Two Drifts, One Principle: Conflict-Aware Spectral Consolidation for Multimodal Continual Learning

Haiyi Zhang[1,*] **Qianyi Cai**[1,*,†] **Hanqing Wang**[1] **Zilin Wang**[3]
Xinyue Di[1,4] **Yijie Xu**[1] **Yili Wang**[1] **Tianfu Wang**[1]
Yifan Han[5] **Hui Xiong**[1,2,‡]
[1]Thrust of Artificial Intelligence, HKUST (Guangzhou), China
[2]Department of Computer Science and Engineering, HKUST, Hong Kong SAR, China
[3]LARK, HKUST (Guangzhou), China
[4]School of Electronic and Information Engineering, South China University of Technology
[5]School of Artificial Intelligence, Shanghai Jiao Tong University
[*]Equal contribution [†]Project leader [‡]Corresponding author
{hzhang152,qcai603}@connect.hkust-gz.edu.cn
xionghui@ust.hk

Abstract

Multimodal continual learning requires MLLMs to acquire new domains and abilities sequentially while retaining previous capabilities. MoE-based methods preserve task-specific knowledge, but inference relies on reliable routing. We instead study continual task-boundary LoRA–projector consolidation, which forms one shared model from sequential task updates under fixed module capacities. LoRA merging naturally supports language-side consolidation, but extending it to MLLMs raises two challenges. First, sequential LoRA updates may interfere and overwrite directions important to earlier tasks. Second, projector mismatch may make the consolidated LoRA update incompatible with the final visual-language alignment. To address these challenges, we propose **CASC** (**C**onflict-**A**ware **S**pectral **C**onsolidation), which jointly consolidates LoRA and projector updates. CASC maintains fixed-rank spectral banks and uses a shared protected subspace memory rule to preserve dominant historical directions while incorporating compatible new updates under fixed rank budgets, yielding a single model without replay data or inference time routing. For specified protected subspaces, we derive the optimal compatible residual under the Frobenius reconstruction objective. On MLLM-CL, CASC achieves the highest MFN and MAA on DCL and ACL, and the highest BWT on DCL among the compared methods. The code of CASC is available at https://github.com/H-Workspace/CASC.

1 Introduction

Continual learning for multimodal large language models (MLLMs) aims to adapt a model to a sequence of new domains and abilities while retaining previously acquired capabilities. Classical continual learning methods mitigate forgetting through replay or regularization, but replay assumes access to historical data and regularization can constrain later adaptation [1, 2]. Recent continual learning methods often reduce interference by isolating task knowledge in adapters, expandable modules, expert pools, or routing mechanisms [3, 4, 5, 6]. As illustrated in Fig. 1(a), MoE-based continual learning methods maintain task-specific LoRA experts and select an expert for each inference input.

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

This preserves task-specific capacity, but also makes inference depend on reliable component selection. When task identity is unavailable, ambiguous, or mixed across domains, this selection becomes an additional source of uncertainty. As more tasks are learned, the number of stored components or routing candidates may also increase [7, 8]. This motivates our central research question: *Can sequential LoRA updates and projector shifts be consolidated at each task boundary into one shared model with fixed capacity and no inference routing?*

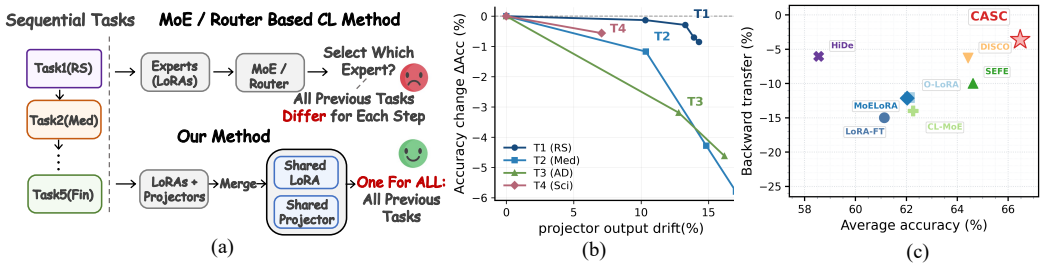


Figure 1: Motivation for unified continual multimodal consolidation. (a) MoE/Router-based methods keep task-specific LoRA experts, whereas our method consolidates them into one shared LoRA update and one consolidated projector. (b) Projector output drift correlates with historical-task accuracy drop. (c) CASC improves average performance and backward transfer.

A common strategy is to train parameter-efficient adapters and then combine the resulting adapters [9, 10]. LoRA is suitable for this strategy because it represents each task adaptation as an explicit low-rank update to the language-model weights [11]. Thus, multiple task-specific LoRA adapters can be consolidated into one shared language-side update [12]. However, extending language-based LoRA merging to continual multimodal consolidation raises **two key challenges: (1) Sequential LoRA interference.** Different tasks can induce low-rank updates that overlap or conflict in parameter space, and naive merging or compression may overwrite dominant historical update directions, a challenge also observed in model merging [13]. In continual learning, this interference is harder to control because updates arrive sequentially, previous task data are unavailable, and the consolidated adapter must remain fixed-size. As a result, later LoRA updates can overwrite directions learned from earlier tasks. **(2) Projector compatibility with consolidated LoRA.** In MLLMs, the consolidated LoRA update operates on visual embeddings produced by the multimodal projector, which maps visual features into the language embedding space [14]. Because this projector is also updated during sequential multimodal tuning, LoRA consolidation alone does not determine the final adapted model. A post-hoc projector choice, such as using the last-task, base, or averaged projector, can introduce different alignment biases. Fig. 1(b) further shows that projector output drift correlates with historical-task accuracy drop, indicating that projector-side consolidation should be handled together with LoRA consolidation. Thus, even if historical language-side update directions are partially preserved, previous multimodal abilities may still degrade when visual evidence is mapped through an incompatible projector. *Therefore, unified continual multimodal consolidation requires joint treatment of low-rank language-side updates and dense projector shifts.*

We propose **C**onflict-**A**ware **S**pectral **C**onsolidation (**CASC**), a recurrent rule for continual task-boundary LoRA–projector consolidation under fixed module capacities. CASC represents each task-induced LoRA update and projector shift in spectral form and stores their accumulated updates in fixed-rank module spectral banks. During consolidation, CASC uses spectral energy as a data-free proxy to protect dominant historical update directions, filters incoming updates against protected input and output subspaces, and recompresses the compatible residual under the module budget. LoRA and projector modules are consolidated separately but governed by the same memory rule, yielding one shared LoRA update and one consolidated projector without replay during consolidation or input-dependent routing at inference. Here, unified consolidation means applying this rule to both module types, not combining their parameters into a common bank or assuming identical functional roles. As shown in Fig. 1(c), CASC improves average accuracy and mitigates forgetting. Our contributions are summarized as follows:

- We introduce recurrent, fixed-capacity consolidation of completed LoRA updates and projector shifts. CASC protects historical input and output update subspaces and reconstructs one adapted MLLM without a growing expert set or inference routing.

- For specified protected subspaces and rank budgets, we derive the closed-form solution to the shared Frobenius reconstruction objective and establish a rank bound and two-sided orthogonality. This establishes one consolidation rule for separate LoRA and projector banks.
- On MLLM-CL, CASC achieves the highest MFN and MAA on both DCL and ACL, and the highest BWT on DCL among the compared methods. Component ablations show the benefit of consolidating both LoRA and projector updates.

2 Related Work

2.1 Continual Learning

Recent continual learning (CL) methods are typically categorized as replay-based [15, 16, 17, 18], regularization-based [19, 20, 21, 22], or architecture-based [23, 24, 25, 26, 27, 28]. For LLMs, continual learning mainly studies continual pre-training [29, 30, 31, 32] and continual instruction tuning [33, 34, 35], often using parameter-efficient tuning such as LoRA [11, 36] to reduce training cost. Continual learning for MLLMs is harder because it must preserve both language reasoning and cross-modal alignment. Recent works have studied multimodal continual instruction tuning, capability continual learning, and federated continual instruction tuning: CoIN [37] proposes MoELoRA for expert-based instruction alignment; MLLM-CL [3] introduces MR-LoRA with parameter isolation and routing; and FCIT [38] addresses data heterogeneity through dynamic knowledge organization. Other methods further explore expert routing, hierarchical expansion, and forgetting decomposition, including CL-MoE [39], HiDe [40], SEFE [41], and SAME [42]. Many of these methods retain task-specific components or routing mechanisms at inference, whereas CASC consolidates sequential LoRA updates and projector shifts into one adapted model.

Subspace protection. GPM [43] constrains training gradients using historical subspaces obtained from activation matrices. InfLoRA [44] designs the trainable low-rank subspace to reduce interference with earlier tasks, while O-LoRA [2] encourages orthogonality between successive LoRA subspaces. These methods constrain task optimization. CASC instead acts on completed task updates at each task boundary, protecting input and output directions of the accumulated update before recompressing the compatible residual. Its contribution lies in this recurrent, fixed-capacity rule applied to separate LoRA and projector banks, rather than in SVD or orthogonal projection alone.

2.2 Multimodal Large Language Models

MLLMs [45, 14, 46, 47, 48, 49, 50] extend LLMs to multimodal inputs, enabling perception, understanding, and reasoning over visual and textual information. Sequential adaptation can degrade both language reasoning and cross-modal alignment. Unlike text-only LLMs [51, 52, 53, 54], MLLMs rely on a multimodal projector that maps visual features into the language embedding space [14]. Thus, continual consolidation should account for both language-side adaptation and projector-side changes. Existing MLLM continual learning methods often adopt parameter-efficient adaptation such as LoRA [11, 3, 38]. Our work jointly consolidates LoRA updates and projector changes to preserve cross-modal compatibility across tasks.

2.3 Model Merging

Model merging combines separately fine-tuned models into one set of weights, usually without retraining. Mode connectivity [55] shows that independently trained networks can often be connected by low-loss paths, which supports simple weight-space interpolation. Model Soups [56] further demonstrates that averaging fine-tuned models from a shared basin can improve accuracy and robustness without increasing inference cost. Git Re-Basin [57] highlights that permutation symmetries in neural networks can obstruct direct weight averaging and proposes to align models before merging. TIES-Merging [13] identifies redundant parameter changes and sign conflicts as major sources of interference, and mitigates them through trimming and sign-aware merging. DARE [58] sparsifies task deltas through random dropping and rescaling before merging. RobustMerge [12] studies parameter-efficient MLLM merging through direction robustness, parameter pruning, and coefficient adjustment. Sequential merging is also studied by OPCM [59], which combines orthogonal projection and adaptive scaling with memory independent of the number of merged models. CASC focuses on continual LoRA–projector consolidation: it consolidates separate language-side updates

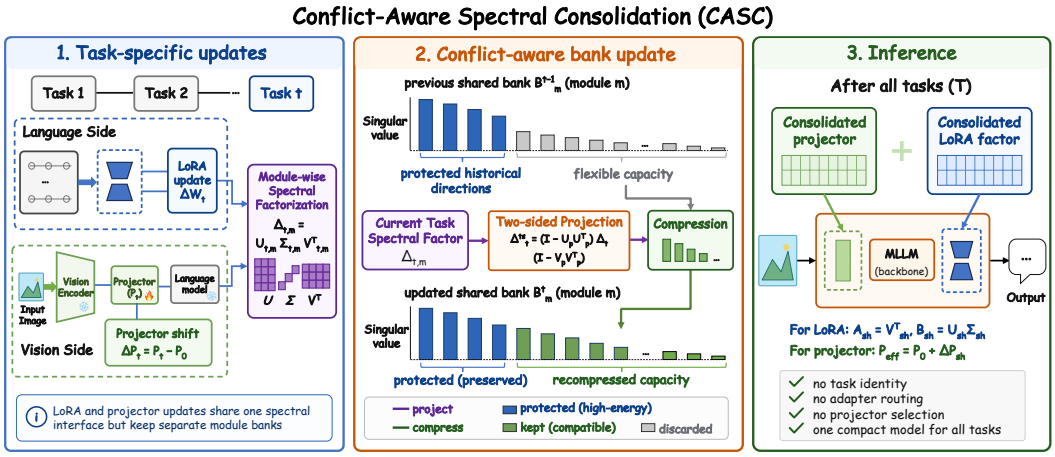


Figure 2: Overview of the proposed CASC framework.

and projector shifts through protected input and output subspaces, with an explicit rank budget and a shared constrained reconstruction objective. Our RobustMerge comparison uses a cumulative adaptation of its static merging procedure, as detailed in Appendix 9.

