Is Our Benchmark Enough? An Analysis of Continual Learning for MLLMs

Van-Tuan Tran [1] **Shruthi Gowda** [2] **Merim Dzaferagic** [1] **Marco Ruffini** [1]

Abstract

Continual adaptation is essential for multimodal large language models (MLLMs) deployed across evolving domains, but the state-of-the-art MR-LoRA method highly relies on the assumption that a MLLM-based router is necessary to process complex multimodal inputs. This paper revisits this claim on the MLLM-CL benchmark and argues for two claims. **First**, routing does not require an MLLM: a simple training-free, replay-free ptotypical routing method (REPRO), uses frozen pretrained features and task prototypes to match the MLLM-based router of MR-LoRA at far lower computational cost. **Second**, shared experts do not improve continual learning for MLLMs, despite their theoretical appeal. We show that these findings arise from two structural limitations of MLLM-CL: (1) its tasks are **highly separable** in representation space, and (2) its fixed task order makes conclusions **sensitive to a single curriculum** rather than robust across diverse continual-learning trajectories. As a result, the benchmark primarily rewards learning in isolation rather than genuine continual transfer. This motivates a new design for future benchmarks of continual MLLM learning, with overlapping task manifolds, multiple task orders, fine-grained domain shifts, and evaluation protocols that reward forward transfer as well as retention.

1. Introduction

Multimodal Large Language Models (MLLMs) (Liu et al., 2023; 2024a; Chen et al., 2024b) now serve as the predominant backbone for vision–language applications. Real-world deployment, however, requires ongoing adaptation to new domains, abilities, and data streams under strict computational budgets: retraining from scratch at every update is infeasible at the scale of modern MLLMs, and sequential fine-tuning suffers from catastrophic forgetting on previously learned tasks (McCloskey & Cohen, 1989; Kirkpatrick et al., 2017). These constraints make ***continual learning*** (CL) a practical necessity rather than a purely theoretical concern. To support systematic study of this problem, Zhao et al. (2025) recently introduced MLLM-CL as a comprehensive CL benchmark for MLLMs. Within this benchmark, MR-LoRA (Zhao et al., 2025) achieves strong performance by assigning each task an isolated LoRA expert and using an MLLM-based router to select the expert at inference time. While effective, this design raises a fundamental question: *is the MLLM-based router truly necessary, or does the benchmark mainly require identifying which task an input belongs to?*

In this work, we take a position by defending two claims. ***(1) Routing does not require an MLLM***. We propose Replay-Free Prototypical Routing (REPRO), a parameter-free router that uses frozen pretrained representations and task prototypes to select among isolated LoRA experts. Despite its simplicity, REPRO matches the MLLM-based router of MR-LoRA while greatly reducing inference cost. This suggests that, on MLLM-CL, expert selection can be solved without MLLM. ***(2) Shared experts do not improve continual learning for MLLMs.*** Although shared Mixture-of-Experts (MoE) designs are motivated by the idea that related tasks should benefit from shared parameters, our analysis shows that shared experts consistently underperform isolated experts on MLLM-CL. Their training losses converge normally, indicating that the issue is not simply an optimization failure. Rather, the benchmark provides limited transferable structure for shared experts to exploit.

We argue that these two findings reveal **a broader benchmark limitation**. The tasks in MLLM-CL are highly separable in representation space, making routing nearly trivial, while the use of a single canonical task order makes conclusions sensitive to one curriculum. As a result, the benchmark rewards isolated tasks aptation more than genuine continual transfer. This motivates the next generation of continual MLLM benchmarks: they should include overlapping task manifolds, fine-grained domain shifts, multiple task orders, and evaluation protocols that measure forward transfer as well as retention.

[1]School of Computer Science and Statistics, Trinity College Dublin, Ireland [2]Department of Mathematics and Computer Science, Eindhoven University of Technology, The Netherlands. Correspondence to: Van-Tuan Tran <tranva@tcd.ie>.

Presented at the ICML 2026 Workshop "Continual Adaptation at Scale: Towards Sustainable AI". Copyright 2026 by the author(s).

Table 1. Task accuracy on the Domain Continual Learning of MLLM-CL benchmark. All trainable methods use LoRA rank $r = 8$ for fair comparison. Both REPRO variants consistently match or exceed the computationally expensive MR-LoRA oracle routing.

Method	Details	Task 1 (RS)	Task 2 (Med)	Task 3 (AD)	Task 4 (Sci)	Task 5 (Fin)	Avg. Acc.
Zero-shot	No fine-tuning	32.28	28.28	15.58	43.21	62.57	36.20
Baselines							
CL-MoE (Huai et al., 2025)	MoE of LoRAs with dual momentum router	69.43	34.07	31.13	38.16	86.01	51.76
O-LoRA (Wang et al., 2023)	Orthogonal subspace LoRA	21.97	28.20	14.19	35.48	58.67	31.70
DISCO-LoRA (Guo et al., 2025b)	Disentangled shared/task-specific LoRA	25.50	28.75	18.70	36.04	59.65	33.73
HiDe-LoRA (Guo et al., 2025a)	Hierarchical decomposition of LoRA	28.70	26.00	18.60	36.84	52.00	32.43
MoELoRA (Liu et al., 2024c)	Mixture of LoRA experts with token-level soft routing	75.23	39.79	36.61	41.47	90.83	56.79
SEFE (Chen et al., 2025)	Regularized LoRA against semantic forgetting	74.38	44.60	40.24	44.41	91.47	59.02
Sequential FT	Naive sequential LoRA fine-tuning	76.70	54.60	42.65	42.71	91.85	61.70
EWC (Kirkpatrick et al., 2017)	Elastic Weight Consolidation regularization	77.60	44.90	32.15	40.53	90.90	57.22
Experience Replay (Chaudhry et al., 2019)	Rehearsal with stored past samples	76.50	54.75	43.85	43.67	91.65	62.08
MR-LoRA (Zhao et al., 2025)	Oracle routing, isolated experts	74.20	65.00	55.80	53.67	91.40	68.01
Ours							
REPRO	Prototypical routing, Isolated experts, Signal: Vision	73.80	64.60	55.80	53.67	91.40	67.85
MULTIMODAL-REPRO	Prototypical Routing, Isolated experts, Signal: Multimodal	75.60	64.70	55.80	53.77	91.40	**68.25**

Contribution. We (1) reduce the MR-LoRA routing pipeline to a prototype router that matches its accuracy with eight orders of magnitude fewer FLOPs and three orders lower latency, undermining the assumption that multimodal routing requires an MLLM; (2) report a systematic negative result for shared-expert routing across representative design points, motivating new sharing mechanisms for continual MLLM adaptation; (3) tie both findings to a single cause: the near-linear separability of MLLM-CL tasks (visualized in Fig. 1); and (4) outline concrete desiderata for the next generation of continual MLLM benchmarks, including overlapping task manifolds, fine-grained specialization under a shared capability, and evaluation protocols that reward forward transfer rather than merely penalizing forgetting.

