \tag{21}$$

where L is the number of adapted modules, R is the per-module active cap, q is the allocation quantum, and ρ is the configured global-cap ratio. Here LR is the maximum number of active Atom slots, not a parameter count. The ratio ρ is an upper-cap safeguard rather than the source of the task-specific budget; the reported full configurations use $\rho = 1$, so the loss-based evidence determines the effective contraction. A smaller ρ can be used when a stricter global memory limit is required. If the requested cap falls below the floors, preserving all selected references takes precedence. The main-text budget equation, Eq. 6, then maps the measured reuse gain to this feasible interval and rounds the residual capacity to the nearest quantum using the implementation's ties-to-even convention. These settings are fixed protocol hyperparameters selected for rank comparability, minimum residual capacity, and coarse-grained allocation stability; they are not adjusted using the test set.

Layer priorities. The global target determines how many slots remain, while the layer rule in Eq. 8 determines where those slots are placed. Each layer's support is computed by summing the positive parts of the mean Atom-level removal benefits estimated on the discovery set over all eligible candidates. Layers with weaker historical support receive higher residual priority. A layer may therefore have strong support even when none of its candidates is in the final selected set. The support values are held fixed during allocation; ties favor the smaller current count and then a deterministic module-name order. A layer can receive several quanta before the procedure moves to another unsaturated layer.

New-Atom count. Allocation stops when the task budget in Eq. 6 is reached. The new count at each layer is the final active count minus the number of selected historical references, as described in the main method. The first task bypasses reuse selection and creates R new Atoms per module. No historical component is deleted or modified by budget construction or layer allocation.

A.5 Input Diversity and Optimization Details

The main text defines the Input Diversity penalty in Eq. 11. The appendix gives its numerically stable Gram-matrix implementation, which avoids explicitly enumerating all Atom pairs. For a layer with $m_{t,\ell} \geq 2$ newly created Atoms, let $\widehat{A}_\ell$ stack the normalized input factors as rows and define $G_\ell = \widehat{A}_\ell \widehat{A}_\ell^\top$. The per-layer penalty is

$$D_\ell = \frac{\|G_\ell - \text{diag}(G_\ell)\|_F^2}{m_{t,\ell}(m_{t,\ell} - 1)}. \tag{22}$$

where each row of $\widehat{A}_\ell$ is $a_i^\top / \max(\|a_i\|_2, \epsilon_a)$, G_ℓ is the resulting Gram matrix, and $\text{diag}(G_\ell)$ retains only its diagonal. Removing the actual diagonal, rather than subtracting an identity matrix, makes zero-norm rows well-defined. Layers with fewer than two new Atoms are omitted and an empty eligible set gives zero ID loss. Above the numerical norm floor, this expression is equivalent to the pairwise squared-cosine penalty in the main text.

The joint optimization objective is Eq. 12. To make clear which tokens contribute to its task term, the answer-only cross-entropy is

$$\mathcal{L}_{\text{CE}}(x) = -\frac{1}{|\mathcal{Y}(x)|} \sum_{j \in \mathcal{Y}(x)} \log p_\theta(y_j \mid I, x, y_{<j}), \tag{23}$$

where $\mathcal{Y}(x)$ is the set of unmasked answer-token positions, y_j is the target token at position j, and p_θ is the model distribution under the assembled catalog. During optimization, the backbone and all historical factors are frozen; only newly created Atoms receive gradients. The ID term differentiates through their input factors, while the cross-entropy term updates both their input and output factors. No orthogonality penalty between new and historical Atoms is implied.

A.6 Catalog Registration and Storage

The catalog membership relation is already given in Eq. 10. Its implementation stores, for every active reference, the module name, owner task, owner slot, ordered position, and scale. After a task finishes, only its newly created factors are appended to the owner-based Atom pool. Existing factors and all earlier catalog references are left unchanged. A later catalog can therefore point to the same physical Atom without creating another tensor copy. The feature encoder and the registered prototype/calibration state are likewise frozen after registration.

For storage accounting, an Atom contributes physical parameters only when it is first created. The newly added count and cumulative unique adapter storage are therefore

$$P_t^{\text{new}} = \sum_{\ell} m_{t,\ell}(d_\ell + k_\ell), \qquad P_{\le T}^{\text{unique}} = \sum_{t=1}^{T} P_t^{\text{new}}. \tag{24}$$

where $m_{t,\ell}$ is the number of newly created Atoms in module ℓ, $k_{t,\ell}$ is the number of active Atoms in that module, and $d_\ell + k_\ell$ is the parameter cost of one rank-one factor pair. Shared references are counted once in physical storage even when used by multiple catalogs. Activated parameter counts replace $m_{t,\ell}$ with $k_{t,\ell}$ and can therefore exceed the unique stored count. Deployment bytes additionally depend on precision and catalog/router metadata; the frozen backbone is excluded from adapter-storage comparisons.

A.7 Task-ID-Free Catalog Retrieval

Frozen routing representation. For an image–instruction pair (I, x), the router first constructs a fused feature for task-catalog retrieval by concatenating independently normalized image and instruction features:

$$z(I, x) = \text{Normalize}(\text{Normalize}(f_{\text{img}}(I)) \oplus \text{Normalize}(f_{\text{text}}(x))) \,. \tag{25}$$

where f_{img} and f_{text} are the frozen image and instruction encoders, $\text{Normalize}(\cdot)$ denotes feature normalization, and $\oplus$ denotes concatenation. For each task, spherical k-means on its fit features produces the unit-norm prototype set $\mathcal{Q}_t$; calibration examples are disjoint from those used to fit the prototypes.

Distance calibration and hard retrieval. The router next measures how close the fused feature is to the nearest prototype of task t. The resulting task distance is

$$d_t(z) = 1 - \max_{p \in \mathcal{Q}_t} z^\top p. \tag{26}$$

where $\mathcal{Q}_t$ is the prototype set for task t and the inner product $z^\top p$ is the cosine similarity because both vectors are unit normalized. To calibrate this distance against the task's own feature distribution, let $d_{t,j}^{\text{cal}}$ be the distances of its n_t calibration examples. The empirical typicality score is

$$\pi_t(z) = \frac{1 + \sum_{j=1}^{n_t} \mathbb{I}[d_{t,j}^{\text{cal}} \ge d_t(z)]}{n_t + 1}. \tag{27}$$

where $\mathbb{I}[\cdot]$ is the indicator function and the numerator counts calibration distances no smaller than the query distance. This empirical rank is a per-task typicality score, not a normalized class probability. The router selects the catalog with the largest typicality, breaking ties by smaller prototype distance and then registration order. It activates only the references in the selected catalog, so task-ID-free inference is a hard catalog lookup that neither blends catalogs nor alters any Atom tensor.

B Training and Implementation Details

We implement BRACE in PyTorch and evaluate it on the UCIT (Guo et al., 2025a), MLLM-DCL (Zhao et al., 2025), and MLLM-ACL (Guo et al., 2025d) multimodal continual-learning benchmarks. All three benchmarks follow the same sequential-learning protocol: tasks arrive one at a time, training data from previous tasks are not replayed, and the base model, historical Atoms, and registered catalogs remain frozen while only the Atoms newly created for the current task are optimized (Guo et al., 2025d). Once a task has been learned, its new Atoms are frozen and added to the global candidate pool for selection and reuse by subsequent tasks. Historical-Atom reuse, adaptive capacity allocation, and Input Diversity regularization follow the same procedure across all benchmarks, with $\lambda_{\mathrm{ID}} = 0.20$. All compared methods use the same base model, data preprocessing, and task order. At test time, routing and answer generation are performed without access to the ground-truth Task ID (Guo et al., 2025d). All experiments are conducted on a single NVIDIA A800 80GB GPU.

C Code, Datasets and Metrics

C.1 Code

We will release the code after publication to facilitate reproducibility.

C.2 Datasets

To evaluate whether BRACE can acquire new multimodal capabilities while retaining previously learned knowledge, we conduct experiments on three complementary continual instruction-tuning benchmarks: UCIT (Guo et al., 2025a), MLLM-DCL (Zhao et al., 2025), and MLLM-ACL (Guo et al., 2025d). UCIT evaluates continual learning across heterogeneous task formats, MLLM-DCL focuses on sequential adaptation across specialized domains, and MLLM-ACL introduces capability-level transitions. Together, they cover task, domain, and capability shifts under the same Atom-based adaptation framework. Their statistics are summarized in Table 8.