3 Methodology

3.1 Overview

Unified multimodal consolidation aims to obtain one adapted MLLM from a continual stream of LoRA updates and projector-side shifts, without replay data, task identity, or input-dependent routing. CASC addresses this problem with streaming module-wise spectral banks. At each task boundary, CASC consolidates the current update with the accumulated historical update under a fixed rank budget. LoRA modules and projector parameters keep separate banks, but are updated under a shared spectral memory rule. As illustrated in Fig. 2, this recurrent consolidation produces one consolidated LoRA adapter and one consolidated projector for use across all tasks.

3.2 Preliminaries

Problem Setup. We consider a sequence of multimodal instruction tuning tasks $\{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_T\}$ under the standard continual learning setting [1, 2]. At stage t, only the current dataset $\mathcal{D}_t$ is available for adaptation. After all tasks are processed, the model is evaluated on the whole task sequence without task identity. The goal is to obtain one adapted MLLM for inference across all tasks by recurrently consolidating each new task update into the previous consolidated model.

Language-side LoRA Update. For a frozen pretrained weight matrix $W_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, LoRA [11] represents the trainable update as

$$W = W_0 + \Delta W, \qquad \Delta W = \frac{\alpha}{r} BA, \tag{1}$$

where $A \in \mathbb{R}^{r \times d_{\text{in}}}$ and $B \in \mathbb{R}^{d_{\text{out}} \times r}$ are low-rank factors, r is the adapter rank, and α controls the update scale. For task t and module m, we denote the effective LoRA update to be inserted at the current consolidation step by $\Delta W_{t,m}$. CASC consolidates this update with the previous bank $\Delta^{t-1}_{\text{sh},m}$, rather than storing all earlier task-specific LoRA adapters. Naive merging or recompression can introduce interference when different tasks use overlapping low-rank directions [13].

Projector-side Update. In LLaVA-style MLLMs [14], the multimodal projector maps visual features into the language embedding space. This projector defines the visual-language interface on which the LoRA-updated language model operates. Since the projector is also updated during multimodal instruction tuning, a consolidated LoRA adapter must be paired with a compatible projector to preserve the learned cross-modal alignment. For each trainable projector parameter p, let $P_{t,p}$ denote

the projector parameter obtained from the current task adaptation, and let $P_{0,p}$ be the corresponding base projector parameter. We define the current projector-side shift in a common base reference frame as

$$\Delta P_{t,p} = P_{t,p} - P_{0,p}. \tag{2}$$

At step t, this shift is inserted into the previous projector bank $\Delta_{\mathrm{sh},p}^{t-1}$. The recurrent bank update keeps historical projector information in the consolidated bank, without storing a growing set of previous task-specific projectors.

Recurrent Spectral Bank. A spectral bank stores a rank-bounded factorization of one module's accumulated update. For an adapted module m, let $\Delta_{t,m}$ denote its current LoRA update or projector shift, and let $\Delta_{\mathrm{sh},m}^{t-1}$ denote the accumulated update represented by its previous spectral bank. CASC produces the updated bank through

$$\Delta_{\mathrm{sh},m}^{t} = \mathrm{Consolidate}\big(\Delta_{\mathrm{sh},m}^{t-1}, \Delta_{t,m}\big). \tag{3}$$

3.3 Unified Spectral Interface for LoRA and Projector Updates

CASC maps LoRA updates and projector shifts to a common spectral interface through their module-wise matrix representations, despite their distinct functional roles. For a LoRA module, the matrix is the effective update $\Delta W_{t,m} = \frac{\alpha}{r} B_{t,m} A_{t,m}$ from Eq. (1). For a projector parameter, the matrix is the current dense shift $\Delta P_{t,m} = P_{t,m} - P_{0,m}$ from Eq. (2), which is inserted into the previous projector bank. We denote either case by $\Delta_{t,m}$ and write its spectral form as

$$\Delta_{t,m} = U_{t,m} \Sigma_{t,m} V_{t,m}^{\top}, \tag{4}$$

where m indexes a specific LoRA module or projector parameter block. For LoRA, retaining all nonzero singular components preserves the trained low-rank update exactly, while exposing orthogonal input and output directions that are not explicit in the raw factors $A_{t,m}$ and $B_{t,m}$. For the projector, Eq. (4) is an exact decomposition before bank compression; the projector rank budget is imposed only during consolidation. Each module retains its own spectral bank. The formal justification for applying the same spectral rule after this conversion is given in Eq. (10) and Appendix 6.

3.4 Streaming Module-wise Spectral Banks

CASC stores historical information as module-wise spectral banks. Let $\mathcal{M}$ be the set of adapted modules, including language-side LoRA modules and projector parameters. After tasks $1, \ldots, t-1$ have been processed, the banks are

$$\mathcal{B}^{t-1} = \{\Delta_{\mathrm{sh},m}^{t-1}\}_{m \in \mathcal{M}}, \tag{5}$$

where $\Delta_{\mathrm{sh},m}^{t-1}$ is the accumulated update for module m. LoRA modules use a bank rank budget R, while projector parameters use a separate budget R_P. We write the corresponding module budget as R_m.

After adapting task t, we extract the active module updates

$$\mathcal{U}_t = \{\Delta_{t,m}\}_{m \in \mathcal{M}_t}, \tag{6}$$

where $\mathcal{M}_t \subseteq \mathcal{M}$ contains the modules updated by the current task. CASC updates the corresponding banks for $m \in \mathcal{M}_t$ according to Eq. (3). This is a recurrent consolidation rule: the bank after task $t-1$ is treated as the historical model, and the current task update is inserted into it. Modules absent from the current task keep their previous banks unchanged. Therefore, CASC stores historical information only through $\mathcal{B}^{t-1}$ and does not keep a growing set of task-specific experts, routes, or previous data.

3.5 Conflict-Aware Bank Update

At each recurrent consolidation step, the incoming update may overlap with leading spectral components of the existing bank. Directly adding and recompressing the current update with the previous

bank can rotate or suppress previously retained historical components. CASC avoids this by protecting leading historical directions before residual insertion and recompression. For module m, suppose the previous bank is

$$\Delta_{\mathrm{sh},m}^{t-1} = U_{\mathrm{sh},m}^{t-1}\Sigma_{\mathrm{sh},m}^{t-1}\left(V_{\mathrm{sh},m}^{t-1}\right)^{\top}, \qquad \mathrm{rank}\left(\Delta_{\mathrm{sh},m}^{t-1}\right) \le R_m. \tag{7}$$

Singular values rank the contribution of orthogonal update directions under the Frobenius metric. CASC uses this spectral energy as a data-free proxy for dominant historical update geometry, rather than as a direct estimate of downstream task utility. It therefore keeps the leading spectral components of the accumulated bank fixed before allocating the remaining capacity to the current task.

Let r_m^{t-1} be the current bank rank. CASC reserves k_m^{t-1} highest-energy components as protected directions, where $0 \le k_m^{t-1} \le r_m^{t-1}$. Let $\mathcal{P}_m^{t-1}$ be the selected index set. The protected output-side and input-side subspaces are

$$U_{p,m} = U_{\mathrm{sh},m}^{t-1}[:, \mathcal{P}_m^{t-1}], \qquad V_{p,m} = V_{\mathrm{sh},m}^{t-1}[:, \mathcal{P}_m^{t-1}]. \tag{8}$$

The corresponding protected component is carried forward unchanged:

$$\Delta_{p,m}^{t-1} = U_{p,m}\Sigma_{p,m}V_{p,m}^{\top}, \qquad \Delta_{\mathrm{unp},m}^{t-1} = \Delta_{\mathrm{sh},m}^{t-1} - \Delta_{p,m}^{t-1}, \tag{9}$$

where $\Sigma_{p,m}$ contains the singular values indexed by $\mathcal{P}_m^{t-1}$. The protected part keeps the leading components of the accumulated update fixed under the spectral proxy, while the unprotected part remains available for recompression.

3.6 Shared-Interface Principle

A natural question is why a low-rank LoRA update and a dense projector shift can be governed by one consolidation rule. After the conversion in Eq. (4), both are module-wise matrices equipped with the Frobenius norm. Thus, for each module m, CASC defines the same Frobenius protected-subspace reconstruction objective, regardless of whether $\Delta_{t,m}$ comes from a LoRA module or a projector parameter. Let

$$Q_{U,m} = I - U_{p,m}U_{p,m}^{\top}, \qquad Q_{V,m} = I - V_{p,m}V_{p,m}^{\top}.$$

We use $\mathrm{TopK}_q(\cdot)$ to denote the rank-q truncated SVD approximation, with $\mathrm{TopK}_0(\cdot) = 0$. Under the Frobenius reconstruction objective, the optimal free update outside the protected input and output subspaces is

$$\begin{aligned} Z_m^{\star} &= \arg \min_{\substack{\mathrm{rank}(Z) \le R_m - k_m^{t-1} \\ U_{p,m}^{\top}Z = 0,\ ZV_{p,m} = 0}} \left\| \Delta_{\mathrm{unp},m}^{t-1} + \Delta_{t,m} - Z \right\|_F^2 \\ &= \mathrm{TopK}_{R_m - k_m^{t-1}}\left(\Delta_{\mathrm{unp},m}^{t-1} + Q_{U,m}\Delta_{t,m}Q_{V,m}\right). \end{aligned} \tag{10}$$

Equation (10) provides the formal justification that LoRA and projector consolidation can be aligned by one spectral interface. The optimizer depends only on the update matrix and its protected subspaces, not on the origin of the update. Therefore, LoRA and projector modules keep separate banks and budgets, but share the same memory criterion: preserve protected historical directions and insert the best rank-limited compatible residual under the Frobenius objective. Appendix 6 proves Eq. (10).

Before inserting the current update, CASC removes components that directly overlap with the protected input and output subspaces:

$$\Delta_{t,m}^{\mathrm{ts}} = Q_{U,m}\Delta_{t,m}Q_{V,m}. \tag{11}$$

Equivalently, the right projection reduces dependence on protected input directions, and the left projection prevents the update from writing into protected output directions. We refer to $\Delta_{t,m}^{\mathrm{ts}}$ as the two-sided residual. This residual is the component of the current task update that is orthogonal to the protected input and output subspaces, and can therefore be inserted without directly writing into those protected subspaces.

Finally, CASC compresses the two-sided residual together with the unprotected component under the remaining rank budget:

$$\Delta_{\mathrm{sh},m}^{t} = \Delta_{p,m}^{t-1} + \mathrm{TopK}_{R_m - k_m^{t-1}}\left(\Delta_{\mathrm{unp},m}^{t-1} + \Delta_{t,m}^{\mathrm{ts}}\right). \tag{12}$$

Equation (12) is the main update rule of CASC. Here, $\mathrm{TopK}_{R_m - k_m^{t-1}}(\cdot)$ keeps the leading spectral components within the remaining rank capacity after reserving the protected directions. Protected components bypass TopK recompression, whereas only the unprotected bank and the projected current residual compete for the remaining rank budget. Algorithm 1 summarizes the procedure.