2. Isolated Experts: Are They Really Enough?

2.1. Routing: Does isolation require an MLLM-based router?

From Sec.A, MR-LoRA relies on a heavyweight MLLM-based router tuned on a replay buffer, under the premise that a full MLLM is needed to infer task identity from an image-instruction pair. We challenge this assumption by hypothesizing that if the underlying tasks are conceptually distinct, *their representations in pre-trained foundational spaces are inherently separable* without requiring learned autoregressive routing. To demonstrate this, we propose **Re**play-Free **Pro**totypical Routing (REPRO), a replay-free, training-free routing methodology, and its MULTIMODAL-REPRO variant.

Specifically, after each isolated expert θ_k is trained on D_k, we sample a small prototype support set S_k of $n = 128$ examples per task, extract the `[CLS]` token from the frozen CLIP vision encoder $\phi_v(\cdot)$, and compute an ℓ_2-normalised centroid:

$$\bar{c}_k^v = \ell_2\Big(\frac{1}{n}\sum_{x\in S_k}\ell_2\big(\phi_v(x)\big)\Big). \tag{1}$$

At inference, x is routed to $k^\star(x) = \arg\max_k \cos\big(\phi_v(x), \bar{c}_k^v\big)$. The MULTIMODAL-REPRO variant computes parallel text prototypes $\bar{c}_k^\ell$ from the mean-pooled hidden states of the frozen LLM backbone over the instruction q, and fuses modalities by $k^\star(x,q) = \arg\max_k \max\big(\cos(\phi_v(x), \bar{c}_k^v), \cos(\phi_\ell(q), \bar{c}_k^\ell)\big)$. Detailed algorithmss and additional results are in App. C.

Tab. 1 compares REPRO against MR-LoRA's MLLM router and all CL baselines on DCL. *(1) The MLLM router is conceptually redundant.* Replacing it with REPRO matches MR-LoRA's accuracy (67.85 vs. 68.01), and MULTIMODAL-REPRO slightly exceeds it (68.25). *(2) Routing cost drops by orders of magnitude.* As shown in Tab. 5 (App. C.4.3), MLLM-based router requires $\sim 1.3\times10^{13}$ FLOPs and $\sim$72 ms per decision, while REPRO operates at 4.1×10^4 FLOPs and $\sim$0.06 ms. This means that using a router with *zero* trainable parameters can reduce 8 orders of magnitude in FLOPs and 3 in latency.

These results indicate that reliance on an MLLM router is *an over-engineered solution* to a highly separable representation problem. Future gains under the isolated-expert paradigm must come from improving experts, not routers.

2.2. Sharing: Can shared experts improve over isolation?

The foundational premise of Mixture-of-Experts (MoE) and parameter-efficient fine-tuning in continual learning is that shared parameters encode shared structure (Dou et al., 2024; Guo et al., 2025b). Given that strict isolation explicitly prohibits this parameter-level co-adaptation, we ask a concrete question: *Can we design a shared-expert architecture*

Table 2. Task accuracy and transfer metrics for the proposed **Mixture of Shared Experts (MoSE)** framework on MLLM-CL benchmark. Despite architectural sophistication and various auxiliary supervisions, shared experts consistently plateau around ∼55%, failing to match the isolated expert baseline (∼68%).

Routing level	Details			Task 1	Task 2	Task 3	Task 4	Task 5	Avg.	BWT	FWT
	Gating	Experts	Signal	(RS)	(Med)	(AD)	(Sci)	(Fin)	Acc.		
Sample-level	Softmax, noisy gating	5 shared experts	Multimodal	73.70	39.70	33.90	37.73	87.70	**54.54**	−7.52	1.34
	Softmax, top-k	5 shared experts, $k = 2$	Multimodal	72.70	34.90	21.30	35.35	84.70	**49.79**	−9.45	−2.27
	Softmax, top-k, task-guided	5 shared experts, $k = 2$	Multimodal	76.50	39.30	35.00	38.64	87.00	**55.28**	−6.52	1.43
	Softmax, top-k, task-guided, orthogonal loss	5 shared experts, $k = 2$	Multimodal	73.70	38.40	32.00	37.73	87.70	**53.90**	−8.23	2.88
Token-level	Sigmoid	5 shared experts, soft gating	Multimodal	75.60	38.20	33.90	38.90	88.10	**54.94**	−7.83	0.89
	Sigmoid, task-guided	5 shared experts, soft gating	Multimodal	76.00	39.60	35.20	38.61	87.20	**55.32**	−5.88	−0.80

that extracts cross-task synergies and outperforms strict isolation on MLLM-CL?

MoSE: Mixture of Shared Experts. MoSE evaluates two representative parameterized-routing paradigms on top of $K = 5$ shared experts. *(1) Sample-level global routing* pools sequence hidden states of both vision and language features, with learnable fusion weights, then applies a noisy softmax gate. *(2) Token-level local routing* operates on individual tokens with a sigmoid-gated, $\alpha_{t,i} = \sigma\big(\beta(s_{t,i} - \tau_i)\big)$, parameterised by learnable per-expert thresholds τ and a learnable sharpness β. We also ablate auxiliary supervisions (task-guided cross-entropy, orthogonality loss) and sparsity constraints (top-k). Full algorithmic formulations and initialisation schemes are in App. D.

Empirical Results and Insights. Despite extensive architectural tuning and the theoretical motivation of MoE, ***no shared-expert variant outperforms strict isolation***. Tab. 2 shows that neither paradigm surpasses the mid-50% range, with the best shared variant (token-level sigmoid with task supervision) reaching only 55.32%. From these results, three diagnostic patterns emerge: **(1) Hard sparsification drives forgetting** – sample-level top-k without task guidance yields the lowest accuracy (49.79%) and the worst BWT (−9.45%); **(2) Task supervision helps but cannot close the gap** – task-guided cross-entropy raises sample-level top-k from 49.79% to 55.28% and improves token-level BWT from −7.83 to −5.88, yet shared architectures remain bottlenecked even when routing is supervised to be near-perfect; **(3) Task orthogonality trades retention for transfer** – adding a task-orthogonality penalty to the top-k task-guided router successfully broke early routing symmetry, yielding the highest FWT in the study (+2.88%), but it hurts retention (BWT drops to −8.23). To rule out the possibility that the observed underperformance is simply due to a failure to learn, we report the training-loss curves in Fig. 2 in the Appendix. These curves show that the models converge with nearly identical per-task loss profiles.

3. The Root Cause Is the Benchmark, Not the Method

To understand the underperformance of shared-expert architectures (Sec. 2.2), the surprising efficacy of training-free routing (Sec. 2.1), we study the properties of MLLM-CL. Our analysis reveals that *the failure lies not in the learning algorithms, but in the structural properties of the benchmark itself.* Specifically, we identify two critical artifacts of MLLM-CL that inherently penalize shared parameterization and confound evaluation.