The UCIT benchmark evaluates generalization across heterogeneous multimodal tasks while limiting information leakage between task formats. It contains six sequential tasks with distinct visual content, answer spaces, and instruction structures: ImageNet-R (Hendrycks et al., 2021) for out-of-distribution visual classification, ArxivQA (Li et al., 2024a) for scientific chart understanding and reasoning, VizWiz (Gurari et al., 2018) for low-quality visual question answering, IconQA (Lu et al., 2021) for abstract diagram reasoning, CLEVR-Math (Lindström & Abraham, 2022) for compositional mathematical reasoning, and Flickr30k (Plummer et al., 2015) for image captioning. UCIT contains approximately 214k training samples, and we evaluate 0.3k test examples for each task. Representative examples are shown in Figure 4.

The MLLM-DCL benchmark evaluates continual adaptation across specialized domains. It consists of five task groups: remote-sensing question answering with RSVQA (Lobry et al., 2020), medical visual question answering with PathVQA (He et al., 2020), autonomous-driving reasoning with DriveLM (Sima et al., 2025), scientific reasoning with AI2D (Kembhavi et al., 2016), SciVerse (Guo et al., 2025e), MapQA (Chang et al., 2022), and TQA (Kembhavi et al., 2017), and financial analysis with StockQA (Zhao et al., 2025). These groups arrive in the fixed sequence RS, Med, AD, Sci, and Fin. The benchmark contains more than 236k training samples and approximately 48k test samples. Its specialized terminology and domain-specific visual evidence test whether the model can preserve earlier domain knowledge while allocating sufficient capacity to new domains. Representative examples are shown in Figure 5.

MLLM-ACL (Zhao et al., 2025) evaluates capability-level continual adaptation through the sequence OCR, Math, VP, and APP. We follow the dataset configuration summarized in MCITlib (Guo et al., 2025d). OCR trains on Monkey (Li et al., 2024b) and evaluates text recognition and understanding on OCRBench (Liu et al., 2024b). Math uses MathV360K (Shi et al., 2024) and MAVIS (Zhang et al., 2025b) for training and evaluates visual mathematical reasoning on MathVista (Lu et al., 2024). VP trains on CLEVR (Johnson et al., 2017) and TallyQA (Acharya et al., 2019) and evaluates visual perception and counting on CV-Bench (Tong et al., 2024). APP uses

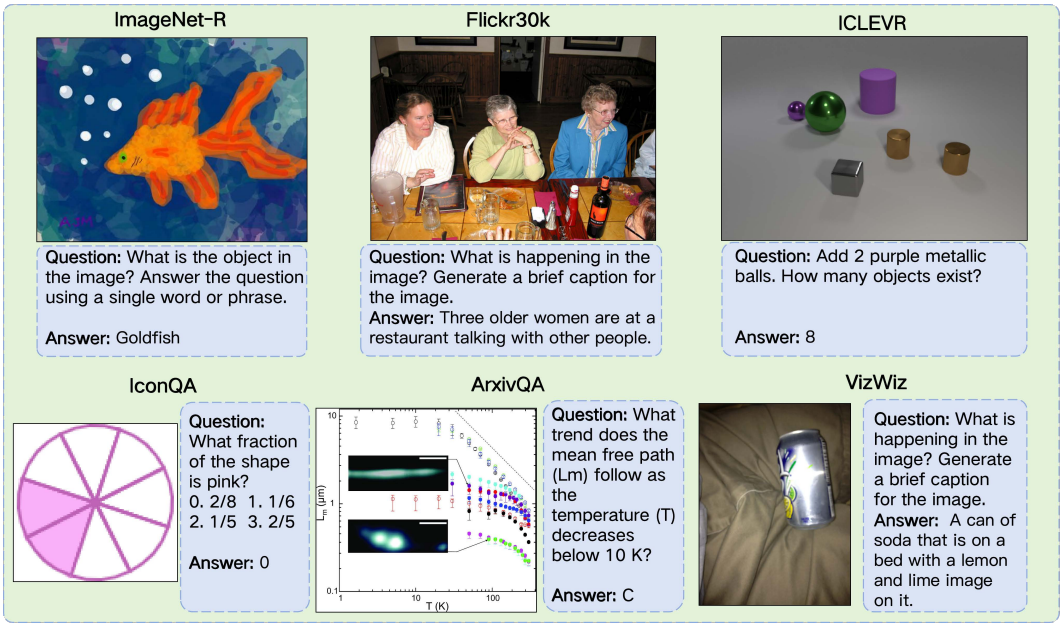


Figure 4: Sample visualization of the UCIT benchmark. The benchmark covers diverse general-purpose multimodal instruction-following tasks, including image classification, image captioning, visual reasoning, chart understanding, and visual question answering.

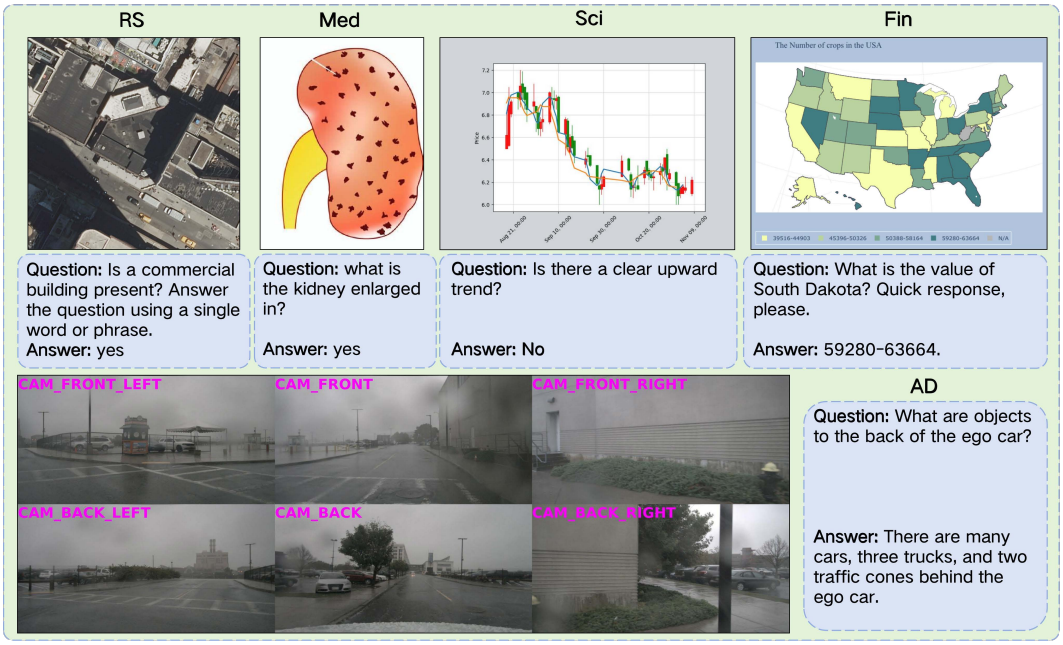


Figure 5: Sample visualization of the MLLM-DCL benchmark. The benchmark includes domain-specific multimodal tasks from remote sensing, medical imaging, scientific chart understanding, financial reasoning, and autonomous driving scenarios.