Algorithm 1 Conflict-Aware Spectral Consolidation

Require: Task sequence $\{\mathcal{D}_t\}_{t=1}^{T}$, base projector P_0, LoRA bank rank R, projector bank rank R_P, protected-rank policy
Ensure: Consolidated MLLM with one LoRA adapter and one projector
Initialize module-wise spectral banks $\mathcal{B}^0$ as empty banks
for $t = 1$ to T **do**
  Adapt the LoRA adapter and projector on current task $\mathcal{D}_t$
  Extract current LoRA updates and projector-side shifts $\mathcal{U}_t = \{\Delta_{t,m}\}_{m \in \mathcal{M}_t}$
  Map each $\Delta_{t,m}$ to the spectral form in Eq. (4)
  for each active module $m \in \mathcal{M}_t$ **do**
    Retrieve the previous consolidated bank $\Delta_{\mathrm{sh},m}^{t-1}$
    Set $R_m = R$ for LoRA modules and $R_m = R_P$ for projector parameters
    Select protected directions $(U_{p,m}, V_{p,m})$ from the top-energy bank components
    Split the bank into $\Delta_{p,m}^{t-1}$ and $\Delta_{\mathrm{unp},m}^{t-1}$
    Compute the two-sided residual $\Delta_{t,m}^{\mathrm{ts}}$ by Eq. (11)
    Update the bank by Eq. (12)
  end for
  Keep inactive banks unchanged
end for
Reconstruct the consolidated LoRA adapter and projector from the final banks
return Consolidated MLLM with one LoRA adapter and one projector

3.7 Factor-Space Implementation and Reconstruction

The $\mathrm{TopK}(\cdot)$ operation in Eq. (12) is implemented in factor space. Given a candidate update written as the sum of low-rank factors, CASC concatenates the left and right bases, applies QR decomposition, and performs SVD only on the resulting small core matrix. This avoids materializing dense update matrices for large language modules. The detailed algebraic form is provided in Appendix 7.2.

After the last task, the final spectral banks are written back to standard model parameters. For a language-side LoRA module L, the consolidated bank $\Delta_{\mathrm{sh},L} = U_{\mathrm{sh},L}\Sigma_{\mathrm{sh},L}V_{\mathrm{sh},L}^{\top}$ is converted to standard LoRA factors. If the deployed LoRA layer uses scaling $s = \alpha/r$, we set

$$A_{\mathrm{sh},L} = V_{\mathrm{sh},L}^{\top}, \qquad B_{\mathrm{sh},L} = s^{-1}U_{\mathrm{sh},L}\Sigma_{\mathrm{sh},L},$$

so that $sB_{\mathrm{sh},L}A_{\mathrm{sh},L} = \Delta_{\mathrm{sh},L}$. Equivalently, one can absorb the scaling into $B_{\mathrm{sh},L}$ and set the deployment scaling to one. For projector parameter p, the bank stores the consolidated shift $\Delta P_{\mathrm{sh},p}$ relative to the base projector. The deployed projector is reconstructed as

$$P_{\mathrm{eff},p} = P_{0,p} + \Delta P_{\mathrm{sh},p}. \tag{13}$$

The final model therefore contains one consolidated LoRA adapter and one consolidated projector, and inference uses the same adapted MLLM across all tasks.

3.8 Complexity Analysis

We analyze one module m with update shape $d_{\mathrm{out}} \times d_{\mathrm{in}}$, bank rank budget R_m, incoming rank $\hat{r}_m$, and protected rank k_m. CASC stores one rank-R_m spectral bank per adapted module, with memory cost $O(R_m(d_{\mathrm{out}} + d_{\mathrm{in}}))$ plus singular values. Since R_m is fixed, memory is bounded by the prescribed rank budget rather than the number of tasks, unlike methods that retain task-specific adapters or experts. Consolidation is performed only at task boundaries and is not used during inference. The two-sided projection in Eq. (11) costs $O((d_{\mathrm{out}} + d_{\mathrm{in}})k_m\hat{r}_m)$ in factor space. The recompression in Eq. (12) applies QR decomposition and SVD only to a small core matrix controlled by R_m and $\hat{r}_m$. Thus, CASC adds bounded consolidation overhead, avoids replay data and task-dependent routing, and deploys one consolidated LoRA adapter with one consolidated projector.

4 Experiments

4.1 Experimental Setup

Benchmarks. We evaluate CASC on MLLM-CL [3], which provides two challenging benchmarks: Domain Continual Learning (DCL) and Ability Continual Learning (ACL). DCL evaluates continual adaptation to domain-specific multimodal knowledge across remote sensing, medical imaging, autonomous driving, science, and finance. ACL evaluates continual adaptation to general multimodal abilities, including OCR, math and logic, visual perception, and GUI-agent tasks. Together, DCL and ACL provide a comprehensive evaluation suite for assessing a method's ability to preserve and integrate knowledge under both domain shifts and ability shifts in multimodal continual learning.

Baselines. We compare with LoRA-FT, O-LoRA [2], MoELoRA [37], CL-MoE [39], SEFE [41], DISCO [38], HiDe [40], and RobustMerge [12]. These representative and competitive baselines provide broad coverage of major continual MLLM strategies, including sequential LoRA tuning, orthogonal or regularized adaptation, expert- or routing-based continual learning, and post-hoc parameter-efficient merging. Appendix 9 provides additional details.

Implementation Details. We use LLaVA-1.5-7B [49] for the main DCL and ACL experiments. Training hyperparameters, task order, data processing, and evaluation settings follow the MCITlib benchmark protocol [4]. Additional experimental results, including DCL statistics over three training seeds and Qwen3-VL-8B-Instruct evaluation, are provided in Appendix 8.

Evaluation Metrics. Following MLLM-CL [3, 4], we report the final accuracy on each task and three summary metrics: mean final performance (MFN), mean average accuracy (MAA), and backward transfer (BWT). Let $\mathbf{R} \in \mathbb{R}^{T\times T}$ denote the continual evaluation matrix, where $R_{t,k}$ is the accuracy on task k after completing task t ($t \geq k$). MFN summarizes the final-stage performance as $\mathrm{MFN} = \frac{1}{T}\sum_{k=1}^{T} R_{T,k}$. MAA first averages accuracy over the tasks seen at each stage, then gives equal weight to all stages: $\mathrm{MAA} = \frac{1}{T}\sum_{t=1}^{T}\left(\frac{1}{t}\sum_{k=1}^{t} R_{t,k}\right)$. BWT measures the final performance change on earlier tasks as $\mathrm{BWT} = \frac{1}{T-1}\sum_{k=1}^{T-1}(R_{T,k} - R_{k,k})$. These metrics reflect final performance, average performance along the task stream, and backward transfer, respectively.

4.2 Results

Main Results. Table 1 compares CASC with continual tuning, MoE-based, and merging-based baselines on DCL and ACL. CASC achieves the highest MFN in both settings, improving over the strongest baseline by 1.76 points on DCL and 1.48 points on ACL. This indicates that one consolidated LoRA–projector pair can effectively absorb task knowledge without task-specific routing. CASC also achieves the best MAA on both benchmarks, showing that the streaming spectral banks remain stable throughout the task sequence. For retention, CASC obtains the strongest BWT on DCL and competitive BWT on ACL, suggesting that the protected-subspace rule reduces overwriting of historical directions. These results support the core design of CASC: jointly consolidating language-side LoRA updates and projector shifts improves both multimodal adaptation and retention.

Forgetting and Stability. The gains of CASC are not limited to final-stage accuracy. A reliable continual MLLM should also maintain strong intermediate models as the task stream evolves. The high MAA of CASC reflects this training-stage stability, while its BWT on DCL suggests that later task updates cause less degradation of earlier domain knowledge. Figure 3 further visualizes this retention behavior. Each off-diagonal entry shows normalized retention on an earlier task after subsequent tasks have been learned. Compared with CL-MoE, CASC better preserves earlier-task accuracy, showing stronger retention of previously learned domains and abilities.

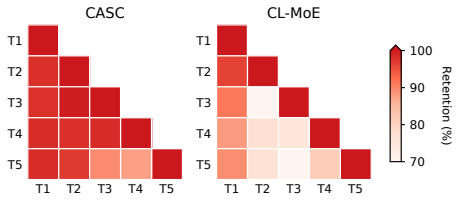


Figure 3: Knowledge retention on DCL.

Table 1: Performance comparison on DCL and ACL tasks. DCL tasks are RS, Med, AD, Sci, and Fin; ACL tasks are OCR, Math, VP, and APP. MFN denotes mean final performance over all tasks. Entries are single-run results. DCL statistics over three training seeds are reported separately in Appendix 8.1.

Methods	Domain Continual Learning (DCL)								Ability Continual Learning (ACL)						
	RS	Med	AD	Sci	Fin	MFN	MAA	BWT	OCR	Math	VP	APP	MFN	MAA	BWT
LoRA-FT	70.44	47.42	29.62	41.43	89.39	55.66	62.15	-12.62	21.60	30.80	56.47	31.62	35.12	35.68	-7.52
O-LoRA	76.03	44.95	28.89	41.28	78.72	53.97	61.36	-7.70	27.20	30.40	38.96	28.50	31.27	33.81	-11.25
MoELoRA	74.41	46.30	29.40	40.49	87.83	55.69	62.03	-12.11	27.70	31.70	51.78	26.50	34.42	35.60	-7.35
CL-MoE	71.17	46.74	28.97	40.77	89.84	55.50	62.26	-14.00	28.70	33.10	53.30	31.13	36.56	36.02	-4.10
SEFE	71.86	52.08	36.71	42.17	90.05	58.57	64.62	-9.97	26.00	33.40	57.74	33.75	37.72	36.59	-4.35
DISCO	76.16	47.75	45.28	43.45	88.86	60.30	64.42	-6.26	32.70	31.80	58.76	29.62	38.22	35.58	**-0.94**
HiDe	72.46	46.30	33.26	37.14	73.69	52.57	58.55	-6.01	24.70	32.10	21.95	30.00	27.19	31.47	-13.08
RobustMerge	58.38	35.77	16.77	28.61	84.94	44.89	60.96	-24.50	25.50	31.20	38.45	6.25	25.35	31.37	-6.97
CASC	79.35	60.45	36.54	44.45	89.51	**62.06**	**66.47**	**-3.60**	32.20	29.70	60.28	36.63	**39.70**	**37.15**	-2.57

4.3 Ablation and Hyperparameter Study

Component Ablation. Table 2 evaluates the contribution of the two consolidated components in CASC. Removing LoRA consolidation reduces MFN from 62.06 to 55.94, confirming that language-side spectral consolidation is essential for integrating sequential task updates. Removing projector consolidation reduces MFN to 57.92, showing that the multimodal interface also plays a substantial role in retention. The gap between each partial variant and full CASC supports the joint design. LoRA consolidation integrates language-side task knowledge, while projector consolidation preserves the visual-language interface required for unified multimodal inference. Table 3 further compares alternative projector choices with the consolidated LoRA held fixed.

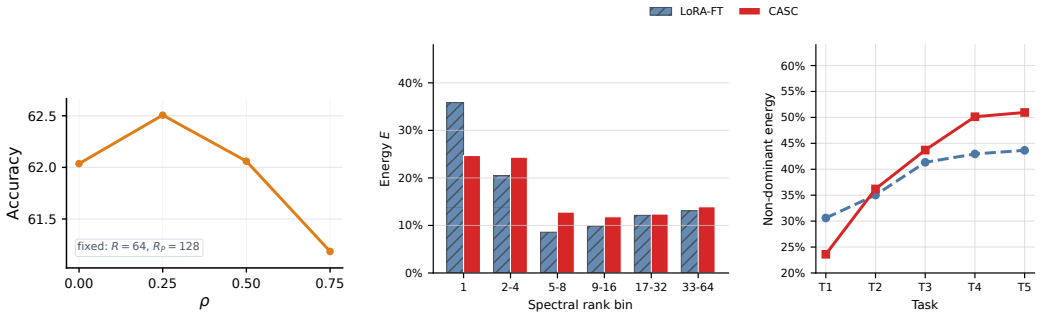
Table 2: Component ablation.