3.1. Disjoint Task Manifolds Preclude Knowledge Transfer

The foundation of parameter-efficient sharing in continual learning is that related tasks can synergistically co-activate subsets of parameters (Dou et al., 2024; Feng et al., 2024; Huai et al., 2025). However, these studies assume that the sequential tasks actually exhibit an overlapping representational manifold.

As illustrated in the t-SNE projections (Fig. 1), this assumption breaks down entirely in the MLLM-CL sequence. When projected into pre-trained foundational spaces, using either a frozen visual encoder (i.e., CLIP `CLS` tokens) or a textual encoder (i.e., LLaMA (Touvron et al., 2023) mean-pooled hidden states), 5 DCL domains form perfectly isolated, highly dense clusters. Quantitatively (Tab. 5 in Appendix), purely textual signals route at 97.4%, purely visual at 99.5%, and the multimodal router reaches 99.9%.

This level of separation has two immediate consequences. **(i) The redundancy of complex routing.** Because the representations are cleanly disjoint prior to any task-specific adaptation, a simple router such as MULTIMODAL-REPRO can achieve near-oracle expert selection. This explains Sec. 2.1. **(ii) Shared-experts have nothing to share.** Parameter sharing helps only when tasks possess overlapping semantic structure that experts can jointly exploit. In this setting, shared low-rank experts are encouraged to fit domain-specific updates rather than reusable cross-domain factors,

so later task updates can interfere with parameters needed for earlier tasks instead of producing positive transfer.

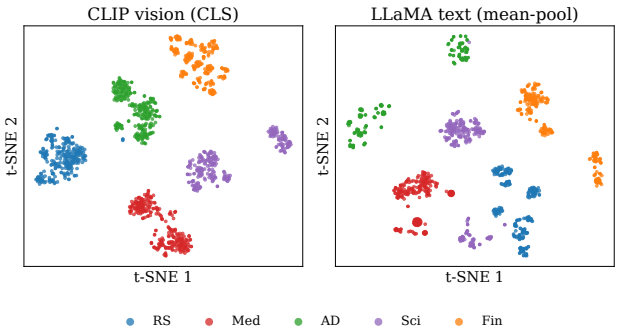

Figure 1. **The MLLM-CL benchmark is representationally too separate.** t-SNE projections of the five DCL domains under a frozen CLIP visual encoder (left, `[CLS]` token) and a frozen LLaMA text encoder (right, mean-pooled hidden states).

3.2. Task-Order Sensitivity

A second observation is that performance on the canonical DCL sequence is markedly non-uniform across positions. In both Tabs. 1–2, every method peaks on Task 1 (RS) and Task 5 (Fin) and bottoms out on the interior positions Tasks 2–4 (Med, AD, Sci). We hypothesize that this order affects the performance of shared experts.

To evaluate our hypothesis, we re-evaluated shared-expert architectures across multiple task permutations. To ensure these permutations are principled, we anchor them on the inherent difficulty of each domain, quantified by the zero-shot accuracy of the unadapted base model (LLaVA-1.5 (Liu et al., 2024a)). This establishes a strict difficulty gradient from the hardest to the easiest domain: Autonomous Driving (AD, 15.58%) $<$ Medical (Med, 28.28%) $<$ Remote Sensing (RS, 32.28%) $<$ Science (Sci, 42.31%) $<$ Finance (Fin, 62.57%). Across the complete four-order ablation, all evaluated shared-pool MoE tuners follow the same order ranking: `easy-to-hard` $>$ `random` $>$ `canonical` $\geq$ `hard-to-easy` (Details in Tab. 6). As shown in Tab. 3, the resulting order spread is large (6.36–8.21 pp) and exceeds the largest between-method spread within any fixed order (4.73 pp), showing that the DCL order can dominate the choice of gating function.

4. Desiderata for the Next CL Benchmark for MLLMs

Our results establish three constraints for the next continual MLLM benchmark. **First**, trivial routing must not be the hidden solution: on MLLM-CL, frozen CLIP/LLaMA features route domains with more than 99% accuracy, and a nearest-prototype router matches an MLLM router. **Second**, sharing cannot be judged on tasks with no shared manifold: all shared-expert MoE variants converge normally yet plateau far below isolated experts. **Third**, a single canonical order is insufficient: changing the DCL order drastically affects shared-expert MoE accuracy, even exceeding the difference between gating designs.

Table 3. Mean accuracy (%) across the five DCL domains, $\bar{a}_5(m, \pi)$, evaluated after the final task under each task order. Δ_{order} reports the spread between the best and worst order for each method. Best per method in **bold**.

Task order	CL-MoE	MoELoRA	MoSE-sigmoid
`canonical`	51.77	47.04	49.70
`hard-to-easy`	48.92	46.15	47.81
`easy-to-hard`	**55.28**	**54.36**	**55.04**
`random`	50.52	48.98	51.20
Δ_{order}	6.36	8.21	7.23

A benchmark should therefore formalize the hard part of CL: sparse, sequential exposure to shifted data, without reducing the problem to simultaneous multi-task training or large-scale replay. This requires streams where task identity is not linearly recoverable from frozen representations, but where tasks still share enough visual concepts, instructions, output formats, or latent rules for positive transfer to be possible. It should include recurring and compositional tasks, distinguish domain shift from capability shift, and evaluate multiple task orders calibrated by the zero-shot base model. Strong methods should retain old skills, acquire new ones, selectively overwrite stale knowledge, and improve on related future tasks without explicit task labels, unbounded expert growth, or full replay.

5. Conclusion

This paper shows that continual learning on MLLM-CL can be solved more simply than current routing-heavy designs suggest. REPRO replaces the MLLM-based router with replay-free prototype routing over frozen features, achieving comparable performance at far lower cost. We also find that shared experts fail to outperform isolated experts, not because they cannot learn, but because the benchmark provides limited transferable structure. Together, these results reveal two limitations of MLLM-CL: highly separable task representations and reliance on a single task order. Future benchmarks should include overlapping task manifolds, multiple curricula, and metrics that reward genuine transfer as well as retention.

Acknowledgements

The "MoSE: Mixture-of-Specialized-Experts for efficient continual adaptation" action has received funding from the European Union, via the oc3-2025-TES-01 issued and implemented by the ENFIELD project, under the grant agreement No 101120657.

References

Cao, J., Lin, T., He, H., Yan, R., Zhang, W., Li, J., Zhang, D., Tang, S., and Zhuang, Y. MoA: Heterogeneous mixture of adapters for parameter-efficient fine-tuning of large language models. *arXiv preprint arXiv:2506.05928*, 2025.

Chaudhry, A., Rohrbach, M., Elhoseiny, M., Ajanthan, T., Dokania, P. K., Torr, P. H. S., and Ranzato, M. On tiny episodic memories in continual learning. In *ICML Workshop on Multi-Task and Lifelong Reinforcement Learning*, 2019.

Chen, C., Zhu, J., Luo, X., Shen, H., Gao, L., and Song, J. CoIN: A benchmark of continual instruction tuning for multimodal large language models. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2024a.