ScreenQA (Hsiao et al., 2025), MultiUI (Liu et al., 2025a), and Screen2Words (Wang et al., 2021) for training and evaluates GUI navigation using the corresponding subset of MMT-Bench (Ying et al., 2024). According to the dataset statistics reported in MCITlib (Guo et al., 2025d), the four

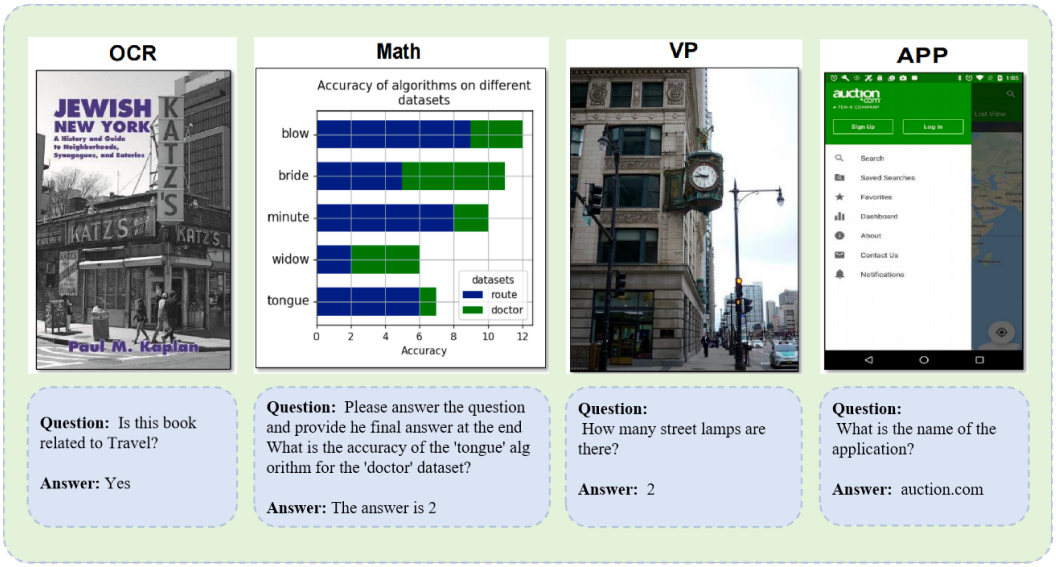


Figure 6: Sample visualization of the MLLM-ACL benchmark.The benchmark covers four capability domains in the sequential order OCR, mathematical reasoning, visual perception, and application-level GUI interaction.

Table 8: Statistics of the training and test datasets for UCIT, MLLM-DCL, and MLLM-ACL.

Task	Train Dataset	Test Dataset	Train Number	Test Number
		UCIT		
ImgNet-R	ImageNet-R	ImageNet-R	24k	0.3k
ArxivQA	ArxivQA	ArxivQA	40k	0.3k
VizWiz	VizWiz	VizWiz	40k	0.3k
IconQA	IconQA	IconQA	30k	0.3k
CLEVR	CLEVR-Math	CLEVR-Math	40k	0.3k
Flickr30k	Flickr30k	Flickr30k	40k	0.3k
		MLLM-DCL		
RS	RSVQA	RSVQA	60k	10k
Med	PathVQA	PathVQA	22.8k	9.8k
AD	DriveLM	DriveLM	60k	10k
Sci	AI2D, SciVerse, MapQA, TQA	AI2D, SciVerse, MapQA, TQA	30.7k (12.4k, 0.9k, 9.6k, 7.8k)	7.6k (3.1k, 0.2k, 2.4k, 1.9k)
Fin	StockQA	StockQA	60k	10k
		MLLM-ACL		
OCR	Monkey	OCRBench	128.1k	1k
Math	MathV360K, MAVIS	MathVista	526.1k	1k
VP	CLEVR, TallyQA	CV-Bench	119.9k	0.8k
GUI Agent	ScreenQA, MultiUI Screen2Words	MMTBench	147.3k	0.8k

groups contain approximately 921.4k training samples and 3.6k test samples in total. Because the training and evaluation datasets differ within each capability, this setup also allows us to assess whether accumulated Atoms support transfer beyond their training data sources.

The three benchmarks expose complementary challenges. UCIT tests reuse across heterogeneous visual semantics and instruction formats; MLLM-DCL tests retention under substantial domain shifts; and MLLM-ACL tests adaptation across distinct multimodal capabilities and evaluation datasets. BRACE uses the same procedure throughout: FixedAtom selects historical Atoms by their measured Atom-level removal benefits, AdaAtom allocates new Atoms according to residual adaptation demand, and Input Diversity reduces directional redundancy among new Atoms. This unified setting allows performance and adaptation overhead to be compared without introducing dataset-specific training mechanisms.

C.3 Evaluation Metrics

Following the evaluation protocol of SEFE (Chen et al., 2025), we assess multimodal continual instruction tuning with four complementary metrics: Mean Finetune Accuracy (MFT), Mean Final Accuracy (MFN), Mean Average Accuracy (MAA), and Backward Transfer (BWT). All four metrics are computed from the same Task-ID-free lower-triangular performance matrix. At each stage, test samples from all observed tasks are routed without their ground-truth Task IDs, so the matrix measures end-to-end performance of both the selected Atom catalog and the subsequent answer generation. Figure 7 provides an intuitive visualization of the entries used by each metric.

Given a continual learning setting with T sequential tasks, let $A_{i,k}$ denote the performance on task k after completing the training of task i. Each entry is evaluated with the scoring function associated with task k and is reported on a common percentage scale. Only entries with $k \leq i$ are defined because task k has not yet been observed at earlier stages.

Mean Finetune Accuracy (MFT) measures the capability to learn newly introduced tasks. It is computed from the diagonal entries of the performance matrix, which correspond to the performance of each task immediately after its own training stage:

$$\text{MFT} = \frac{1}{T}\sum_{i=1}^{T} A_{i,i}. \tag{28}$$

Because subsequent tasks have not yet been learned at the time of $A_{i,i}$, MFT primarily reflects forward plasticity: whether the active Atom budget and the newly created residual Atoms are sufficient to fit an incoming task. It is not a direct measure of later interference, although low MFT can indicate insufficient capacity or ineffective reuse for the new task.

Mean Final Accuracy (MFN) evaluates the knowledge retained after the full task stream has been processed. It averages the performance over all tasks after training the final task T:

$$\text{MFN} = \frac{1}{T}\sum_{i=1}^{T} A_{T,i}. \tag{29}$$

MFN captures the final deployed state of the system. Relative to MFT, it reveals how much of the initially acquired task capability remains accessible after subsequent adaptation. In a Task-ID-free setting, a lower MFN may be caused by changes within the selected catalog path, by later-task interference, or by routing an input to an incorrect catalog; the metric intentionally evaluates their combined deployment effect.

Mean Average Accuracy (MAA) provides a trajectory-level assessment of model performance. After each training stage i, we first average the performance over all tasks observed so far, and then average these intermediate values over the full sequence:

$$\text{MAA} = \frac{1}{T}\sum_{i=1}^{T}\left(\frac{1}{i}\sum_{k=1}^{i} A_{i,k}\right). \tag{30}$$

Unlike MFN, which considers only the final model, MAA reflects the complete adaptation trajectory. It rewards methods that learn new tasks effectively while preserving previously learned capabilities at every intermediate stage, and it avoids masking unstable middle-stage behavior with a single final-state average.

Backward Transfer (BWT) quantifies the influence of subsequent learning on previously acquired knowledge. It is defined as the average difference between the final performance of each task and its performance immediately after initial training:

$$\text{BWT} = \frac{1}{T-1}\sum_{i=1}^{T-1}(A_{T,i} - A_{i,i}). \tag{31}$$

The last task is excluded because no subsequent task can affect it. Negative BWT values indicate that later tasks reduce the performance attained immediately after learning earlier tasks and therefore reflect forgetting. Values close to zero indicate stable retention, whereas positive values

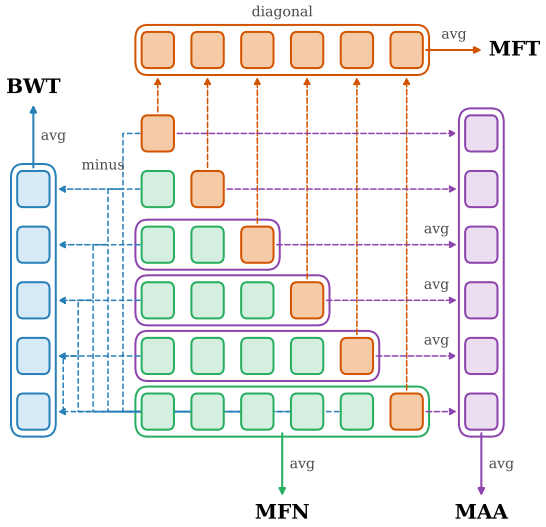


Figure 7: Conceptual visualization of the evaluation metrics used in multimodal continual instruction tuning.

Table 9: Detailed parameter settings of different continual learning methods across LLaVA and InternVL on the UCIT and MLLM-DCL benchmarks.