Variant	LoRA	Proj.	MFN
w/o LoRA	–	✓	55.94
w/o Proj.	✓	–	57.92
Full CASC	✓	✓	**62.06**

Hyperparameter Sensitivity. Figure 4(a) summarizes the protection-ratio trend, while Appendix 8.5 reports rank-capacity sensitivity. The protection ratio ρ controls how many high-energy directions in the previous bank are explicitly preserved before compression. CASC remains stable at moderate ρ, achieving the best accuracy at $\rho = 0.25$ and similar performance at $\rho = 0.50$. Performance drops when ρ increases to 0.75, reflecting the capacity trade-off in CASC. Insufficient protection leaves accumulated directions vulnerable to repeated recompression, whereas excessive protection reserves too much capacity for historical directions and limits incoming residual updates.

4.4 Mechanistic Analysis and Insights

LoRA Spectral Distribution. We further examine whether CASC exhibits the intended spectral behavior of the bank update rule: preserving dominant historical directions while inserting later-task residuals into remaining spectral capacity. For each adapted LoRA module m, we form the effective update $\Delta W_m = \frac{\alpha}{r} B_m A_m$ and compute its singular values $\{\sigma_{m,i}\}$. We measure cumulative spectral energy as $E_m(k) = \frac{\sum_{i=1}^{k} \sigma_{m,i}^2}{\sum_i \sigma_{m,i}^2}$. A high value at small k indicates that the update is concentrated in a few leading directions. We report both rank-bin energy, obtained from differences between adjacent cumulative ranges, and non-dominant energy, defined as the fraction of energy outside the top-4 singular directions. Figure 4(b) compares CASC with sequential LoRA fine-tuning. At the final checkpoint, LoRA-FT concentrates much of its energy in the first singular direction, whereas CASC assigns more energy to later rank bins. Across the task sequence, CASC also retains higher non-dominant energy than LoRA-FT, indicating more stable residual capacity. This supports our mechanism: protected directions are kept stable, while compatible residual updates are inserted into unprotected spectral capacity instead of repeatedly overwriting dominant directions. Additional qualitative case studies with task-specific examples are provided in Appendix 8.8.



(a) Protection-ratio sensitivity.

(b) LoRA spectral energy distribution.

Figure 4: Hyperparameter and mechanistic analysis. (a) Effect of the protection ratio ρ on DCL. (b) Left: normalized energy over spectral rank bins at the final checkpoint. Right: non-dominant spectral energy across tasks, measured as the energy outside the top-4 singular directions.

Controlled Projector Comparison. To isolate projector consolidation, we fix the consolidated LoRA adapter and vary only the final projector on DCL. Table 3 shows that CASC achieves the highest MFN in this comparison, exceeding rank matched SVD by 1.08 points at the same projector rank budget of 128. This supports a benefit beyond spectral compression of the averaged projector shift alone. Appendix 8.6 provides the projector definitions and construction details.

Table 3: Comparison of projector variants on DCL with a fixed consolidated LoRA. Entries are single-run results. Bold indicates the best value in each column.

Projector variant	RS	Med	AD	Sci	Fin	MFN ↑
Base P_0	75.51	50.67	38.24	44.20	88.33	59.39
Last task P_T	74.64	50.77	31.47	42.81	**89.89**	57.92
Uniform average	77.41	54.46	**38.68**	**45.33**	89.41	61.06
SVD (rank matched)	77.33	54.35	38.54	45.20	89.49	60.98
CASC (ours)	**79.35**	**60.45**	36.54	44.45	89.51	**62.06**

5 Conclusion

We study continual task-boundary LoRA–projector consolidation for MLLMs, where sequential updates are consolidated into one adapted model without replay data or inference routing. We identified two key challenges in this setting: interference among sequential LoRA updates and compatibility between the merged LoRA update and the multimodal projector. To address them, CASC maintains separate LoRA and projector banks of fixed capacity and updates them with a shared protected-subspace rule. Among the compared methods, CASC achieves the highest MFN and MAA on both DCL and ACL, with the highest BWT on DCL and competitive BWT on ACL. These results highlight the importance of consolidating both language-side updates and the visual-language interface in multimodal continual learning.

Acknowledgments and Disclosure of Funding

This work was supported in part by the National Natural Science Foundation of China (Grant No. 92370204), in part by the National Key R&D Program of China (Grant No. 2023YFF0725001), in part by Guangdong Provincial Key Laboratory of Frontier Basic Science for All-domain Intelligence, in part by the Guangdong Basic and Applied Basic Research Foundation (Grant No. 2023B1515120057), and in part by the Key-Area Special Project of Guangdong Provincial Ordinary Universities (2024ZDZX1007).

Competing interests: The authors declare that they have no competing financial or non-financial interests, including financial relationships outside the submitted work during the past 36 months, that could be perceived as influencing this work.

References

[1] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

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

[3] 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.

[4] 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*, 2025.

[5] Mao-Lin Luo, Zi-Hao Zhou, Yi-Lin Zhang, Yuanyu Wan, Min-Ling Zhang, and Tong Wei. Keeplora: Continual learning with residual gradient adaptation. In *International Conference on Learning Representations*, volume 2026, pages 136017–136040, 2026.

[6] Qianyi Cai, Ziyue Qiao, Rui Cai, Huijie Liu, Junyi Li, Xiao Luo, and Hui Xiong. Continual test-time training on graphs via adaptive prompts integration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

[7] Lucas Page-Caccia, Edoardo Maria Ponti, Zhan Su, Matheus Pereira, Nicolas Le Roux, and Alessandro Sordoni. Multi-head adapter routing for cross-task generalization. *Advances in Neural Information Processing Systems*, 36:56916–56931, 2023.

[8] Ted Zadouri, Ahmet Üstün, Arash Ahmadian, Beyza Ermis, Acyr Locatelli, and Sara Hooker. Pushing mixture of experts to the limit: Extremely parameter efficient MoE for instruction tuning. In *The Twelfth International Conference on Learning Representations*, 2024.

[9] Xun Wu, Shaohan Huang, and Furu Wei. Mixture of LoRA experts. In *The Twelfth International Conference on Learning Representations*, 2024.

[10] Ziyu Zhao, Leilei Gan, Guoyin Wang, Wangchunshu Zhou, Hongxia Yang, Kun Kuang, and Fei Wu. LoRARetriever: Input-aware LoRA retrieval and composition for mixed tasks in the wild. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4447–4462, 2024.

[11] 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.

[12] Fanhu Zeng, Haiyang Guo, Fei Zhu, Li Shen, and Hao Tang. RobustMerge: Parameter-efficient model merging for MLLMs with direction robustness. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[13] Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. TIES-Merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36:7093–7115, 2023.

[14] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[15] Oleksiy Ostapenko, Mihai Puscas, Tassilo Klein, Patrick Jahnichen, and Moin Nabi. Learning to remember: A synaptic plasticity driven framework for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11321–11329, 2019.

[16] Yuxuan Luo, Jinpeng Chen, Runmin Cong, Horace Ho Shing Ip, and Sam Kwong. Trace back and go ahead: Completing partial annotation for continual semantic segmentation. *Pattern Recognition*, 165:111613, 2025.

[17] Jinpeng Chen, Runmin Cong, Yuxuan Luo, Horace Ip, and Sam Kwong. Saving 100x storage: Prototype replay for reconstructing training sample distribution in class-incremental semantic segmentation. *Advances in Neural Information Processing Systems*, 36:35988–35999, 2023.

[18] Lucas Caccia, Rahaf Aljundi, Nader Asadi, Tinne Tuytelaars, Joelle Pineau, and Eugene Belilovsky. New insights on reducing abrupt representation change in online continual learning. *arXiv preprint arXiv:2104.05025*, 2021.

[19] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154, 2018.

[20] Cheng Chen, Ji Zhang, Jingkuan Song, and Lianli Gao. Class gradient projection for continual learning. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 5575–5583, 2022.

[21] Jinpeng Chen, Runmin Cong, Yuxuan Luo, Horace Ho Shing Ip, and Sam Kwong. Strike a balance in continual panoptic segmentation. In *European Conference on Computer Vision*, pages 126–142. Springer, 2024.

[22] Yuxuan Luo, Runmin Cong, Xialei Liu, Horace Ho Shing Ip, and Sam Kwong. Modeling inner-and cross-task contrastive relations for continual image classification. *IEEE Transactions on Multimedia*, 26:10842–10853, 2024.

[23] Haeyong Kang, Rusty John Lloyd Mina, Sultan Rizky Hikmawan Madjid, Jaehong Yoon, Mark Hasegawa-Johnson, Sung Ju Hwang, and Chang D Yoo. Forget-free continual learning with winning subnetworks. In *International Conference on Machine Learning*, pages 10734–10750. PMLR, 2022.

[24] Tatsuya Konishi, Mori Kurokawa, Chihiro Ono, Zixuan Ke, Gyuhak Kim, and Bing Liu. Parameter-level soft-masking for continual learning. In *International Conference on Machine Learning*, pages 17492–17505. PMLR, 2023.

[25] Lingfeng He, De Cheng, Huaijie Wang, Xi Yang, Nannan Wang, and Xinbo Gao. Task-driven subspace decomposition for knowledge sharing and isolation in loRA-based continual learning. In *Forty-third International Conference on Machine Learning*, 2026.

[26] Meng Lou, Yunxiang Fu, and Yizhou Yu. Scaling continual learning to 300+ tasks with bi-level routing mixture-of-experts. In *Forty-third International Conference on Machine Learning*, 2026.

[27] Qianyi Cai, Ziyue Qiao, Minghao Yang, Xiao Luo, and Hui Xiong. Graph cross-domain continual fine-tuning via orthogonal lora routing with contrastive expert specialization. In *Proceedings of the ACM Web Conference 2026*, pages 1026–1037, 2026.

[28] Da-Wei Zhou, Yuanhan Zhang, Yan Wang, Jingyi Ning, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Learning without forgetting for vision-language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(6):4489–4504, 2025.

[29] Kshitij Gupta, Benjamin Thérien, Adam Ibrahim, Mats L Richter, Quentin Anthony, Eugene Belilovsky, Irina Rish, and Timothée Lesort. Continual pre-training of large language models: How to (re) warm your model? *arXiv preprint arXiv:2308.04014*, 2023.

[30] Chengyue Wu, Yukang Gan, Yixiao Ge, Zeyu Lu, Jiahao Wang, Ye Feng, Ying Shan, and Ping Luo. LLaMA Pro: Progressive LLaMA with block expansion. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6518–6537, 2024.

[31] Joel Jang, Seonghyeon Ye, Changho Lee, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, and Minjoon Seo. TemporalWiki: A lifelong benchmark for training and evaluating ever-evolving language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6237–6250, 2022.

[32] Andrea Cossu, Antonio Carta, Lucia Passaro, Vincenzo Lomonaco, Tinne Tuytelaars, and Davide Bacciu. Continual pre-training mitigates forgetting in language and vision. *Neural Networks*, 179:106492, 2024.

[33] Shu Yang, Muhammad Asif Ali, Cheng-Long Wang, Lijie Hu, and Di Wang. MoRAL: MoE augmented LoRA for LLMs' lifelong learning. *arXiv preprint arXiv:2402.11260*, 2024.

[34] Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Wei Shen, Limao Xiong, Yuhao Zhou, Xiao Wang, Zhiheng Xi, Xiaoran Fan, et al. LoRAMoE: Alleviating world knowledge forgetting in large language models via MoE-style plugin. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1932–1945, 2024.

[35] Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. Progressive prompts: Continual learning for language models. *arXiv preprint arXiv:2301.12314*, 2023.

[36] Tairan Huang, Yili Wang, Beibei Hu, Yiting Shi, Qiutong Li, Changlong He, and Jianliang Gao. UNIT: Unleash large language models potential for graph continual learning. *arXiv preprint arXiv:2607.10159*, 2026.

[37] Cheng Chen, Junchen Zhu, Xu Luo, Heng T Shen, Jingkuan Song, and Lianli Gao. CoIN: A benchmark of continual instruction tuning for multimodel large language models. *Advances in Neural Information Processing Systems*, 37:57817–57840, 2024.