Chen, J., Cong, R., Zhao, Y., Yang, H., Hu, G., Ip, H. H. S., and Kwong, S. SEFE: Superficial and essential forgetting eliminator for multimodal continual instruction tuning. In *International Conference on Machine Learning (ICML)*, 2025.

Chen, Z., Wu, J., Wang, W., Su, W., Chen, G., Xing, S., Zhong, M., Zhang, Q., Zhu, X., Lu, L., 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 (CVPR)*, pp. 24185–24198, 2024b.

Dou, S., Zhou, E., Liu, Y., Gao, S., Shen, W., Xiong, L., Zhou, Y., Wang, X., Xi, Z., Fan, X., Pu, S., Zhu, J., Zheng, R., Gui, T., Zhang, Q., and Huang, X. 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 (ACL)*, pp. 1932–1945, 2024.

Feng, W., Hao, C., Zhang, Y., Han, Y., and Wang, H. Mixture-of-LoRAs: An efficient multitask tuning method for large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING)*, pp. 11371–11380, 2024.

Ge, C., Wang, X., Zhang, Z., Chen, H., Fan, J., Huang, L., Xue, H., and Zhu, W. D-MoLE: Dynamic mixture of curriculum LoRA experts for continual multimodal instruction tuning. In *International Conference on Machine Learning (ICML)*, 2025.

Gou, Y., Liu, Z., Chen, K., Hong, L., Xu, H., Li, A., Yeung, D.-Y., Kwok, J. T., and Zhang, Y. Mixture of cluster-conditional LoRA experts for vision-language instruction tuning. *arXiv preprint arXiv:2312.12379*, 2023.

Guo, H., Zeng, F., Xiang, Z., Zhu, F., Wang, D.-H., Zhang, X.-Y., and Liu, C.-L. HiDe-LLaVA: Hierarchical decoupling for continual instruction tuning of multimodal large language model. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13572–13586, 2025a.

Guo, H., Zhu, F., Zhao, H., Zeng, F., Liu, W., Ma, S., Wang, D.-H., and Zhang, X.-Y. MCITlib: Multimodal continual instruction tuning library and benchmark. *arXiv preprint arXiv:2508.07307*, 2025b. Code: https://github.com/Ghy0501/MCITlib.

He, J., Duan, Z., and Zhu, F. CL-LoRA: Continual low-rank adaptation for rehearsal-free class-incremental learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

Huai, T., Zhou, J., Wu, X., Chen, Q., Bai, Q., Zhou, Z., and He, L. 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 (CVPR)*, pp. 19608–19617, 2025.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., and Hadsell, R. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

Kunwar, P., Vu, M. N., Gupta, M., Abdelsalam, M., and Bhattarai, M. TT-LoRA MoE: Unifying parameter-efficient fine-tuning and sparse mixture-of-experts. *arXiv preprint arXiv:2504.21190*, 2025.

Li, D., Ma, Y., Wang, N., Ye, Z., Cheng, Z., Tang, Y., Zhang, Y., Duan, L., Zuo, J., Yang, C., and Tang, M. MixLoRA: Enhancing large language models fine-tuning with LoRA-based mixture of experts. *arXiv preprint arXiv:2404.15159*, 2024.

Li, Y., Jiang, S., Hu, B., Wang, L., Zhong, W., Luo, W., Ma, L., and Zhang, M. Uni-MoE: Scaling unified multimodal LLMs with mixture of experts. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 47 (5):3424–3439, 2025.

Liang, Y. and Li, W.-J. InfLoRA: Interference-free low-rank adaptation for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Liao, M., Chen, W., Shen, J., Guo, S., and Wan, H. HMoRA: Making LLMs more effective with hierarchical mixture of LoRA experts. In *International Conference on Learning Representations (ICLR)*, 2025.

Liu, H., Li, C., Wu, Q., and Lee, Y. J. Visual instruction tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Liu, H., Li, C., Li, Y., and Lee, Y. J. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26296–26306, 2024a.

Liu, J., Wu, J., Liu, J., and Duan, Y. Learning attentional mixture of LoRAs for language model continual learning. *arXiv preprint arXiv:2409.19611*, 2024b.

Liu, Q., Wu, X., Zhao, X., Zhu, Y., Xu, D., Tian, F., and Zheng, Y. When MoE meets LLMs: Parameter efficient fine-tuning for multi-task medical applications. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2024c.

Liu, W., Zhu, F., Wei, L., and Tian, Q. C-CLIP: Multimodal continual learning for vision-language model. In *International Conference on Learning Representations (ICLR)*, 2025.

Liu, X. and Chang, X. LoRA subtraction for drift-resistant space in exemplar-free continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Liu, Z. and Luo, J. AdaMoLE: Fine-tuning large language models with adaptive mixture of low-rank adaptation experts. In *Conference on Language Modeling (COLM)*, 2024.

McCloskey, M. and Cohen, N. J. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*, volume 24, pp. 109–165. Elsevier, 1989.

Rebuffi, S.-A., Kolesnikov, A., Sperl, G., and Lampert, C. H. iCaRL: Incremental classifier and representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Srinivasan, T., Chang, T.-Y., Pinto Alva, L., Chochlakis, G., Rostami, M., and Thomason, J. CLiMB: A continual learning benchmark for vision-and-language tasks. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, volume 35, pp. 29440–29453, 2022.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. LLaMA 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Wang, X., Chen, T., Ge, Q., Xia, H., Bao, R., Zheng, R., Zhang, Q., Gui, T., and Huang, X. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP*, 2023.

Wei, X., Li, G., and Marculescu, R. Online-LoRA: Task-free online continual learning via low-rank adaptation. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2025.

Wu, X., Huang, S., and Wei, F. Mixture of LoRA experts. In *International Conference on Learning Representations (ICLR)*, 2024.

Wu, Y., Piao, H., Huang, L.-K., Wang, R., Li, W., Pfister, H., Meng, D., Ma, K., and Wei, Y. SD-LoRA: Scalable decoupled low-rank adaptation for class incremental learning. In *International Conference on Learning Representations (ICLR)*, 2025.

Zhang, X., Zhang, F., and Xu, C. VQACL: A novel visual question answering continual learning setting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19102–19112, 2023.

Zhao, H., Zhu, F., Guo, H., Wang, M., Wang, R., Meng, G., and Zhang, Z. MLLM-CL: Continual learning for multimodal large language models. *arXiv preprint arXiv:2506.05453*, 2025.

Zhu, Y., Wichers, N., Lin, C.-C., Wang, X., Chen, T., Shu, L., Lu, H., Liu, C., Luo, L., Chen, J., and Meng, L. SiRA: Sparse mixture of low rank adaptation. *arXiv preprint arXiv:2311.09179*, 2023.

A. Background

This section establishes the formal setting and the technical building blocks used throughout the paper. We first define continual learning for MLLMs, then describe the MLLM-CL benchmark that all our experiments use, and finally summarize the MR-LoRA pipeline that our analysis takes as its reference design.