Methods	LLaVA				InternVL			
	LR	Epoch	PEFT	Parameter setting	LR	Epoch	PEFT	Parameter setting
UCIT Benchmark								
LoRA-FT	2e-4	{1,1,1,1,1,1}	LoRA	rank = 16	1e-4	{1,1,1,1,1,1}	LoRA	rank = 16
O-LoRA	2e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6	1e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6
MoELoRA	2e-4	{1,1,1,1,1,1}	LoRA	rank = 18, expert num = 6	1e-4	{1,1,1,1,1,1}	LoRA	rank = 18, expert num = 6
ModalPrompt	2e-4	{1,1,1,1,1,1}	Prompt	prefix len = 10, expert num = 6	1e-4	{1,1,1,1,1,1}	Prompt	prefix len = 10, expert num = 6
CL-MoE	2e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6	1e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6
HiDe	2e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6	1e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6
SEFE	2e-4	{1,1,1,1,1,1}	LoRA	rank = 16	1e-4	{1,1,1,1,1,1}	LoRA	rank = 16
DISCO	2e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6	1e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6
DGG	2e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6	1e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6
MoE-CL	2e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6	1e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6
PCLR	2e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6	1e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6
BRACE	2e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6	1e-4	{1,1,1,1,1,1}	LoRA	rank = 96, expert num = 6
MLLM-DCL Benchmark								
LoRA-FT	2e-5	{1,3,1,2,1}	LoRA	rank = 32	1e-4	{1,3,1,2,1}	LoRA	rank = 32
O-LoRA	2e-5	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5	1e-4	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5
MoELoRA	2e-5	{1,3,1,2,1}	LoRA	rank = 35, expert num = 5	1e-4	{1,3,1,2,1}	LoRA	rank = 35, expert num = 5
ModalPrompt	2e-4	{1,3,1,2,1}	Prompt	prefix len = 20, expert num = 5	1e-4	{1,3,1,2,1}	Prompt	prefix len = 20, expert num = 5
CL-MoE	2e-5	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5	1e-4	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5
HiDe	2e-5	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5	1e-4	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5
SEFE	2e-5	{1,3,1,2,1}	LoRA	rank = 32	1e-4	{1,3,1,2,1}	LoRA	rank = 32
DISCO	2e-5	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5	1e-4	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5
DGG	2e-5	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5	1e-4	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5
MoE-CL	2e-5	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5	1e-4	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5
PCLR	2e-5	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5	1e-4	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5
BRACE	2e-5	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5	1e-4	{1,3,1,2,1}	LoRA	rank = 160, expert num = 5
MLLM-ACL Benchmark								
LoRA-FT	2e-5	{3,1,1,3}	LoRA	rank = 32	1e-4	{3,1,1,3}	LoRA	rank = 32
O-LoRA	2e-5	{3,1,1,3}	LoRA	rank = 128, expert num = 4	1e-4	{3,1,1,3}	LoRA	rank = 128, expert num = 4
MoELoRA	2e-5	{3,1,1,3}	LoRA	rank = 32, expert num = 4	1e-4	{3,1,1,3}	LoRA	rank = 32, expert num = 4
ModalPrompt	2e-4	{3,1,1,3}	Prompt	prefix len = 20, expert num = 4	1e-4	{3,1,1,3}	Prompt	prefix len = 20, expert num = 4
CL-MoE	2e-5	{3,1,1,3}	LoRA	rank = 128, expert num = 4	1e-4	{3,1,1,3}	LoRA	rank = 128, expert num = 4
HiDe	2e-5	{3,1,1,3}	LoRA	rank = 128, expert num = 4	1e-4	{3,1,1,3}	LoRA	rank = 128, expert num = 4
SEFE	2e-5	{3,1,1,3}	LoRA	rank = 32	1e-4	{3,1,1,3}	LoRA	rank = 32
DISCO	2e-5	{3,1,1,3}	LoRA	rank = 128, expert num = 4	1e-4	{3,1,1,3}	LoRA	rank = 128, expert num = 4
BRACE	2e-5	{3,1,1,3}	LoRA	rank = 128, expert num = 4	1e-4	{3,1,1,3}	LoRA	rank = 128, expert num = 4

indicate positive backward transfer. Together, MFT, MFN, MAA, and BWT distinguish new-task plasticity, final knowledge retention, the stability of the full learning trajectory, and the net effect of subsequent adaptation; reporting only one of these quantities cannot characterize the complete continual-learning behavior of BRACE.

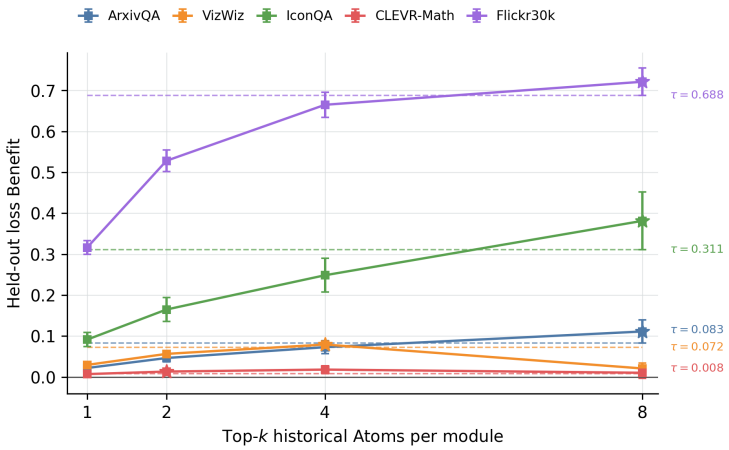


Figure 8: UCIT configuration benefit, measured as the validation-loss reduction relative to zero reuse versus the number k of historical Atoms selected per module. Error bars indicate standard errors, stars mark the selected depths, and dashed horizontal lines mark the task-specific thresholds τ.

C.4 Details

We follow the official LLaVA LoRA fine-tuning recipe and insert LoRA modules into all linear layers of the language model. We use a warm-up ratio of 0.03 with cosine learning-rate decay. On UCIT, all six tasks are trained for one epoch, following the schedule $\{1, 1, 1, 1, 1, 1\}$. On MLLM-DCL, we adopt the task-specific schedule $\{1, 3, 1, 2, 1\}$. On MLLM-ACL, the four tasks, namely OCR, Math, VP, and APP, are trained sequentially with the schedule $\{3, 1, 1, 3\}$. For LLaVA-v1.5-7B, the learning rate is set to 2×10^{-4} on UCIT, 2×10^{-5} on MLLM-DCL, and 2×10^{-5} on MLLM-ACL. For InternVL-Chat-7B, we use a learning rate of 1×10^{-4} on all benchmarks. All methods are trained with a global batch size of 32. The learning rates and training schedules are summarized in Table 9.

For fair comparison, we align the initial capacity of task-wise LoRA experts across methods, as detailed in Table 9. On UCIT, parameter-extension methods maintain six experts with a total LoRA rank of 96, corresponding to rank 16 per expert. On MLLM-DCL, they maintain five experts with a total LoRA rank of 160, corresponding to rank 32 per expert. On MLLM-ACL, they maintain four experts with a total LoRA rank of 128, corresponding to rank 32 per expert. The same capacity settings are used for both the LLaVA and InternVL backbones. Accordingly, single-LoRA baselines use rank 16 on UCIT and rank 32 on MLLM-DCL and MLLM-ACL, while expert-based baselines are allocated the corresponding task-wise capacity. For BRACE, these values define the maximum rank available to each task-wise expert, whereas the actual number of activated Atoms may be smaller because of adaptive capacity allocation. This protocol controls the initial adaptation capacity and enables comparison of how different methods organize, reuse, and select trainable parameters.

D Additional Experiments and Analysis

D.1 Historical Reuse and Adaptive Capacity Allocation

Reuse depth, layer-wise distribution, and task-wise distribution. Figure 8, Figure 9, and Table 10 characterize these three aspects of BRACE, respectively. The selected reuse depth varies across UCIT tasks: ArxivQA, IconQA, and Flickr30k select $k = 8$, VizWiz achieves its largest mean configuration benefit at $k = 4$, and CLEVR-Math selects $k = 2$. Deeper reuse is therefore not uniformly preferred; the selection follows each task's mean validation configuration-benefit curve.