[38] 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*, pages 1325–1335, 2025.

[39] 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*, pages 19608–19617, 2025.

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

[41] 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*, 2025.

[42] Zhen-Hao Xie, Jun-Tao Tang, Yu-Cheng Shi, Han-Jia Ye, De-Chuan Zhan, and Da-Wei Zhou. SAME: Stabilized mixture-of-experts for multimodal continual instruction tuning. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306, pages 139659–139683. PMLR, 2026.

[43] Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. In *International Conference on Learning Representations*, 2021.

[44] 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*, 2024.

[45] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International Conference on Machine Learning*, pages 19730–19742. PMLR, 2023.

[46] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. InstructBLIP: Towards general-purpose vision-language models with instruction tuning. *Advances in Neural Information Processing Systems*, 36:49250–49267, 2023.

[47] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-VL: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 2023.

[48] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. 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*, pages 24185–24198, 2024.

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

[50] Yili Wang, Lu Dai, Tairan Huang, Yijie Xu, and Hui Xiong. Vl-eraser: Vacuum distillation for machine unlearning in vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 31504–31513, 2026.

[51] Lu Dai, Ziyang Rao, Yili Wang, Hanqing Wang, Hao Liu, and Hui Xiong. Towards mechanistically understanding why memorized knowledge fails to generalize in large language model finetuning. *arXiv preprint arXiv:2607.08393*, 2026.

[52] Yijie Xu, Huizai Yao, Zhiyu Guo, Pengteng Li, Aiwei Liu, Xuming Hu, Weiyu Guo, and Hui Xiong. You only need 4 extra tokens: Synergistic test-time adaptation for llms. *arXiv preprint arXiv:2510.10223*, 2025.

[53] Xianpeng Shang, Jiang Li, Zehua Duo, Qianyi Cai, and Xiangdong Su. Training–inference consistent segmented execution for long-context LLMs. In *Forty-third International Conference on Machine Learning*, 2026.

[54] Yijie Xu, Qianyi Cai, Huizai Yao, Yili Wang, Tianfu Wang, Cehao Yang, Xingbo Yao, Zhiyu Guo, Aiwei Liu, Xuming Hu, et al. Unsupervised post-training of foundation models: A survey. In *Findings of the Association for Computational Linguistics: EMNLP 2026*, pages 43485–43504, 2026.

[55] Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P Vetrov, and Andrew G Wilson. Loss surfaces, mode connectivity, and fast ensembling of DNNs. *Advances in Neural Information Processing Systems*, 31, 2018.

[56] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*, pages 23965–23998. PMLR, 2022.

[57] Samuel Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git Re-Basin: Merging models modulo permutation symmetries. In *The Eleventh International Conference on Learning Representations*, 2023.

[58] Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are Super Mario: Absorbing abilities from homologous models as a free lunch. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

[59] Anke Tang, Enneng Yang, Li Shen, Yong Luo, Han Hu, Lefei Zhang, Bo Du, and Dacheng Tao. Merging on the fly without retraining: A sequential approach to scalable continual model merging. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Appendix

6 Why a Shared Spectral Interface is Well Defined

This appendix provides the formal justification for the shared spectral interface used by CASC. The result does not assume that LoRA modules and projector parameters play the same role in the MLLM. LoRA updates modify language-side linear transformations, whereas projector shifts modify the visual-language interface. Instead, the key observation is that after each task adaptation is represented as a module-wise update matrix, both types of updates can be measured under the same Frobenius geometry. Under this geometry, the protected-subspace consolidation objective has the same closed-form solution for both LoRA updates and projector shifts.

The proof is stated for one recurrent consolidation step. At step t, the previous consolidated bank $\Delta_{\mathrm{sh}}^{t-1}$ is fixed, and the current task update Δ_t is inserted into this bank. Since the derivation only uses the current update matrix, the previous bank, the protected subspaces, and the rank budget, it applies repeatedly along the task stream and independently to each LoRA or projector module.

6.1 Module Updates in a Common Matrix Space

For any adapted module m, let the current task update be

$$\Delta_{t,m} \in \mathbb{R}^{d_{\mathrm{out}} \times d_{\mathrm{in}}}.$$

We use the Frobenius inner product and norm

$$\langle A, B \rangle_F = \mathrm{tr}(A^\top B), \qquad \|A\|_F^2 = \langle A, A \rangle_F. \tag{14}$$

For a LoRA module, the effective task update is

$$\Delta W_{t,m} = \frac{\alpha}{r} B_{t,m} A_{t,m}, \tag{15}$$

which is a low-rank matrix. For a projector parameter block, the update is the dense shift from the base projector,

$$\Delta P_{t,m} = P_{t,m} - P_{0,m}. \tag{16}$$

Vector parameters can be treated as single-column matrices, and multi-layer projectors are handled block by block. Thus, in both cases, the task update can be written in spectral form:

$$\Delta_{t,m} = U_{t,m} \Sigma_{t,m} V_{t,m}^\top. \tag{17}$$

For LoRA, retaining all nonzero singular components preserves Eq. (15) exactly. For projector shifts, Eq. (17) is also exact before bank compression. Therefore, the consolidation rule only needs a module-wise matrix update $\Delta_{t,m}$; it does not depend on whether the update was produced by a LoRA product or by a dense projector shift.

6.2 Constrained Consolidation Objective

We now analyze a single module and omit the module index. Let the consolidated bank before task t be

$$\Delta_{\mathrm{sh}}^{t-1} = U_{\mathrm{sh}} \Sigma_{\mathrm{sh}} V_{\mathrm{sh}}^\top, \qquad \mathrm{rank}(\Delta_{\mathrm{sh}}^{t-1}) \le R. \tag{18}$$

CASC protects the top-k singular components of this bank:

$$\Delta_p^{t-1} = U_p \Sigma_p V_p^\top. \tag{19}$$

By the Eckart–Young–Mirsky theorem, Δ_p^{t-1} is the best rank-k approximation to the historical bank under the Frobenius norm:

$$\Delta_p^{t-1} = \mathrm{TopK}_k\big(\Delta_{\mathrm{sh}}^{t-1}\big) = \arg \min_{\mathrm{rank}(X) \le k} \big\|\Delta_{\mathrm{sh}}^{t-1} - X\big\|_F^2. \tag{20}$$

Let

$$P_U = U_p U_p^\top, \qquad P_V = V_p V_p^\top, \qquad Q_U = I - P_U, \qquad Q_V = I - P_V \tag{21}$$

be the orthogonal projectors onto the protected and unprotected output/input subspaces. The unprotected part of the historical bank is

$$\Delta_{\mathrm{unp}}^{t-1} = \Delta_{\mathrm{sh}}^{t-1} - \Delta_{p}^{t-1}. \tag{22}$$

Since $\Delta_{\mathrm{unp}}^{t-1}$ consists only of non-protected singular components from the same SVD, it lies outside the protected output and input subspaces:

$$P_U \Delta_{\mathrm{unp}}^{t-1} = 0, \qquad \Delta_{\mathrm{unp}}^{t-1} P_V = 0, \qquad \Delta_{\mathrm{unp}}^{t-1} = Q_U \Delta_{\mathrm{unp}}^{t-1} Q_V. \tag{23}$$

At step t, the ideal accumulated update before applying the rank budget would be

$$\Delta_{\mathrm{sh}}^{t-1} + \Delta_t.$$

However, CASC keeps the protected historical component fixed and only allows the remaining rank budget to store a compatible free update Z. Thus the new bank is restricted to the form

$$\Delta_{\mathrm{new}} = \Delta_{p}^{t-1} + Z, \tag{24}$$

where Z must avoid the protected output and input subspaces:

$$U_p^\top Z = 0, \qquad Z V_p = 0, \qquad \mathrm{rank}(Z) \le R - k. \tag{25}$$

Equivalently, $Z = Q_U Z Q_V$. Because Δ_p^{t-1} and Z occupy orthogonal output and input subspaces, the rank budget is preserved:

$$\mathrm{rank}(\Delta_{\mathrm{new}}) = \mathrm{rank}(\Delta_p^{t-1} + Z) = \mathrm{rank}(\Delta_p^{t-1}) + \mathrm{rank}(Z) \le k + (R-k) = R. \tag{26}$$

CASC therefore chooses Z by solving the following constrained reconstruction problem:

$$\begin{aligned} Z^\star = \arg\min_{Z} \quad & \left\| \Delta_{\mathrm{sh}}^{t-1} + \Delta_t - \left(\Delta_p^{t-1} + Z\right) \right\|_F^2 \\ \text{s.t.} \quad & \mathrm{rank}(Z) \le R - k, \qquad U_p^\top Z = 0, \qquad Z V_p = 0. \end{aligned} \tag{27}$$

6.3 Closed-Form Solution

We now derive the solution of Eq. (27). Using $\Delta_{\mathrm{sh}}^{t-1} = \Delta_p^{t-1} + \Delta_{\mathrm{unp}}^{t-1}$, the objective becomes

$$\left\| \Delta_{\mathrm{unp}}^{t-1} + \Delta_t - Z \right\|_F^2. \tag{28}$$

Next, decompose the incoming update Δ_t according to the protected and unprotected output/input subspaces:

$$\Delta_t = P_U \Delta_t P_V + P_U \Delta_t Q_V + Q_U \Delta_t P_V + Q_U \Delta_t Q_V. \tag{29}$$

The first three terms contain overlap with at least one protected subspace. By Eq. (23) and Eq. (25), both $\Delta_{\mathrm{unp}}^{t-1}$ and Z lie in the Q_U–Q_V block. Therefore, the first three blocks in Eq. (29) are Frobenius-orthogonal to both $\Delta_{\mathrm{unp}}^{t-1}$ and Z. Substituting Eq. (29) into Eq. (28), we obtain

$$\begin{aligned} & \left\| \Delta_{\mathrm{unp}}^{t-1} + \Delta_t - Z \right\|_F^2 \\ & = \left\| P_U \Delta_t P_V + P_U \Delta_t Q_V + Q_U \Delta_t P_V \right\|_F^2 + \left\| \Delta_{\mathrm{unp}}^{t-1} + Q_U \Delta_t Q_V - Z \right\|_F^2. \end{aligned} \tag{30}$$

The first term in Eq. (30) is independent of Z. Thus, minimizing Eq. (27) is equivalent to solving

$$Z^\star = \arg \min_{\substack{\mathrm{rank}(Z) \le R-k \\ Z = Q_U Z Q_V}} \left\| \Delta_{\mathrm{unp}}^{t-1} + Q_U \Delta_t Q_V - Z \right\|_F^2. \tag{31}$$

The target matrix

$$M = \Delta_{\mathrm{unp}}^{t-1} + Q_U \Delta_t Q_V \tag{32}$$

also satisfies $M = Q_U M Q_V$. Therefore, the constraint $Z = Q_U Z Q_V$ is automatically respected by any truncated SVD of M. By the Eckart–Young–Mirsky theorem, the optimal rank-$(R-k)$ approximation is

$$Z^\star = \mathrm{TopK}_{R-k}(M) = \mathrm{TopK}_{R-k}\left(\Delta_{\mathrm{unp}}^{t-1} + Q_U \Delta_t Q_V\right). \tag{33}$$

Substituting Eq. (33) into Eq. (24) gives

$$\Delta_{\mathrm{new}}^\star = \Delta_p^{t-1} + \mathrm{TopK}_{R-k}\left(\Delta_{\mathrm{unp}}^{t-1} + Q_U \Delta_t Q_V\right). \tag{34}$$

This proves the closed-form CASC update. □

6.4 Connection to the Main Method

Equation (33) corresponds to Eq. (10) in the main text after restoring the module index m. The projected update

$$\Delta_t^{\mathrm{ts}} = Q_U \Delta_t Q_V \tag{35}$$

corresponds to Eq. (11), and Eq. (34) corresponds to the bank update in Eq. (12). Therefore, the method in Sec. 3 is not a heuristic average of LoRA and projector updates. It is the solution of a rank-constrained reconstruction problem after the protected historical subspaces are fixed.