A.1. Continual Learning for MLLMs

Following Zhao et al. (2025), an MLLM is a function f_θ that maps an image–instruction pair $(x^{\text{img}}, x^{\text{ins}})$ to an answer y of L tokens, trained autoregressively. In the continual setting the model observes a sequence of T tasks $\{\mathcal{D}_t\}_{t=1}^{T}$, where each $\mathcal{D}_t = \{(x_{t,i}^{\text{img}}, x_{t,i}^{\text{ins}}, y_{t,i})\}_{i=1}^{N_t}$ is drawn IID from a task-specific distribution $\mathcal{P}_t = \mathcal{X}_t^{\text{img}} \times \mathcal{X}_t^{\text{ins}} \times \mathcal{Y}_t$. At stage t only $\mathcal{D}_t$ is available; data from tasks $1, \ldots, t-1$ cannot be revisited at full scale. The training loss at stage t is the standard next-token objective,

$$\mathcal{L}_t(\theta) = -\sum_{i=1}^{N_t}\sum_{l=1}^{L} \log p_\theta\Big(y_{t,i}^{l} \mid x_{t,i}^{\text{img}},\, x_{t,i}^{\text{ins}},\, y_{t,i}^{<l}\Big). \quad (2)$$

After the full sequence has been seen, the model is evaluated on test inputs drawn from any of $\{\mathcal{P}_j\}_{j=1}^{T}$. The two failure modes of interest are *catastrophic forgetting*, the degradation on tasks $1, \ldots, t-1$ after training on $\mathcal{D}_t$, and *loss of plasticity*, the failure to acquire $\mathcal{D}_t$ itself due to interference with prior parameters. Performance is reported with the standard CL metrics: Average accuracy after the final task, forward transfer (FWT), and backward transfer (BWT).

A.2. MLLM-CL Benchmark

Several vision–language continual-learning benchmarks precede MLLM-CL, but they target different model scales and problem formulations. CLiMB (Srinivasan et al., 2022) evaluates multimodal task-to-task learning and downstream low-shot transfer with ViLT-style encoders and classification-oriented heads. VQACL (Zhang et al., 2023) focuses on VQA continual learning, organizing tasks by question type and visual category to test novel skill–concept compositions. CoIN (Chen et al., 2024a) moves toward continual instruction tuning for MLLMs, but it only benchmarks the forgetting aspect of MLLM methods across a chain of instruction-tuning datasets. We therefore use MLLM-CL (Zhao et al., 2025) as the single testbed for this paper because it best matches our target setting: benchmarking MLLM's ability to obtain new domains/abilities and standardized continual-learning metrics for evaluation.

MLLM-CL comprises two regimes: Domain Continual Learning (DCL), which measures absorption of domain-specific knowledge under IID train/test splits, and Ability Continual Learning (ACL), which targets fundamental capabilities (OCR, math & logic, visual perception, GUI agent) under non-IID splits.

We restrict our analysis to DCL because two ACL tasks lack deterministic ground truth: math & logic labels are produced by an LLM-as-judge, and GUI agent answers are scored by an AI-based grading platform. Both inject a stochastic, model-driven component into the evaluation, which would confound the controlled comparisons our position paper claims rely on.

DCL comprises five tasks: remote sensing (RS), medical (Med), autonomous driving (AD), science (Sci), and finance (Fin) — presented in the fixed order:

$$\text{RS} \rightarrow \text{Med} \rightarrow \text{AD} \rightarrow \text{Sci} \rightarrow \text{Fin},$$

with roughly 60k training and 10k test examples per task, IID within each task.

A.3. MR-LoRA: Isolated Experts with an MLLM Router

MR-LoRA (Zhao et al., 2025) is built on Low-Rank Adaptation (LoRA; Hu et al., 2022), which freezes a pretrained weight W_0 and adds a low-rank update $\Delta W = BA$ with $r \ll \min(d, k)$, training only A and B. MR-LoRA combines LoRA with strict per-task isolation and a generative router.

Isolated expert training. For each task t, MR-LoRA trains a dedicated LoRA ϕ_t on $\mathcal{D}_t$ from the frozen MLLM backbone. Experts are mutually non-interacting: ϕ_t never sees $\mathcal{D}_{<t}$, and $\phi_{<t}$ are never updated when $\mathcal{D}_t$ arrives. By construction, this eliminates parameter-level forgetting, since the past experts are bit-identical at the end of training as they were when frozen. The cost is that some mechanism must decide, at inference, *which* expert to apply to a given input.

MLLM-based routing. MR-LoRA treats expert selection as a generation task. After each stage, it collects a small replay buffer $\mathcal{M}_t = \{(x_{t,i}^{\text{img}}, x_{t,i}^{\text{ins}})\}_{i=1}^{m}$ with $m = 20 \ll N_t$ and finetunes a separate *router LoRA* ϕ_R on $\bigcup_{j \le t} \mathcal{M}_j$ using a structured prompt that lists all expert descriptions. Given a test input, the full MLLM with ϕ_R runs a forward pass over the prompt and image and autoregressively emits a single token: the identifier of the chosen expert. The selected $\phi_{\hat{t}}$ is then swapped in to produce the final answer in a second forward pass.

Two properties of this design matter for the rest of the paper. First, every test query incurs *two* MLLM forward passes, one to route and one to answer, even though the routing pass produces only one token. Second, the experts are entirely decoupled at training time: the architecture admits no cross-

task gradient flow and no parameter sharing beyond the frozen backbone.

B. Related Work

CL for MLLMs. CoIN (Chen et al., 2024a) and MLLM-CL (Zhao et al., 2025) established continual instruction tuning benchmarks for MLLMs. Comes with MLLM-CL is the MR-LoRA baseline, which allocates one fresh LoRA expert per task to avoid parameter interference. The price of this isolation is a replay-trained Router LoRA that runs the full MLLM to generate an expert identifier for every test query. Other multimodal adaptation methods use different forms of expert or representation management: MoCLE (Gou et al., 2023) clusters instructions offline and activates cluster-conditional LoRA experts with a universal expert for generalization; D-MoLE (Ge et al., 2025) dynamically allocates layer-wise LoRA experts and uses a modality curriculum for continual multimodal instruction tuning; and C-CLIP (Liu et al., 2025) studies continual learning for CLIP-style image–text matching while preserving zero-shot ability.