Figure 9(a) reports the selected number of historical Atoms *per module*, rather than contributions from particular historical tasks. Panel (b) reports newly allocated Atoms per module for each task and Transformer layer. The first task has no historical Atoms to reuse and allocates new Atoms

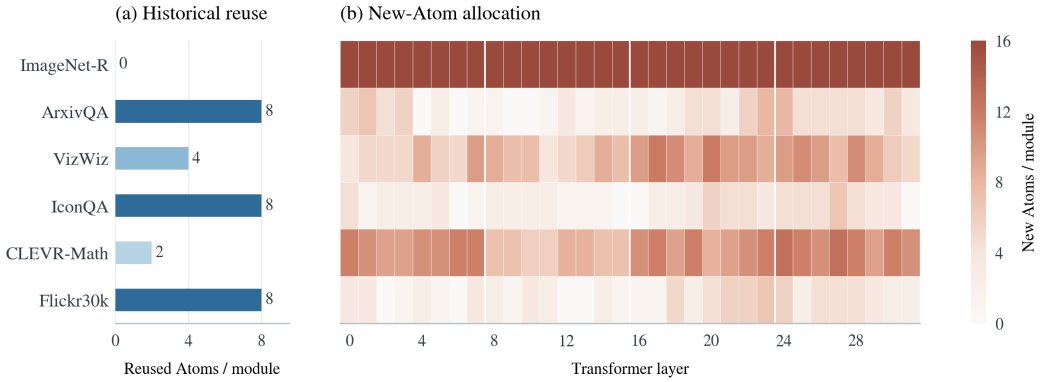


Figure 9: Task- and layer-wise capacity allocation on UCIT. (a) Selected historical Atoms per adapted module for each task; ImageNet-R is the initial task and has no historical reuse. (b) New Atoms per module across Transformer layers (columns) and tasks (rows); darker cells indicate more new Atoms.

throughout the network; subsequent tasks allocate new capacity nonuniformly across layers. At the task level, Table 10 shows that ArxivQA, IconQA, and Flickr30k each reuse 1,792 Atoms and add only 616–752 new Atoms, whereas VizWiz and CLEVR-Math reuse 896 and 448 Atoms and add 1,772 and 2,224, respectively. Thus, less reuse accompanies more new capacity for these UCIT tasks.

On MLLM-DCL, all four subsequent tasks reuse 1,792 Atoms, yet their new allocations range from 1,664 to 4,328. Med and Fin obtain larger validation mean configuration benefits and receive fewer new Atoms than AD and Sci. Equal reuse counts therefore need not imply equal support for the incoming task. All configurations in Table 10 have positive validation mean configuration benefits.

Table 10: Selected reuse configuration and new-Atom allocation by task. Benefit denotes the reduction in validation loss relative to zero reuse; the validation set also determines the selected reuse depth.

Benchmark	Task	Reused	New	Benefit
UCIT	ArxivQA	1,792	752	0.111
	VizWiz	896	1,772	0.079
	IconQA	1,792	616	0.381
	CLEVR-Math	448	2,224	0.013
	Flickr30k	1,792	628	0.721
MLLM-DCL	Med	1,792	3,056	2.984
	AD	1,792	4,328	0.577
	Sci	1,792	4,040	0.273
	Fin	1,792	1,664	2.465

D.2 Capacity Efficiency and Input-Direction Redundancy

We compare three configurations: *w/o AdaAtom & ID*, which uses uniform allocation without Input Diversity; *w/o ID*, which adds adaptive allocation without Input Diversity; and BRACE, which uses both components.

New capacity and cumulative storage. Table 11 reports the newly registered Atoms on UCIT. The total decreases from 15,680 for w/o AdaAtom & ID to 11,144 for w/o ID and 9,576 for BRACE, corresponding to reductions of 28.9% and 38.9% relative to w/o AdaAtom & ID. The task-wise allocations are nonuniform: compared with w/o ID, BRACE allocates fewer Atoms on t_2, t_3, and t_5, slightly more on t_4, and the same number on t_1 and t_6. Thus, its lower cumulative capacity reflects task-dependent allocation rather than uniform compression.

Table 11: Newly registered Atoms by UCIT task for three ablation configurations. ID denotes Input Diversity.

Variant	t_1	t_2	t_3	t_4	t_5	t_6	Total
w/o AdaAtom & ID	3,584	2,688	2,688	1,792	3,136	1,792	15,680
w/o ID	3,584	1,688	2,228	560	2,456	628	11,144
BRACE	3,584	752	1,772	616	2,224	628	9,576

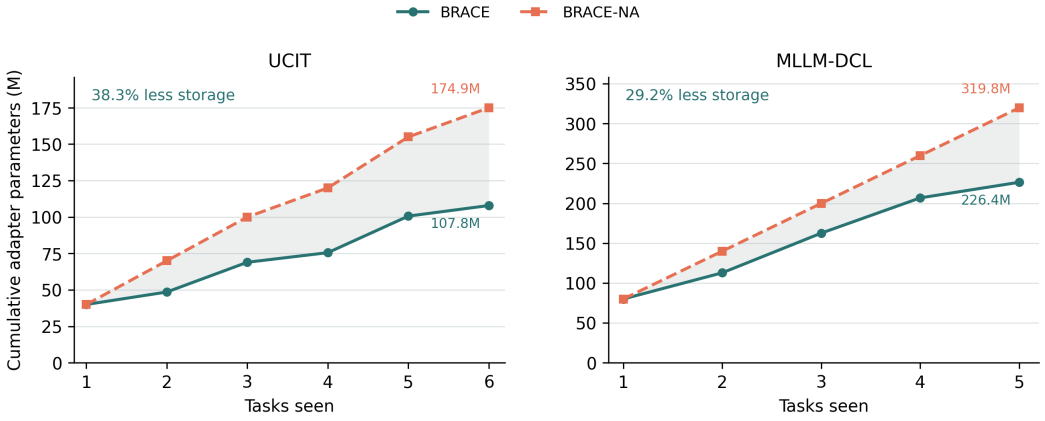


Figure 10: Cumulative unique adapter parameters (millions) after each sequential task for BRACE and BRACE-NA (w/o AdaAtom & ID). UCIT (left) has six tasks; MLLM-DCL (right) has five. The final storage reductions are 38.3% and 29.2%.

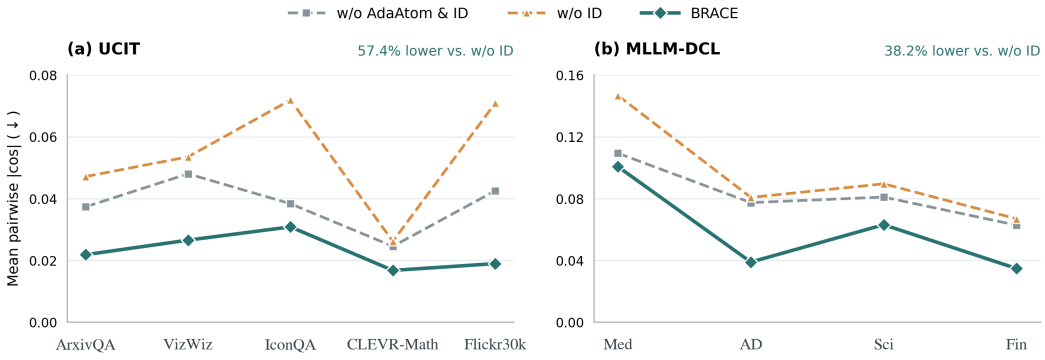


Figure 11: Mean pairwise absolute cosine between newly allocated Atom input directions on (a) UCIT and (b) MLLM-DCL, shown for w/o AdaAtom & ID, w/o ID, and BRACE after the initial task.

Figure 10 shows that the storage gap grows over the task sequence. After six UCIT tasks, BRACE-NA stores 174.9M unique adapter parameters and BRACE stores 107.8M, a 38.3% reduction. After five MLLM-DCL tasks, the corresponding totals are 319.8M and 226.4M, a 29.2% reduction. Historical Atoms are stored once and referenced by multiple task catalogs, avoiding duplicated tensor storage. Atom counts and parameter counts are distinct measures because an Atom's parameter cost depends on its module dimensions.

Directional overlap of new Atoms. Figure 11 and Table 12 report the mean pairwise absolute cosine similarity between new-Atom input directions. On UCIT, the mean overlap is 0.0381 for w/o AdaAtom & ID, 0.0539 for w/o ID, and 0.0229 for BRACE. The corresponding values on MLLM-

DCL are 0.0825, 0.0959, and 0.0592. BRACE has the lowest overlap on every reported subsequent task in both benchmarks.

Compared with w/o AdaAtom & ID, w/o ID uses fewer new Atoms but exhibits higher mean input-direction overlap on both benchmarks. BRACE achieves lower overlap than w/o ID under adaptive allocation. Together, these results show that BRACE combines lower cumulative capacity with more diverse new-Atom input directions. This analysis complements the predictive-performance comparison in Table 4.