This derivation also explains why LoRA and projector modules can share one consolidation rule. The solution depends only on the update matrix, the protected input/output subspaces, the Frobenius objective, and the rank budget. It does not depend on whether Δ_t is a low-rank LoRA product or a dense projector shift. Thus, CASC can keep LoRA and projector banks separate, assign them different rank budgets, and still update them with the same protected-subspace memory rule.

Finally, the two-sided projection gives the exact orthogonality property

$$U_p^\top \Delta_t^{\mathrm{ts}} = U_p^\top Q_U \Delta_t Q_V = 0, \qquad \Delta_t^{\mathrm{ts}} V_p = Q_U \Delta_t Q_V V_p = 0. \tag{36}$$

Hence, the inserted residual neither writes into protected output directions nor depends on protected input directions. This is the formal reason CASC can preserve leading historical directions while allocating the remaining capacity to compatible current-task information. In the main method, this property corresponds to protecting high-energy historical components, removing overlap from incoming updates, and compressing only the compatible residual under the fixed module budget.

7 Additional Implementation Details

7.1 Projector Banks and Spectral Proxy

Projector banks. CASC builds projector banks at the level of trainable matrix-valued projector parameters. For each projector matrix m, we maintain an independent spectral bank with projector budget R_P. The inserted update is the dense shift relative to the base projector,

$$\Delta P_{t,m} = P_{t,m} - P_{0,m}.$$

After the final task, the deployed projector is reconstructed as

$$P_{\mathrm{eff},m} = P_{0,m} + \Delta P_{\mathrm{sh},m}.$$

Thus, projector consolidation is performed block by block and does not require storing or selecting task-specific projectors. During DCL and ACL consolidation, projector biases are reshaped into column matrices and processed by the same spectral rule, with rank at most one. In ACL, the first-stage evaluation uses the first expert's projector directly, and projector consolidation begins at the second stage.

Spectral proxy. The protected directions in CASC are selected by spectral energy of the accumulated bank. This should be understood as a data-free proxy for dominant update geometry, not as a direct estimator of downstream task utility. The choice is consistent with the setting considered in this work: previous task data, task identity, and task-specific gradient statistics are not available during consolidation. Under the Frobenius objective analyzed in Appendix 6, preserving high-energy singular directions retains the dominant low-rank approximation of the historical update.

Module-wise banks. CASC keeps separate banks for different adapted matrices. This follows the structure of LoRA and projector tuning: language-side LoRA updates and projector-side shifts have different shapes, scales, and functional roles. A single global decomposition over all adapted parameters would mix heterogeneous layers and substantially increase the consolidation cost. The module-wise design therefore preserves local update geometry and keeps the rank budget interpretable. CASC does not explicitly model cross-layer interactions; instead, it reduces direct overwriting within each adapted matrix and reconstructs one final model for inference.

Rank budgets and protection ratio. Each module has a fixed bank capacity R_m. Given its current historical rank r_m^{t-1}, the default protected rank is

$$k_m^{t-1} = \min\{r_m^{t-1}, \max[k_{m,\min}, \mathrm{round}(\rho_m r_m^{t-1})]\},$$

where ρ_m and $k_{m,\min}$ are the module's protection ratio and minimum protected rank. Table 1 uses this stage-dependent rule. The rank-capacity sweeps instead use fixed protected ranks, as specified in Table 8. A larger ρ_m preserves more historical directions, while a smaller ρ_m leaves more capacity for incoming updates.

Task-agnostic inference. After consolidation, all banks are reconstructed into ordinary model parameters: one consolidated LoRA adapter and one consolidated projector. At each stage, the same current model evaluates all tasks seen so far. Final task scores and MFN use the last-stage model, while MAA and BWT are computed from the progressive evaluation matrix. Task labels are used only for grouping test examples and computing MFN, MAA, and BWT; they are not used to select adapters, projectors, experts, or routes during inference.

7.2 Detailed Factor-Space Implementation and Reconstruction

CASC is implemented directly in factor space without materializing dense language-side update matrices. For LoRA extraction, we compute an exact orthonormal factorization of $\frac{\alpha}{r}BA$ using QR decomposition followed by SVD on a small core matrix.

The bank update in Eq. (12) is also computed in factor space. Suppose a candidate update is the sum of two low-rank factors,

$$\Delta_{\mathrm{cand}} = U_1\Sigma_1 V_1^\top + U_2\Sigma_2 V_2^\top. \tag{37}$$

We concatenate the factors as

$$U_c = [U_1, U_2], \qquad V_c = [V_1, V_2], \qquad \Sigma_c = \mathrm{diag}([\sigma_1, \sigma_2]),$$

so that $\Delta_{\mathrm{cand}} = U_c\Sigma_c V_c^\top$. We then compute QR decompositions

$$U_c = Q_U R_U, \qquad V_c = Q_V R_V,$$

and apply SVD only to the small core matrix:

$$R_U\Sigma_c R_V^\top = \widetilde{U}\widetilde{\Sigma}\widetilde{V}^\top. \tag{38}$$

Keeping the top-K singular components yields

$$\mathrm{TopK}_K(\Delta_{\mathrm{cand}}) = (Q_U\widetilde{U}_{[:,1:K]})\widetilde{\Sigma}_{1:K}(Q_V\widetilde{V}_{[:,1:K]})^\top. \tag{39}$$

This factor-space computation is used both for bank merging and for recompressing projected updates.

After the last task, the final shared bank of each module defines the deployed model. For a LoRA module m, we convert the final factor $\Delta_{\mathrm{sh},m} = U_{\mathrm{sh},m}\Sigma_{\mathrm{sh},m}V_{\mathrm{sh},m}^\top$ back to the standard LoRA parameterization. If the deployed LoRA layer uses scaling $s = \alpha/r$, we set

$$A_{\mathrm{sh},m} = V_{\mathrm{sh},m}^\top, \qquad B_{\mathrm{sh},m} = s^{-1}U_{\mathrm{sh},m}\Sigma_{\mathrm{sh},m}, \tag{40}$$

so that $sB_{\mathrm{sh},m}A_{\mathrm{sh},m} = \Delta_{\mathrm{sh},m}$ exactly. Equivalently, the scaling can be absorbed into $B_{\mathrm{sh},m}$ and the deployment scaling can be set to one. For a projector parameter m, we reconstruct the final weight as

$$P_{\mathrm{eff},m} = P_{0,m} + \Delta_{\mathrm{sh},m}. \tag{41}$$

8 Additional Experimental Results

8.1 Variation Across Training Seeds

Table 4 reports DCL results over three training seeds under the same reference protocol and task order, RS → Med → AD → Sci → Fin. Each run is summarized by its own progressive evaluation matrix before computing the mean and standard deviation across runs. These statistics are separate from the single-run reference results in Table 1. CASC has the highest mean MFN, MAA, and BWT among the four methods in this comparison, with a mean MFN advantage of 2.15 points over DISCO.

Table 4: DCL performance across three training seeds (mean $\pm$ standard deviation). MAA uses the stage-balanced definition in Section 4.

Method	MFN ↑	MAA ↑	BWT ↑
CL-MoE	55.19 ± 0.31	62.03 ± 0.24	-13.98 ± 0.02
SEFE	59.26 ± 0.69	64.61 ± 0.01	-8.72 ± 1.25
DISCO	59.78 ± 0.52	63.89 ± 0.54	-6.36 ± 0.10
CASC	$\mathbf{61.93 \pm 0.20}$	$\mathbf{66.39 \pm 0.18}$	$\mathbf{-3.43 \pm 0.34}$

8.2 Additional Backbone and Science Retention

We evaluate Qwen3-VL-8B-Instruct[1] on the same five-task DCL sequence. This experiment applies CASC to LoRA updates only, with bank rank $R = 64$, protection ratio $\rho = 0.5$, and minimum protected rank $k_{\min} = 8$. It tests transfer of the recurrent LoRA consolidation rule to another backbone.

Table 5: Qwen3-VL-8B-Instruct on DCL. All entries are single-run results. CASC consolidates LoRA updates only. Bold indicates the best value in each column among the listed methods.

Method	RS	Med	AD	Sci	Fin	MFN ↑	MAA ↑	BWT ↑
DISCO	77.66	60.63	42.18	55.45	92.04	65.59	70.78	-12.36
SEFE	79.11	**61.56**	**43.55**	55.31	**92.23**	66.35	**71.29**	-11.13
O-LoRA	77.46	60.83	40.07	54.94	92.12	65.08	70.80	-13.32
CASC	**80.03**	59.64	43.20	**78.73**	90.95	**70.51**	70.92	$\mathbf{-2.62}$

Table 5 shows that CASC improves MFN by 4.16 points over SEFE, the strongest baseline by MFN in this comparison. The gain is concentrated in Science: CASC reaches 78.73 compared with 55.31 for SEFE, a difference of 23.42 points. SEFE achieves the highest MAA.

Table 6 shows the CASC trajectory. Science accuracy is 81.36 immediately after Science and 78.73 after Finance, a decline of 2.63 points. This directly documents retention of Science by CASC after learning the subsequent task.

Table 6: Progressive DCL evaluation for CASC on Qwen3-VL-8B-Instruct, from the same single run as Table 5. Each row evaluates all tasks seen up to that stage.

Stage	RS	Med	AD	Sci	Fin
1: RS	81.53	–	–	–	–
2: Med	81.13	63.29	–	–	–
3: AD	80.80	61.95	45.88	–	–
4: Sci	80.72	61.16	46.72	81.36	–
5: Fin	80.03	59.64	43.20	78.73	90.95

8.3 Additional Six-Task UCIT Sequence

We additionally evaluate LLaVA-1.5-7B on the six-task UCIT sequence: ImageNet-R $\rightarrow$ ArxivQA $\rightarrow$ VizWiz $\rightarrow$ IconQA $\rightarrow$ CLEVR-Math $\rightarrow$ Flickr30k. The CASC configuration uses task experts with LoRA rank $r = 32$, bank capacities $R = 64$ and $R_P = 128$, protection ratios $\rho = \rho_P = 0.25$, and minimum protected ranks of 8 for both module types. Projector bias deltas are accumulated directly in this UCIT configuration. Progressive evaluation covers all 21 observed task-stage pairs, with MAA computed using the stage-balanced definition in Section 4.

CASC obtains the highest BWT among the compared methods, with -2.08 versus -5.58 for HiDe, while HiDe achieves the highest MFN and MAA.

[1] `https://huggingface.co/Qwen/Qwen3-VL-8B-Instruct`

Table 7: Performance on the six-task UCIT sequence. Entries are single-run results. MFN is the mean final score over the six tasks. Image-R, Viz-cap, and CLEVR denote ImageNet-R, VizWiz captioning, and CLEVR-Math, respectively. Bold indicates the best value in each column among the listed methods.

Method	Image-R	ArxivQA	Viz-cap	IconQA	CLEVR	Flickr30k	MFN ↑	MAA ↑	BWT ↑
LoRA-FT	65.50	76.23	44.41	66.37	54.87	**57.80**	60.86	67.61	−17.44
O-LoRA	83.40	**94.20**	41.51	58.50	56.80	53.37	64.63	75.98	−8.03
MoELoRA	68.57	77.73	44.26	65.43	45.20	57.31	59.75	75.57	−17.54
CL-MoE	67.80	73.47	44.58	69.37	47.80	57.40	60.07	74.89	−18.25
HiDe	86.93	91.37	**48.68**	67.80	47.87	52.95	**65.93**	**79.03**	−5.58
SEFE	83.53	65.77	45.72	**75.23**	**63.03**	57.74	65.17	76.64	−12.80
CASC	**89.87**	**94.20**	32.58	63.73	44.17	44.00	61.43	76.17	**−2.08**

8.4 DCL Baseline Trajectory Analysis

Figure 5 provides an additional view of DCL baseline behavior. Panel (a) compares final macro accuracy after the last task. Panel (b) reports CASC's domain-wise gain over the strongest baseline for each evaluation domain. Panel (c) tracks the average seen-domain accuracy of baseline methods across continual stages and marks the final CASC checkpoint. The figure shows that CASC improves the final task-agnostic checkpoint while common sequential baselines lose stability as more domains are introduced.