MoE-LoRA with learned routers. LoRA-based MoE and adaptive-adapter families include LoRAMoE (Dou et al., 2024), MoELoRA (Liu et al., 2024c), MixLoRA (Li et al., 2024), Mixture-of-LoRAs (Feng et al., 2024), Uni-MoE (Li et al., 2025), SiRA (Zhu et al., 2023), MoLE (Wu et al., 2024), MoA (Cao et al., 2025), HMoRA (Liao et al., 2025), AdaMoLE (Liu & Luo, 2024), AM-LoRA (Liu et al., 2024b), and Online-LoRA (Wei et al., 2025). Most MoE-style variants assume a fixed expert pool and train a softmax, top-k, attentional, or thresholded router, often with load-balancing, contrastive, or curriculum to prevent expert collapse. TT-LoRA-MoE (Kunwar et al., 2025) is architecturally closest to the isolated-expert side of our analysis because it trains specialized low-rank experts independently and then freezes them; however, its router is a learned top-1 gate, the task set is predefined before router training, and its evaluation is text-only rather than continual MLLM routing.

Isolation, orthogonality, and prototypes. Parameter-level non-interference is enforced by O-LoRA (Wang et al., 2023), InfLoRA (Liang & Li, 2024), CL-LoRA (He et al., 2025), SD-LoRA (Wu et al., 2025), and LoRA-Subtraction (Liu & Chang, 2025). They constrain new updates through orthogonal subspaces, null-space projections, shared/task-specific adapter splits, direction–magnitude decoupling, or drift-resistant spaces. This line of work addresses forgetting by shaping where LoRA updates may live, whereas expert isolation style avoids interference by never sharing task-specific updates and then delegating the remaining problem to a router. Prototype-based CL traces back to iCaRL (Rebuffi et al., 2017), where class means support incremental recognition. Our contribution is to lift this idea from class prototypes to task prototypes over frozen visual and language representations, and to use those prototypes as a replay-free, zero-parameter router for generative MLLM experts.

MLLM-CL baselines. The baselines in Tab. 1 cover the main families of continual learning. *Replay/regularization.* EWC (Kirkpatrick et al., 2017) adds a quadratic Fisher-weighted penalty that anchors parameters important to past tasks, while Experience Replay (Chaudhry et al., 2019) interleaves stored past samples with new-task batches. SEFE (Chen et al., 2025) applies element-wise regularization to LoRA updates so that parameters consolidated for past tasks are protected while new-task adaptation continues. *Subspace isolation.* O-LoRA (Wang et al., 2023) constrains each new task's LoRA update to a subspace orthogonal to those of all previous tasks. *MoE-LoRA routing.* MoELoRA (Liu et al., 2024c) attaches a token-level soft router over a fixed pool of LoRA experts trained with a contrastive load-balancing objective; CL-MoE (Huai et al., 2025) extends this with dual momentum updates on the router and experts to trade off plasticity and stability for continual VQA. *Hierarchical/structured LoRA for MLLMs.* HiDe-LoRA (Guo et al., 2025a) hierarchically decouples task-shared and task-specific LoRA components for continual instruction tuning of MLLMs, and DISCO-LoRA (as implemented in MCITlib (Guo et al., 2025b)) keeps a per-task LoRA bank and selects parameter embeddings at inference via textual similarity to a codebook of past instructions. *Isolated experts with a learned MLLM router.* MR-LoRA (Zhao et al., 2025) trains one fresh LoRA expert per task in isolation and uses a replay-tuned MLLM-LoRA that autoregressively emits the expert identifier at inference.

C. Algorithmic Details and Additional Results for Prototypical Routing (RePro)

This appendix provides the training configuration, prototype bank construction protocol, and expanded empirical ablations for the RePro router introduced in Sec. 2.1. We restate the routing pipeline in Sec. C.1, document the isolated expert-training regime in Sec. C.2, describe prototype bank construction in Sec. C.3, and report additional diagnostics in Sec. C.4.

C.1. RePro and Multimodal-RePro

RePro decouples expert selection from autoregressive answer generation. The pipeline consists of three stages:

1. **Isolated expert training.** For each domain $k \in [K]$, we estimate a task-specific LoRA adapter θ_k by minimizing

the autoregressive negative log-likelihood on $\mathcal{D}_k$:

$$\mathcal{L}_{\text{LM}}(\theta_k) = -\sum_{(x,q,y)\in\mathcal{D}_k} \sum_{t=1}^{|y|} \log p_{\theta_k}(y_t \mid x, q, y_{<t}). \tag{3}$$

The backbone, previous adapters, and routing rule are held fixed, so expert training introduces no router-specific gradients.

2. **Prototype bank construction.** For each domain k, we sample a prototype support set $\mathcal{S}_k \subset \mathcal{D}_k$ with $|\mathcal{S}_k| = n$ ($n = 128$ in our experiments). We extract the `[CLS]` representation from the frozen CLIP vision encoder, $\phi_v(\cdot)$, and compute an ℓ_2-normalized centroid:

$$\bar{\mathbf{c}}_k^v = \ell_2\left(\frac{1}{n}\sum_{x\in\mathcal{S}_k} \ell_2\big(\phi_v(x)\big)\right). \tag{4}$$

3. **Nearest-prototype routing.** At inference time, a query image x is assigned to the expert whose prototype has maximum cosine similarity to the frozen visual representation:

$$k^\star(x) = \arg\max_{k\in[K]} \ \ell_2\big(\phi_v(x)\big)\cdot \bar{\mathbf{c}}_k^v. \tag{5}$$

The MULTIMODAL-REPRO variant adds instruction-level task evidence. It constructs text prototypes $\bar{\mathbf{c}}_k^\ell$ from mean-pooled hidden states of the frozen LLM backbone, $\phi_\ell(q)$, and applies late fusion:

$$k^\star(x,q) = \arg\max_{k\in[K]} \max\Big(\cos\big(\phi_v(x), \bar{\mathbf{c}}_k^v\big),\ \cos\big(\phi_\ell(q), \bar{\mathbf{c}}_k^\ell\big)\Big) \tag{6}$$

The complete procedure is provided in Alg. 1.

Algorithm 1 REPRO: Replay-Free Prototypical Routing

Require: base MLLM f_θ, frozen vision encoder ϕ_v, frozen text encoder ϕ_ℓ, task stream $\{\mathcal{D}_k\}_{k=1}^K$, support-set size n

Stage 1: Isolated expert training
for $k = 1, \dots, K$ **do**
  Load f_θ with a fresh LoRA θ_k
  Optimize θ_k on $\mathcal{D}_k$ using $\mathcal{L}_{\text{LM}}$
end for
Stage 2: Prototype bank construction
for $k = 1, \dots, K$ **do**
  Sample prototype support set $\mathcal{S}_k \subset \mathcal{D}_k$, $|\mathcal{S}_k| = n$
  $\bar{c}_k^v \leftarrow \ell_2\big(\frac{1}{n}\sum_{(x,q,y)\in\mathcal{S}_k} \ell_2(\phi_v(x))\big)$
  *(*MULTIMODAL-REPRO*)* $\bar{c}_k^\ell \leftarrow \ell_2\big(\frac{1}{n}\sum_{(x,q,y)\in\mathcal{S}_k} \ell_2(\phi_\ell(q))\big)$
end for
Stage 3: Nearest-prototype routing (per query (x, q))
$k^\star \leftarrow \arg\max_k \cos\big(\phi_v(x), \bar{c}_k^v\big)$
*(*MULTIMODAL-REPRO*)* $k^\star \leftarrow \arg\max_k \max\big(\cos(\phi_v(x), \bar{c}_k^v),\ \cos(\phi_\ell(q), \bar{c}_k^\ell)\big)$
Activate expert $\theta_{k^\star}$ in f_θ and decode the answer

C.2. Expert Isolation and Training Configurations

To implement strict expert isolation, each task T_k in the continual-learning sequence is trained in an independent DeepSpeed process. We load the base LLaVA-1.5 model, attach a fresh rank-8 LoRA adapter, and optimize only that adapter on the corresponding domain dataset $\mathcal{D}_k$. No trainable parameters are shared across task-specific adapters.