Table 12: New-Atom input-direction redundancy for three ablation configurations, measured by the mean pairwise absolute cosine. The initial tasks have no historical reuse and are omitted. Lower values indicate less directional overlap.

Benchmark	Variant	t_2	t_3	t_4	t_5	t_6	Mean
UCIT	w/o AdaAtom & ID	0.0373	0.0480	0.0383	0.0245	0.0425	0.0381
	w/o ID	0.0471	0.0535	0.0719	0.0261	0.0709	0.0539
	BRACE	0.0218	0.0265	0.0308	0.0167	0.0189	0.0229
MLLM-DCL	w/o AdaAtom & ID	0.1093	0.0773	0.0809	0.0627	–	0.0825
	w/o ID	0.1467	0.0807	0.0895	0.0668	–	0.0959
	BRACE	0.1006	0.0388	0.0630	0.0346	–	0.0592

D.3 Evaluation Across Backbones and Capabilities

InternVL on MLLM-DCL. Table 13 extends the comparison to InternVL. DGG uses its frozen-projector rerun; BWT for the completed DCL baseline matrices is recomputed from unrounded results using the $T-1$ denominator in Appendix C.3. BRACE achieves MFN 68.86, exceeding MoE-CL, PCLR, and DGG by 1.84, 6.73, and 13.75 points, respectively. Its BWT of -1.12 is also closer to zero than those of these baselines.

Table 13: InternVL on MLLM-DCL. Task columns give final-stage performance, MFT, MFN, MAA, and BWT use the definitions in Appendix C.3.

Method	Venue	RS	Med	AD	Sci	Fin	**MFT** (↑)	**MFN** (↑)	**MAA** (↑)	**BWT** (↑)
Zero-shot	–	31.16	29.81	14.06	33.93	64.32	34.66	–	–	–
Individual	–	81.49	66.42	54.56	54.48	91.24	69.64	–	–	–
LoRA-FT	ICLR-22	69.93	52.17	33.04	42.67	91.07	69.06	57.78	65.22	-14.11
O-LoRA	EMNLP-23	74.48	54.16	39.60	48.30	88.54	65.51	61.02	65.83	-5.62
MoELoRA	NeurIPS-24	69.90	52.08	33.17	42.19	90.58	68.83	57.58	65.97	-14.06
ModalPrompt	EMNLP-25	50.20	30.95	14.11	33.91	65.45	38.93	38.92	38.70	**-0.01**
CL-MoE	CVPR-25	78.12	52.51	35.53	42.69	91.24	69.22	60.02	67.60	-11.51
HiDE	ACL-25	75.40	57.66	36.73	41.48	88.59	65.26	59.97	65.94	-6.61
SEFE	ICML-25	78.21	57.59	51.45	44.65	91.37	69.55	64.65	68.84	-6.12
DISCO	ICCV-25	77.90	47.50	49.13	49.37	90.92	68.55	62.96	67.81	-6.98
DGG	CVPR-26	79.47	36.92	37.42	38.12	83.63	58.67	55.11	60.13	-4.45
MoE-CL	WWW-26	79.21	60.97	53.85	49.49	91.60	69.34	67.02	**70.08**	-2.90
PCLR	ICLR-26	79.67	50.99	42.27	46.23	91.49	68.47	62.13	68.27	-7.92
BRACE-NA	–	**80.79**	**64.81**	**54.30**	**53.70**	**91.62**	**69.86**	**69.04**	68.87	-1.02
BRACE	–	**80.79**	64.38	54.26	53.44	91.42	69.79	68.86	68.72	-1.12

InternVL on UCIT. Table 14 shows that BRACE obtains MFN 74.70 and MAA 84.29, compared with MoE-CL's 72.97 and 83.36. BRACE also achieves the highest reported ArxivQA and VizWiz scores, 95.37 and 60.03, without leading on every task. On both InternVL benchmarks, BRACE-NA slightly outperforms BRACE in MFN: 69.04 versus 68.86 on MLLM-DCL and 75.02 versus 74.70 on UCIT. It also obtains slightly higher MAA on both.

Extension to MLLM-ACL. MLLM-ACL (Guo et al., 2025d) follows the sequence OCR → Math → VP → APP, covering image text recognition, visual mathematical reasoning, visual perception, and GUI navigation. Table 15 reports the LLaVA-1.5 results, with task-ID-free routing for BRACE-NA and BRACE.

Table 14: InternVL on UCIT. Task columns give final-stage performance.

Method	Venue	ImgNet-R	ArxivQA	VizWiz	IconQA	CLEVR	Flickr30k	**MFT** (↑)	**MFN** (↑)	**MAA** (↑)	**BWT** (↑)
Zero-shot	–	21.10	63.20	40.59	24.70	21.20	44.67	35.91	–	–	–
Individual	–	91.67	90.83	57.87	78.43	76.63	61.72	76.19	–	–	–
LoRA-FT	ICLR-22	76.50	78.60	45.57	71.13	61.63	**58.82**	79.20	65.38	80.66	-16.59
O-LoRA	EMNLP-23	90.30	93.53	46.26	68.40	65.17	56.06	70.97	69.95	81.23	-1.22
MoELoRA	NeurIPS-24	68.57	80.47	45.43	69.23	65.93	58.57	**79.25**	64.70	80.07	-17.46
ModalPrompt	EMNLP-25	57.10	63.90	42.11	37.40	47.50	45.76	48.98	48.96	53.52	-0.02
CL-MoE	CVPR-25	74.23	78.77	44.77	53.90	72.73	58.23	71.39	63.77	77.44	-9.15
HiDE	ACL-25	90.00	93.77	50.49	71.03	61.10	55.08	76.29	70.25	82.97	-7.25
SEFE	ICML-25	85.93	76.90	47.42	69.77	63.37	58.19	73.55	66.93	79.44	-7.94
DISCO	ICCV-25	92.13	94.13	48.11	**73.90**	67.53	58.06	78.92	72.31	84.24	-7.93
DGG	CVPR-26	**93.20**	87.57	43.19	49.60	43.67	53.39	63.05	61.77	75.24	-1.54
MoE-CL	WWW-26	91.27	94.53	54.75	69.30	70.60	57.35	75.40	72.97	83.36	-2.92
PCLR	ICLR-26	74.97	94.27	45.19	56.00	**75.47**	56.17	66.29	67.01	76.94	**0.87**
BRACE-NA	–	92.73	95.30	59.23	69.70	74.67	58.50	75.31	**75.02**	**84.42**	-0.34
BRACE	–	92.60	**95.37**	**60.03**	69.30	72.43	58.45	75.38	74.70	84.29	-0.82

Within the matched BRACE-NA–BRACE comparison, MFN increases from 41.23 to 41.55 and BWT improves from -0.54 to -0.02. APP agreement increases by 1.50 points, OCR by 0.10 points, VP is unchanged, and Math decreases by 0.30 points. MFT and MAA also decrease slightly, from 41.63 to 41.57 and from 37.94 to 37.73.

Table 15: MLLM-ACL comparison on LLaVA-1.5.

Method	Venue	OCR	Math	VP	APP	**MFT** (↑)	**MFN** (↑)	**MAA** (↑)	**BWT** (↑)
Zero-shot	–	31.20	30.20	60.79	10.00	33.05	–	–	–
Individual	–	33.60	36.50	65.10	32.50	41.93	–	–	–
LoRA-FT	ICLR-22	23.60	33.70	55.84	32.50	41.28	36.41	36.58	-6.49
O-LoRA	EMNLP-23	29.60	32.90	52.41	33.75	39.72	37.17	35.42	-3.41
MoELoRA	NeurIPS-24	26.70	32.80	56.85	27.22	39.45	35.89	36.07	-4.75
ModalPrompt	EMNLP-25	31.80	32.50	60.53	10.00	33.61	33.71	34.16	**0.13**
CL-MoE	CVPR-25	19.90	32.70	53.43	30.69	40.50	34.18	35.65	-8.43
HiDE	ACL-25	24.60	32.10	46.32	28.75	37.98	32.94	34.60	-6.72
SEFE	ICML-25	26.00	33.40	57.74	33.75	40.99	37.72	36.59	-4.35
DISCO	ICCV-25	**32.90**	33.10	60.15	30.14	39.02	39.07	36.57	0.07
BRACE-NA	–	32.60	**35.00**	**61.93**	35.38	**41.63**	41.23	**37.94**	-0.54
BRACE	–	32.70	34.70	**61.93**	**36.88**	41.57	**41.55**	37.73	-0.02

Table 16: Available MLLM-ACL baseline scores on InternVL.