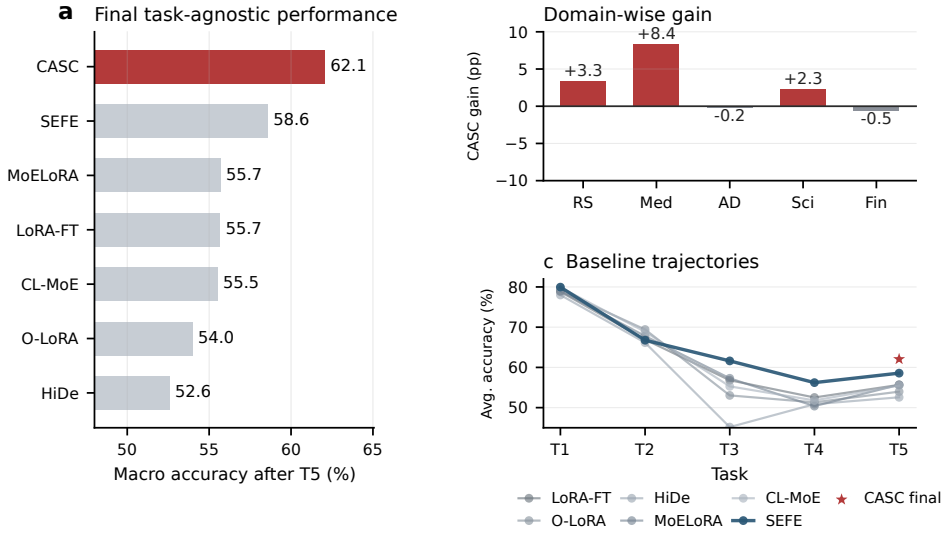


Figure 5: Supplementary DCL baseline trajectories.

8.5 Detailed Hyperparameter Sensitivity

The hyperparameter study uses one-factor-at-a-time sweeps rather than a full factorial grid. For the LoRA rank sweep, we vary the LoRA bank rank R while fixing the projector bank rank to $R_P = 128$ and keeping the protected ranks at $k = 32$ and $k_P = 64$. For the projector rank sweep, we vary the projector bank rank R_P while fixing $R = 64$, $k = 32$, and $k_P = 64$. This isolates the effect of bank capacity from the effect of protection strength.

In Fig. 6, the left-panel x-axis denotes the per-module LoRA bank rank R, which is the maximum number of spectral directions kept for each language-side LoRA bank. The right-panel x-axis denotes the projector bank rank R_P, which is the maximum rank of the consolidated projector-shift bank. Both are bank-capacity parameters rather than a shared global rank pool. Table 8 provides the corresponding task-wise scores. The rows show that CASC is not highly sensitive to moderate

changes in either bank capacity, while the protection ratio controls the trade-off between preserving accumulated directions and leaving capacity for later-task residuals.

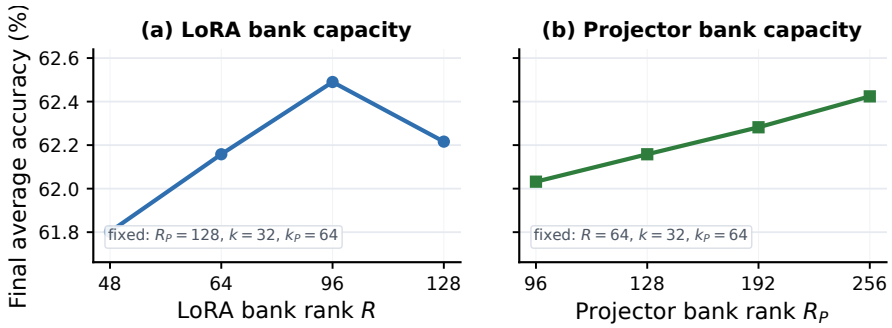


Figure 6: Bank-rank sensitivity on DCL. Left: LoRA bank capacity. Right: projector bank capacity.

Table 8: Detailed hyperparameter sensitivity of CASC on DCL tasks. Each block sweeps one hyperparameter group while keeping the remaining symbols fixed.

Study	Value	RS	Med	AD	Sci	Fin	Avg.
Sweep R	*fixed* $(k, R_P, k_P) = (32, 128, 64)$						
	48	79.48	60.08	36.27	43.71	89.47	61.80
	64	79.41	60.30	37.06	44.48	89.54	62.16
	96	79.51	60.54	37.66	45.01	89.73	**62.49**
	128	79.50	60.51	36.60	44.65	89.82	62.22
Sweep R_P	*fixed* $(R, k, k_P) = (64, 32, 64)$						
	96	79.48	59.90	36.95	44.27	89.56	62.03
	128	79.41	60.30	37.06	44.48	89.54	62.16
	192	79.46	60.66	37.06	44.65	89.58	62.28
	256	79.42	60.92	37.30	44.86	89.62	**62.42**
Sweep $\rho = \rho_P$	*fixed* $(R, R_P) = (64, 128)$						
	0.00	79.20	59.90	37.14	44.55	89.39	62.04
	0.25	79.31	61.20	37.12	45.17	89.73	**62.51**
	0.50	79.35	60.45	36.54	44.45	89.51	62.06
	0.75	79.34	59.48	34.77	42.75	89.59	61.19

Note. The LoRA and projector banks have capacities R and R_P and protected ranks k and k_P, respectively. The ratio sweep uses $k_{\min} = k_{P,\min} = 8$ for $\rho > 0$, but sets both minima to zero at $\rho = 0$. At $(R, R_P) = (64, 128)$, the $\rho = 0.50$ row matches Table 1 (62.06, first-merge LoRA $k = 16$), whereas the capacity sweeps fix $(k, k_P) = (32, 64)$ (62.16). The table reports final task-agnostic accuracy after the full DCL task sequence.

8.6 Controlled Comparison of Projector Alternatives

To isolate projector choice, we keep the consolidated LoRA adapter fixed and replace only the final projector on DCL. All five choices use the same evaluation protocol. We compare the base projector P_0, the last task projector P_T, the arithmetic average $T^{-1}\sum_{t=1}^{T} P_t$, a rank matched SVD projector, and the CASC projector, with $T = 5$. For each projector weight matrix m, the SVD alternative is

$$P_{\mathrm{SVD},m} = P_{0,m} + \mathrm{TruncSVD}_{q_m}\left(\frac{1}{T}\sum_{t=1}^{T}(P_{t,m} - P_{0,m})\right), \qquad q_m = \min\{128, d_{\mathrm{out},m}, d_{\mathrm{in},m}\}.$$

Its bias parameters use the arithmetic average of the task biases. The SVD rank budget matches the CASC projector bank capacity in this comparison.

Table 3 reports the full taskwise results. This comparison evaluates only the final model, so it reports MFN rather than MAA or BWT.

CASC achieves the highest MFN in this controlled comparison, exceeding rank matched SVD by 1.08 points. The aggregate improvement therefore extends beyond rank 128 spectral compression of the averaged projector shift alone.

8.7 Detailed Accuracy Matrices by Method

The following blocks organize the detailed results by method. For each method, the DCL matrix is shown on the left and the ACL matrix is shown on the right. The last column reports the average over evaluated domains in the corresponding row. CASC is evaluated after each task on all tasks seen so far.

LoRA-FT

Stage	RS	Med	AD	Sci	Fin	Avg.
RS	78.82	–	–	–	–	78.82
Med	76.06	57.80	–	–	–	66.93
AD	70.78	47.21	52.43	–	–	56.81
Sci	71.36	47.36	41.03	50.35	–	52.52
Fin	70.44	47.42	29.62	41.43	89.39	55.66

Stage	OCR	Math	VP	APP	Avg.
OCR	33.30	–	–	–	33.30
Math	30.80	32.00	–	–	31.40
VP	30.20	32.40	66.12	–	42.91
APP	21.60	30.80	56.47	31.62	35.12

O-LoRA

Stage	RS	Med	AD	Sci	Fin	Avg.
RS	79.04	–	–	–	–	79.04
Med	78.71	60.14	–	–	–	69.42
AD	77.48	45.16	36.48	–	–	53.04
Sci	77.36	45.12	36.47	46.30	–	51.31
Fin	76.03	44.95	28.89	41.28	78.72	53.97

Stage	OCR	Math	VP	APP	Avg.
OCR	33.40	–	–	–	33.40
Math	25.20	32.20	–	–	28.70
VP	27.80	33.10	64.72	–	41.87
APP	27.20	30.40	38.96	28.50	31.27

HiDe

Stage	RS	Med	AD	Sci	Fin	Avg.
RS	78.00	–	–	–	–	78.00
Med	74.99	57.31	–	–	–	66.15
AD	74.23	28.41	32.85	–	–	45.16
Sci	73.04	47.52	37.86	45.04	–	50.87
Fin	72.46	46.30	33.26	37.14	73.69	52.57

Stage	OCR	Math	VP	APP	Avg.
OCR	33.60	–	–	–	33.60
Math	21.70	34.00	–	–	27.85
VP	28.20	33.20	50.38	–	37.26
APP	24.70	32.10	21.95	30.00	27.19

MoELoRA

Stage	RS	Med	AD	Sci	Fin	Avg.
RS	78.98	–	–	–	–	78.98
Med	76.36	59.22	–	–	–	67.79
AD	76.69	43.05	52.15	–	–	57.30
Sci	74.82	45.14	32.85	48.70	–	50.38
Fin	74.41	46.30	29.40	40.49	87.83	55.69

Stage	OCR	Math	VP	APP	Avg.
OCR	32.80	–	–	–	32.80
Math	30.30	32.80	–	–	31.55
VP	30.70	32.50	67.64	–	43.61
APP	27.70	31.70	51.78	26.50	34.42

CL-MoE

Stage	RS	Med	AD	Sci	Fin	Avg.
RS	79.87	–	–	–	–	79.87
Med	76.69	61.05	–	–	–	68.87
AD	72.67	40.70	52.49	–	–	55.29
Sci	70.29	47.09	39.56	50.24	–	51.80
Fin	71.17	46.74	28.97	40.77	89.84	55.50

Stage	OCR	Math	VP	APP	Avg.
OCR	32.30	–	–	–	32.30
Math	31.60	33.80	–	–	32.70
VP	32.00	34.30	61.29	–	42.53
APP	28.70	33.10	53.30	31.13	36.56

SEFE

Stage	RS	Med	AD	Sci	Fin	Avg.
RS	79.95	–	–	–	–	79.95
Med	74.84	58.69	–	–	–	66.77
AD	77.32	54.76	52.77	–	–	61.62
Sci	74.97	52.87	45.76	51.27	–	56.22
Fin	71.86	52.08	36.71	42.17	90.05	58.57

Stage	OCR	Math	VP	APP	Avg.
OCR	33.00	–	–	–	33.00
Math	32.20	32.60	–	–	32.40
VP	31.80	33.30	64.59	–	43.23
APP	26.00	33.40	57.74	33.75	37.72

CASC

Stage	RS	Med	AD	Sci	Fin	Avg.
RS	80.97	–	–	–	–	80.97
Med	79.12	62.44	–	–	–	70.78
AD	79.05	62.09	40.95	–	–	60.70
Sci	79.33	61.01	40.23	50.84	–	57.85
Fin	**79.35**	**60.45**	**36.54**	**44.45**	**89.51**	**62.06**

Stage	OCR	Math	VP	APP	Avg.
OCR	33.40	–	–	–	33.40
Math	32.30	33.30	–	–	32.80
VP	31.10	33.80	63.20	–	42.70
APP	**32.20**	**29.70**	**60.28**	**36.63**	**39.70**

DISCO

Stage	RS	Med	AD	Sci	Fin	Avg.
RS	79.15	–	–	–	–	79.15
Med	76.07	55.52	–	–	–	65.80
AD	76.36	52.74	52.64	–	–	60.58
Sci	76.15	47.98	50.70	50.37	–	56.30
Fin	76.16	47.75	45.28	43.45	88.86	60.30

Stage	OCR	Math	VP	APP	Avg.
OCR	33.40	–	–	–	33.40
Math	26.80	31.90	–	–	29.35
VP	30.70	32.50	60.79	–	41.33
APP	32.70	31.80	58.76	29.62	38.22

8.8 Qualitative DCL Case Studies

We provide qualitative DCL examples from the final task-agnostic checkpoint. CASC matches the benchmark reference answer in each example. The checkmarks and crosses follow the benchmark scoring rules rather than an independent semantic assessment. In the medical example, string matching distinguishes spelling variants of the stain name.