Hyperparameters. Training is conducted for 1 epoch per task using a learning rate of 10^{-4} with a cosine decay schedule, 0.03 warmup ratio, per-device batch size of 4, and gradient accumulation of 2. No LoRA weights, optimizer states, or scheduler momentum are carried over between tasks.

C.3. Prototype Bank Construction

After training the $K = 5$ isolated experts, we construct a prototype bank from prototype support sets $\mathcal{S}_k$ with 128 training instances per domain $k \in \{\text{RS, Med, AD, Sci, Fin}\}$. Let $x_i^{(k)}$ denote the image and $q_i^{(k)}$ denote the human-turn instruction. Answers are excluded to prevent label leakage into textual prototypes. We evaluate three feature spaces:

1. **Vision (ϕ_v):** The `[CLS]` token of the frozen LLaVA vision feature extractor ($d = 1024$).
2. **Text-CLIP (ϕ_t):** The text pooler output from the frozen CLIP text encoder ($d = 768$).
3. **Text-LLM (ϕ_ℓ):** The mean-pooled last-hidden state of the frozen LLaMA backbone over non-pad tokens ($d = 4096$). By operating on tokenized text directly, we bypass multimodal label preparation, requiring no image tokens for this step.

For each modality $m \in \{v, t, \ell\}$, the domain prototype is the ℓ_2-normalized mean of the ℓ_2-normalized feature vectors. Prototype bank construction for all five domains requires approximately 3 minutes on a single H100 GPU.

C.4. Additional Results

C.4.1. PER-MODALITY ROUTING ACCURACY

A single modality is sufficient: pure CLIP vision already routes at 99.52%, and pure LLaMA text at 97.42%. Adding the second modality gets an additional 0.36 percentage points of routing accuracy and only 0.10 percentage points of task accuracy. The signal is consistent with the t-SNE visualisation in Fig. 1: domains are pre-separated in both pretrained spaces, so the routing problem reduces to nearest-centroid lookup.

Table 4. **Prototypical Routing Accuracy by Modality.** Evaluated over a 5,000-sample intersection test set using the REPRO router. Near-perfect routing accuracies of unimodal signals confirm the extreme linear separability of MLLM-CL task domains in the latent space, rendering over-parameterized routers redundant.

MODALITY	ENCODER	ROUTE ACC.	AVG. TASK ACC.
TEXT	LLAMA	97.42	67.65
VISION	CLIP	99.52	68.13
MULTIMODAL	CLIP + LLAMA	99.88	68.23

C.4.2. MODALITY FUSION ABLATION

To rigorously determine the most effective mechanism for combining ϕ_v and ϕ_ℓ in MULTIMODAL-REPRO, we conducted an ablation over different fusion rules. Let $s_k \in \mathbb{R}$ denote the routing score for expert k. The principal configurations are:

- **Weighted Sum**: $s_k = \alpha \cos(\phi_v, \bar{\mathbf{c}}_k^v) + (1 - \alpha)\cos(\phi_\ell, \bar{\mathbf{c}}_k^\ell)$, with $\alpha \in \{0.0, 0.2, 0.4, 0.5, 0.6, 0.8, 1.0\}$. Pure text is $\alpha = 0$; pure vision is $\alpha = 1$.
- **Concatenation**: $s_k = \cos\big(\ell_2(\phi_v \oplus \phi_\ell),\ \ell_2(\bar{\mathbf{c}}_k^v \oplus \bar{\mathbf{c}}_k^\ell)\big)$.
- **Z-Norm Sum**: $s_k = z(\cos_k^v) + z(\cos_k^\ell)$, with per-sample z-score over k, correcting for scale mismatches between the 1024-d and 4096-d spaces.
- **Max Fusion**: $s_k = \max\big(\cos(\phi_v, \bar{\mathbf{c}}_k^v),\ \cos(\phi_\ell, \bar{\mathbf{c}}_k^\ell)\big)$.

In this ablation, fusion rules (**Concatenation**) and statistical normalizations (**Z-norm Sum**) fail to outperform naive late-fusion. **Max Fusion** proves to be the optimal fusion rule. Notably, CLIP-Text prototypes (ϕ_t) were empirically redundant due to cross-modal contrastive alignment during pre-training; therefore, we use LLaMA hidden states (ϕ_ℓ) for the textual modality.

C.4.3. ROUTING COST

REPRO reduces the per-decision cost by approximately *eight orders of magnitude in FLOPs* and *three orders of magnitude in latency* relative to MR-LoRA's MLLM-LoRA selector. The router has zero learnable parameters and a 20 KB on-disk footprint for the prototype bank, so it adds no gradient interference, no optimiser state, and no GPU memory overhead at training time. Combined with the prototype bank construction cost of ~ 3 minutes on a single GPU, the entire routing pipeline, including training, prototype bank construction, and inference, is dominated by the cost of the experts themselves.

Table 5. Measured routing cost on H100 NVL, batch size of 1, datatype is bf16. $L = 32, T = 64, d = 4096, d_\phi = 1024, K = 5$. The MR-LoRA routing prompt is 999 tokens (Vicuna system prompt + the routing instructions and expert descriptions of Zhao et al. (2025) + 576 image tokens + the user question), so we report the full per-query routing pass: CLIP-ViT-L/14-336 forward, the projector head, and a prefill process of 999 tokens followed by a single decoded token.

ROUTER	PARAMS	FLOPS	MS
ORACLE ROUTING (MR-LORA, FULL)	7.06 B	1.33×10^{13}	71.82
CLIP-VIT-L/14-336	0.30 B	3.5×10^{11}	5.78
MM_PROJECTOR	21 M	2.4×10^{10}	0.10
LLAMA-LM (PREFILL 999 + 1)	6.74 B	1.30×10^{13}	65.93
TOKEN SOFTMAX (CL-MOE)	20 K	8.4×10^{7}	53.10
OURS (REPRO)	**0**	$\mathbf{4.1 \times 10^{4}}$	**0.06**

D. Algorithmic Details and Extended Results for Mixture of Shared Experts (MoSE)

In this section, we provide the algorithmic formulations, auxiliary objective details, and extended empirical results for the **Mixture of Shared Experts (MoSE)** framework discussed in Section 4.2. All shared-expert models use a fixed expert cardinality of $K = 5$, matching the number of task domains in the DCL sequence. We use the same descriptive variant names as the main text rather than internal run identifiers.