Method	Venue	OCR	Math	VP	APP	**MFT** (↑)	**MFN** (↑)	**MAA** (↑)	**BWT** (↑)
Zero-shot	–	30.00	31.20	56.09	2.50	29.95	–	–	–
Individual	–	32.20	33.40	67.77	33.75	41.78	–	–	–
LoRA-FT	ICLR-22	21.40	32.80	60.28	29.86	40.85	36.09	36.38	-6.35
O-LoRA	EMNLP-23	25.50	32.30	64.59	24.44	38.64	36.71	36.05	-2.57
MoELoRA	NeurIPS-24	17.20	32.70	55.33	32.50	41.41	34.43	35.36	-9.30
ModalPrompt	EMNLP-25	25.90	31.60	55.58	11.39	32.55	31.12	31.44	-1.90
CL-MoE	CVPR-25	25.10	31.90	60.91	27.50	40.15	36.35	35.75	-5.06
HiDE	ACL-25	17.80	33.00	41.12	33.33	40.28	31.31	34.12	-11.96
SEFE	ICML-25	19.00	32.20	62.18	31.94	41.45	36.33	36.29	-6.83
DISCO	ICCV-25	30.60	33.10	**65.36**	27.50	39.21	39.14	36.73	**-0.10**
BRACE-NA	–	**31.80**	36.90	64.34	32.75	41.95	41.45	38.36	-0.67
BRACE	–	31.20	**37.90**	63.83	**36.13**	**42.37**	**42.26**	**38.38**	-0.15

D.4 Qualitative Analysis

Figures 12 and 13 compare final-stage outputs from BRACE, DGG, and PCLR. These selected examples illustrate specific response differences rather than estimate error frequencies.

UCIT. In Figure 12, BRACE matches the ground-truth labels, choices, and counts, and preserves the central semantics of the open-ended captions. DGG and PCLR instead confuse a tiger with a lion, select an incorrect science answer, describe a black shirt as blue, and return incorrect counts for IconQA and CLEVR-Math. On Flickr30k, BRACE captures a couple getting married, whereas

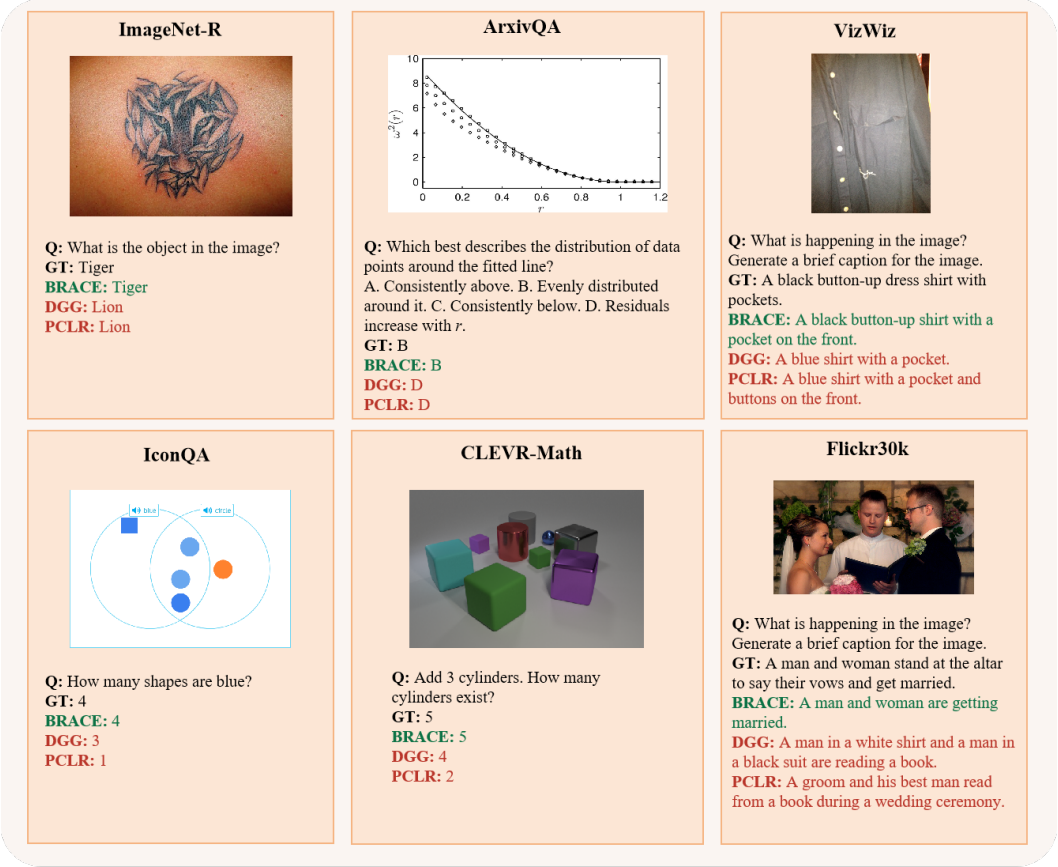


Figure 12: UCIT examples at the final evaluation stage, showing the ground truth (GT) and outputs from BRACE, DGG, and PCLR. The six tasks test object identity, a scientific multiple-choice answer, a clothing attribute, visual counting, arithmetic counting, and event-level image captioning.

the other methods describe two men reading from a book. These contrasts concern visual categories, attributes, quantities, and events, illustrating that fluent answers can still omit the evidence required by the question.

MLLM-DCL. Figure 13 covers remote-sensing counting, pathology recognition, vehicle-attribute perception, scientific-diagram reasoning, and financial-chart interpretation. BRACE matches the ground truth in all displayed cases. DGG and PCLR overcount buildings, misidentify the esophagus, fail to recover the target vehicle attribute, and select incorrect science and finance options. These examples complement the aggregate results by showing the specific domain information recovered or missed by each method.

D.5 Robustness to Optimization and Task Order

Variation across seeds. Table 17 reports four seeds at the selected rank and learning rate for each benchmark. UCIT MFN ranges from 73.96 to 74.36 , and MLLM-DCL MFN ranges from 70.15 to 70.31. The corresponding MAA ranges are 82.30–82.63 and 71.50–71.62. These runs show limited variation in the reported aggregate scores, while UCIT BWT changes sign across seeds.

Task-order sensitivity. Table 18 compares forward and reverse task sequences under the same rank and learning rate within each benchmark. Reversing UCIT reduces MFN from 74.34 to 73.30 and MAA from 82.62 to 65.36. On MLLM-DCL, the reverse order produces a similar MFN (70.16 versus 70.31) but a higher MAA (74.10 versus 71.52). BWT moves closer to zero under the reverse