Q: What is the amount of buildings?
GT: 14
CASC: 14 ✓
CL-MoE: 19 ×
MoELoRA: 23 ×
O-LoRA: 21 ×

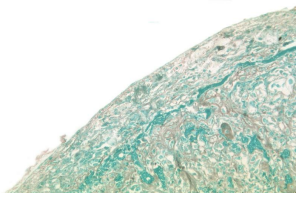


Q: How many small buildings are there?
GT: 1
CASC: 1 ✓
CL-MoE: 2 ×
MoELoRA: 2 ×
O-LoRA: 0 ×

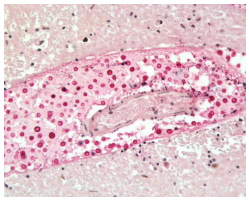


Q: What is the number of medium schools?
GT: 0
CASC: 0 ✓
CL-MoE: 1 ×
MoELoRA: 1 ×
O-LoRA: 1 ×

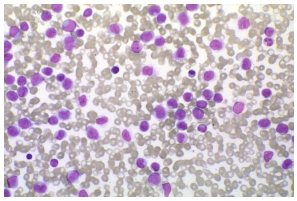
Figure 7: Qualitative Remote Sensing (RS) cases.



Q: what stain?
GT: TRICHROME
CASC: TRICHROME ✓
CL-MoE: GOLD ×
MoELoRA: IRON ×
O-LoRA: NONE ×



Q: what stain?
GT: MUCICARMINE
CASC: MUCICARMINE ✓
CL-MoE: PAS ×
MoELoRA: PAS ×
O-LoRA: PERIODIC ACID-SCHIFF ×



Q: what stain pleomorphic leukemic cells in peripheral blood prior to therapy?
GT: WRIGHTS
CASC: WRIGHTS ✓
CL-MoE: WRIGHT ×
MoELoRA: WRIGHT'S ×
O-LoRA: NO ×

Figure 8: Qualitative Medical Imaging (Med) cases.



Q: Are there barriers to the front left of the ego car?
GT: NO
CASC: NO ✓
CL-MoE: YES ×
MoELoRA: YES ×
O-LoRA: YES ×

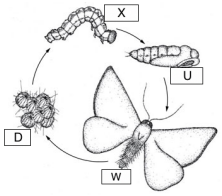


Q: Are there stopped buses to the front of the ego car?
GT: NO
CASC: NO ✓
CL-MoE: YES ×
MoELoRA: YES ×
O-LoRA: YES ×

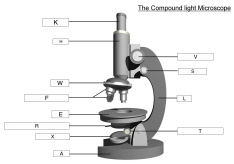


Q: Are there barriers to the front right of the ego car?
GT: NO
CASC: NO ✓
CL-MoE: YES ×
MoELoRA: YES ×
O-LoRA: YES ×

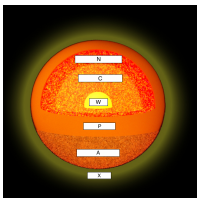
Figure 9: Qualitative Autonomous Driving (AD) cases.



Q: Identify the larva A.U B.X C.W D.D
GT: B: X
CASC: B: X ✓
CL-MoE: A: U ×
MoELoRA: A: U ×
O-LoRA: C: W ×



Q: Identify the stage A.W B.K C.F D.E
GT: D: E
CASC: D: E ✓
CL-MoE: C: F ×
MoELoRA: C: F ×
O-LoRA: C: F ×



Q: How deep is layer W? A.1 B.6 C.4 D.3
GT: C: 4
CASC: C: 4 ✓
CL-MoE: B: 6 ×
MoELoRA: D: 3 ×
O-LoRA: B: 6 ×

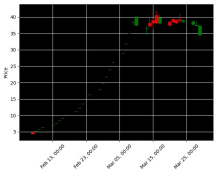
Figure 10: Qualitative Science QA (Sci) cases.



Q: What was the approximate highest price reached during the period shown in the candlestick chart? A. 32.00 B. 30.00 C. 26.0...
GT: B: 30.00
CASC: B: 30.00 ✓
CL-MoE: D: 28.00 ×
MoELoRA: D: 28.00 ×
O-LoRA: D: 28.00 ×



Q: Based on the candlestick chart, what was the highest price reached by the stock during the period shown? A. 8.00 B. 7.98 C...
GT: B: 7.98
CASC: B: 7.98 ✓
CL-MoE: A: 8.00 ×
MoELoRA: D: 7.75 ×
O-LoRA: C: 7.00 ×



Q: Based on the candlestick chart, what was the highest price reached during the period from September 10 to October 30? A. 8...
GT: B: 7.89
CASC: B: 7.89 ✓
CL-MoE: A: 8.00 ×
MoELoRA: D: 7.51 ×
O-LoRA: C: 7.06 ×

Figure 11: Qualitative Finance (Fin) cases.

9 Comparison Method Details

We re-implement all comparison methods under the same DCL and ACL task orders and evaluate them with the same task-agnostic protocol as CASC. RobustMerge is included as an additional LoRA-merging baseline. Because RobustMerge was originally designed for parameter-efficient model merging rather than continual learning, we adapt it to the continual setting by applying its merging procedure to the sequence of task-specific LoRA updates and reporting the final single-checkpoint performance. The row is labeled RobustMerge (adapted) in Table 1. At each task boundary, we merge the accumulated LoRA checkpoint with the next task expert, using `mask_ratio=0.2` and `fuse_weight=2.0`. The mask retains approximately the top 20% of entries by magnitude in each input LoRA tensor, corresponding to approximately 80% pruning. Projector parameters are not pruned and are combined with equal weights at each binary merge, $P^{(t)} = 0.5P^{(t-1)} + 0.5P_t$, starting from $P^{(1)} = P_1$. This cumulative projector average differs from an equal average over all task projectors.

10 Compute Resources

The training setup reported here concerns the main LLaVA-1.5-7B DCL and ACL experiments on a server with 8 NVIDIA A40 GPUs, not the additional backbone experiments in Appendix 8.2. For the DCL benchmark, a full continual run takes approximately 8 hours on this hardware. We use AdamW, batch size 64, cosine learning-rate decay, warmup ratio 0.03, LoRA rank 32, DeepSpeed stage 2, and base learning rate 1×10^{-4}. The maximum number of epochs among DCL tasks is 3.

For the ACL benchmark, a full continual run takes approximately 18 hours on the same 8-GPU A40 server. We use AdamW, batch size 128, cosine learning-rate decay, warmup ratio 0.03, LoRA rank 32, and DeepSpeed stage 2. For LLaVA-1.5 7B, the base learning rate is 5×10^{-5} on OCR and 2×10^{-4} on Math & Logic, VP, and GUI Agent. The maximum number of epochs among ACL tasks is 3.

For the CASC results in Table 1, the training LoRA rank is $r = 32$ and the LoRA bank capacity is $R = 64$. The projector bank capacity is $R_P = 128$ for DCL and $R_P = 384$ for ACL. Both LoRA and projector banks use a protection ratio of 0.5 and a minimum protected rank of 8.

11 Broader Impact

This work studies continual consolidation for multimodal large language models. A single consolidated model may simplify deployment by reducing the need to store or route among task-specific adapters. It may also help retain useful capabilities when historical data cannot be replayed. Consolidation may preserve inherited biases or hallucinations, and benchmark retention does not establish safety for medical, driving, or financial applications. These uses require domain-specific validation and human oversight under the target deployment conditions.

12 Limitations

This work focuses on standard domain and ability continual learning benchmarks. Further evaluation on broader task collections and model families would provide additional evidence for generality. We leave this extension to future work.

13 Existing Assets and Licenses

This paper uses existing models, benchmarks, and comparison methods only for research evaluation, and credits the original sources through citations in the main text. LLaVA-1.5 7B is used as the backbone MLLM. Its model card states that the underlying Llama 2 model is governed by the LLaMA 2 Community License. The additional Qwen3-VL-8B-Instruct model is released under Apache 2.0, as specified in the model card linked in Appendix 8.2. MLLM-CL and MCITlib are used for benchmark task definitions, data processing, and evaluation protocols, including the additional UCIT sequence. We follow their public usage terms and do not redistribute their datasets, model weights, or source code as part of this submission. Baseline methods are cited in Sec. 4, and our comparisons use research re-implementations under the same evaluation protocol.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction state the paper's scope as unified multimodal consolidation for continual MLLM adaptation, and the claimed contributions are CASC, joint LoRA-projector consolidation, and empirical gains on DCL and ACL. The experimental section reports the corresponding metrics and comparisons.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The paper discusses practical limitations related to broader task collections in Appendix 12.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [Yes]

Justification: The paper provides the proof in Appendix 6.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: The method section specifies the spectral bank construction and consolidation rule, while the experiments specify the benchmark, task order, metrics, baselines, and LLaVA-1.5 based setting. The appendix provides implementation details, detailed matrices, ablations, and hyperparameter sensitivity.

Guidelines:

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

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: The code is provided for review. The experiments use the open-source datasets provided by the MLLM-CL benchmark, and the paper follows the corresponding benchmark protocol for data access, preparation, and evaluation.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Section 4 describes the DCL and ACL benchmarks, comparison methods, metrics, backbone, and benchmark protocol. Appendix 8 provides detailed result matrices, ablations, and hyperparameter sweeps.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: Appendix 8.1 reports means and standard deviations across three training seeds for CASC, DISCO, SEFE, and CL-MoE on DCL. Table 1 is explicitly labeled as single-run results.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Appendix 10 reports the 8-GPU NVIDIA A40 setup, approximate wall-clock time for DCL and ACL runs, and the main training configuration used to reproduce the experiments.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The work studies algorithmic consolidation for existing MLLM continual learning benchmarks and does not involve human subjects, private data collection, or deployment decisions. We are not aware of any deviation from the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

Answer: [Yes]

Justification: Appendix 11 discusses deployment benefits, potential inherited biases and hallucinations, and the need for domain-specific validation and human oversight.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

Answer: [N/A]

Justification: The paper does not release a new dataset, deployed model, or high-risk generative system. It evaluates a consolidation algorithm on existing benchmarks.

Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: The paper credits the existing backbones, benchmarks, and baseline methods through citations. Appendix 13 summarizes the main external assets and the known license or usage terms for LLaVA-1.5 7B, Qwen3-VL-8B-Instruct, MLLM-CL, and MCITlib.

Guidelines:

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

13. **New assets**

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

    Answer: [Yes]

    Justification: The CASC code repository includes running instructions, environment dependencies, and a license. The paper introduces no new dataset or benchmark.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing experiments or research with human subjects.

    Guidelines:

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

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

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

Answer: [N/A]

Justification: The paper does not involve crowdsourcing, user studies, or human-subject experiments, so IRB approval is not applicable.

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: LLMs were not used as an important, original, or non-standard component of the proposed method. Any use of LLM-based tools was limited to writing, editing, formatting, or coding assistance and did not affect the scientific claims, experimental results, or originality of the method.

Guidelines:

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