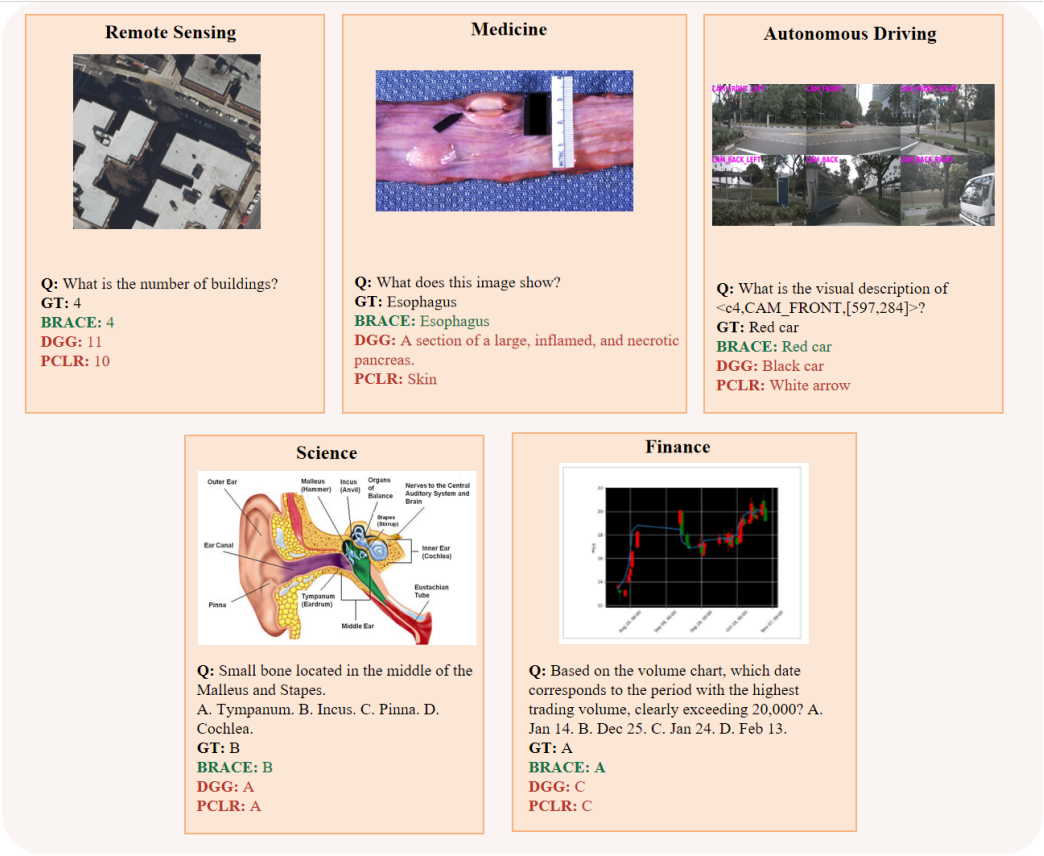


Figure 13: MLLM-DCL examples at the final evaluation stage, showing the ground truth (GT) and outputs from BRACE, DGG, and PCLR. The five tasks assess building counts, anatomical identification, vehicle color, scientific-diagram reasoning, and financial-chart interpretation.

Table 17: Multi-seed results of BRACE. UCIT uses rank 16 and MLLM-DCL uses rank 32; UCIT uses a learning rate of 2×10^{-4} and MLLM-DCL uses a learning rate of 2×10^{-5}.

Benchmark	Seed	MFT (↑)	MFN (↑)	MAA (↑)	BWT (↑)
UCIT	42	**74.40**	**74.36**	**82.63**	-0.053
	43	74.23	74.35	82.30	**+0.139**
	44	74.24	74.33	82.62	+0.111
	45	73.96	73.96	82.49	+0.002
MLLM-DCL	42	**70.37**	**70.31**	71.52	-0.076
	43	70.25	70.23	71.60	**-0.023**
	44	70.19	70.15	71.50	-0.043
	45	70.33	**70.31**	**71.62**	-0.031

order on both benchmarks. Across these two orderings, UCIT shows a marked change in MAA, whereas MLLM-DCL maintains similar final MFN.

E Theoretical Analysis

First-order removal Benefit. Fix an input x, the model parameters, and all candidate gates except the gate of Atom u. Let $f(g) = \mathcal{L}(x; g)$ for $g \in [0, 1]$, and assume that f is twice continuously differentiable on $[0, 1]$. The exact removal benefit and its first-order probe estimate are defined as

$$\beta_u(x) = f(0) - f(1), \qquad \widetilde{\beta}_u(x) = -f'(1). \tag{32}$$

Table 18: Sensitivity to task order. Forward and reverse runs use the same rank and learning rate within each benchmark.

Benchmark	Order	MFT (↑)	MFN (↑)	MAA (↑)	BWT (↑)
UCIT	Forward	74.39	74.34	82.62	−0.063
UCIT	Reverse	73.31	73.30	65.36	−0.016
MLLM-DCL	Forward	70.37	70.31	71.52	−0.076
MLLM-DCL	Reverse	70.17	70.16	74.10	−0.021

By the fundamental theorem of calculus,

$$\begin{aligned}\beta_u(x) &= -\int_0^1 f'(t)\,dt \\ &= -f'(1) + \int_0^1 t f''(t)\,dt.\end{aligned} \tag{33}$$

Since f'' is continuous, the weighted mean-value theorem gives some $\xi \in (0,1)$ such that

$$\beta_u(x) = -f'(1) + \frac{1}{2}f''(\xi) = \widetilde{\beta}_u(x) + \frac{1}{2}f''(\xi). \tag{34}$$

Therefore, if

$$|f''(g)| \le H_u(x), \qquad g \in [0,1],$$

then

$$\left|\beta_u(x) - \widetilde{\beta}_u(x)\right| \le \frac{H_u(x)}{2}.$$

Thus, the gate derivative provides a first-order estimate of the single-Atom removal benefit, with an error determined by the curvature along the gate-removal path.

Joint effects of multiple Atoms. Let U be a set of Atoms whose gates are being analyzed, and let $F_x(\mathbf{g})$ denote the loss when their gates vary according to $\mathbf{g}$, while all other model components are fixed at a reference configuration. Assume that F_x is twice continuously differentiable on the corresponding gate domain. For any subset $A \subseteq U$, let $\mathbf{e}_A$ denote the gate vector that activates exactly the Atoms in A, and let $\mathbf{e}_\varnothing = \mathbf{0}$.

Consider an ordered set

$$S = \{u_1, \dots, u_M\}, \qquad S_j = \{u_1, \dots, u_j\}, \qquad S_0 = \varnothing.$$

The individual addition effect of Atom u and the complete configuration benefit are defined as

$$\Delta_x(u) = F_x(\mathbf{e}_{\{u\}}) - F_x(\mathbf{0}), \qquad B_x(S) = F_x(\mathbf{0}) - F_x(\mathbf{e}_S). \tag{35}$$

The joint loss change admits the exact decomposition

$$F_x(\mathbf{e}_S) - F_x(\mathbf{0}) = \sum_{u\in S} \Delta_x(u) + I_x(S), \tag{36}$$

where the interaction remainder is

$$I_x(S) = \sum_{j=2}^{M} \left[F_x(\mathbf{e}_{S_j}) - F_x(\mathbf{e}_{S_{j-1}}) - F_x(\mathbf{e}_{\{u_j\}}) + F_x(\mathbf{0})\right]. \tag{37}$$

Equivalently, the complete configuration benefit satisfies

$$B_x(S) = -\sum_{u\in S} \Delta_x(u) - I_x(S). \tag{38}$$

For each $j \ge 2$, define

$$G_j(t,s) = F_x\big(t\mathbf{e}_{\{u_j\}} + s\mathbf{e}_{S_{j-1}}\big), \qquad (t,s) \in [0,1]^2.$$

Applying the fundamental theorem of calculus twice gives

$$\begin{aligned} & F_x(\mathbf{e}_{S_j}) - F_x(\mathbf{e}_{S_{j-1}}) - F_x(\mathbf{e}_{\{u_j\}}) + F_x(\mathbf{0}) \\ &= \int_0^1 \int_0^1 \frac{\partial^2 G_j(t,s)}{\partial t\, \partial s}\, ds\, dt \\ &= \sum_{v \in S_{j-1}} \int_0^1 \int_0^1 \frac{\partial^2 F_x}{\partial g_{u_j}\, \partial g_v} \left(t\mathbf{e}_{\{u_j\}} + s\mathbf{e}_{S_{j-1}} \right)\, ds\, dt. \end{aligned} \tag{39}$$

Substituting Eq. 39 into Eq. 37 expresses the complete configuration effect through cross-Atom derivatives. When U contains Atoms from multiple adapted modules, these mixed derivatives also include cross-module interactions.

The single-Atom probe therefore provides a first-order estimate of the removal benefit for candidate ranking, whereas the complete configuration benefit additionally depends on the joint interaction remainder. FixedAtom accordingly uses the probe scores to construct candidate top-k sets and evaluates each assembled multi-layer configuration on held-out examples for reuse-depth selection.

F Limitations and Future Directions

Despite the favorable performance storage trade-offs observed on the evaluated benchmarks, BRACE has several limitations. First, benefit-aware selection relies on a finite set of current-task probes. Although separate subsets are used for ranking and configuration validation, these examples may not fully represent tasks with more diverse input distributions. Second, the candidate pool can expand over longer task streams, increasing the cost of evaluating historical Atoms even though shared Atoms are stored only once. The present evaluation also focuses on finite task sequences, leaving performance and efficiency over substantially longer streams to be investigated.

Future work could improve the coverage and efficiency of benefit estimation through more representative probing and indexed candidate retrieval. It would also be valuable to study longer task streams and additional model backbones, including how task-free routing and adaptive capacity allocation behave as the Atom pool expands. More broadly, coordinating fine-grained knowledge reuse with selective capacity growth remains a promising direction for multimodal continual learning.