D.1. Sample-level Global Routing

Sample-level global routing is a parameterized shared-expert router that computes one routing distribution per input sequence and applies that distribution uniformly to all tokens. The design is intended to capture coarse task identity from pooled multimodal representations before expert composition.

Algorithmic Formulation. Given an input sequence of hidden states $\{x_1, x_2, \ldots, x_T\}$ with $x_t \in \mathbb{R}^d$, we first pool the sequence to obtain a global representation $\bar{x} \in \mathbb{R}^d$:

$$\bar{x} = \frac{1}{T}\sum_{t=1}^{T} x_t \tag{7}$$

The router features two parallel branches—a linear vision branch and a non-linear language multi-layer perceptron (MLP) branch:

$$s_v = W_v \bar{x}, \qquad s_\ell = W_\ell^{(2)}\,\mathrm{ReLU}\big(W_\ell^{(1)}\bar{x}\big) \tag{8}$$

where $W_v \in \mathbb{R}^{K\times d}$. The language projection bottleneck dimension is 256, so $W_\ell^{(1)} \in \mathbb{R}^{256\times d}$ and $W_\ell^{(2)} \in \mathbb{R}^{K\times 256}$. The modalities are dynamically fused via learnable branch weights $w \in \mathbb{R}^2$ (initialized to $[0.5, 0.5]$):

$$\hat{w} = \mathrm{softmax}(w), \qquad s = \hat{w}_1 s_v + \hat{w}_2 s_\ell \tag{9}$$

During training, we inject Gaussian noise to encourage expert exploration: $s' = s + \varepsilon\sigma_n$ where $\varepsilon \sim \mathcal{N}(0,1)$ and $\sigma_n = 0.1$. The final routing mixture is $r = \mathrm{softmax}(s')$. For every token t, the layer output is computed as:

$$y_t = W_0 x_t + \frac{\alpha}{r_{lora}} \sum_{k=1}^{K} r_k \big(B_k A_k x_t\big) \tag{10}$$

where r_{lora} is the LoRA rank. This dual-branch architecture adds ∼1M parameters per target module.

Ablation configurations and analysis. We evaluate four sample-level configurations in Table 2:

- **Noisy softmax routing.** The router is trained only through the language-modeling loss $\mathcal{L}_{LM}$, with Gaussian routing noise for exploration. Router parameters are initialized from $\mathcal{N}(0, 0.01^2)$.
- **Top-k sparsified softmax routing.** The router retains only the largest $k = 2$ logits and masks the remaining logits to $-\infty$ before the softmax. This hard sparsification produces premature commitment to early domains and yields the worst backward transfer in the sample-level family (BWT $= -9.45\%$).
- **Top-k routing with task-guided supervision.** We add cross-entropy supervision on the routing logits, $\mathcal{L}_{tg} = \mathrm{CE}(\mathrm{softmax}(s), k_{gt})$, with $\lambda_{tg} = 1.0$. This improves task differentiation and gives the best sample-level mean accuracy (55.28%), but it remains well below isolated expert routing.
- **Top-k routing with task-guided supervision and orthogonal loss.** We use a wider initialization, $\mathcal{N}(0, 0.1^2)$, and penalize expert overlap with a task-similarity-weighted orthogonality objective, $\mathcal{L}_{orth} = \sum_{i<j}(1 - \mathrm{sim}(T_i, T_j))\|A_i A_j^\top\|_F^2$. This increases forward transfer (FWT $= +2.88\%$) but reduces retention, consistent with the trade-off reported in the main text.

D.2. Token-level Local Routing

Token-level local routing tests whether the shared-expert plateau is specific to sample-level assignment. Instead of assigning one mixture to the whole sequence, this router computes token-specific expert weights using a continuous, differentiable sigmoid gate.

Algorithmic Formulation. For each individual token t, the dense hidden state x_t is linearly projected to base routing logits $s_t = W_r x_t \in \mathbb{R}^K$. To enable independent suppression or activation of experts without hard masking, we compute a per-token, per-expert gate $\alpha_{t,i} \in (0,1)$ using a parameterized sigmoid function:

$$\alpha_{t,i} = \sigma\Big(\beta\,(s_{t,i} - \tau_i)\Big) \tag{11}$$

where $\tau \in \mathbb{R}^K$ are learnable per-expert thresholds (initialized to 0) and $\beta \in \mathbb{R}$ is a learnable global scaling factor controlling the sharpness of the threshold (initialized to 1.0). The final mixture weights for the token are computed via a masked softmax:

$$r_t = \mathrm{softmax}(s_t \odot \alpha_t) \tag{12}$$

This parameter-efficient dense router requires only ∼20K parameters per module.

Ablation configurations and analysis. We evaluate two token-level configurations in Table 2:

- **Sigmoid routing.** The router is trained through $\mathcal{L}_{LM}$ without explicit task labels and obtains 54.94% mean accuracy. Empirical logs show that β increases to approximately 1.5–2.0 and the thresholds τ diverge around zero, indicating that the gate learns non-trivial expert modulation rather than collapsing to a uniform mixture.
- **Sigmoid routing with task-guided supervision.** We add sequence-aggregated task supervision, $\mathcal{L}_{tg} = \mathrm{CE}(\mathrm{softmax}(\bar{s}), k_{gt})$ with $\bar{s} = \frac{1}{T}\sum_{t=1}^{T} s_t$ and $\lambda_{tg} = 0.1$. This improves mean accuracy to 55.32% and reduces backward forgetting (BWT improves from -7.83% to -5.88%), but the result still falls far short of isolated expert routing.

D.3. Training-Loss Diagnostics

Figure 2 provides an optimization diagnostic for the negative shared-expert result. If MoSE failed because the shared routers or experts simply did not learn, its training loss should diverge, plateau early, or show a qualitatively worse decay profile than the sequential and replay baselines. Instead, the loss curves for MoSE-softmax and MoSE-sigmoid closely track the standard baselines on each task. The final logged losses for the full-data MoE-style runs are also nearly matched: MoELoRA, MoSE-softmax, and MoSE-sigmoid end at 0.354/0.347/0.352 on RS, 1.332/1.317/1.294 on Med, 0.786/0.760/0.775 on Sci, and 0.145/0.148/0.146 on Fin. Where all three are available for AD, MoSE-softmax and MoSE-sigmoid also end near Seq-FT (0.352/0.363 vs. 0.352 for Seq-FT with rank 32).

This evidence rules out a simple optimization-failure account. **The shared-MoE models do learn the current task objective, but their learned updates still do not transfer into high retained accuracy across the DCL sequence**. The failure is therefore better explained by cross-task interference and the absence of useful overlap in the task manifolds, as argued in Sec. 3.

The exceptions are informative. O-LoRA and HiDe-LoRA include auxiliary regularization/decomposition terms that

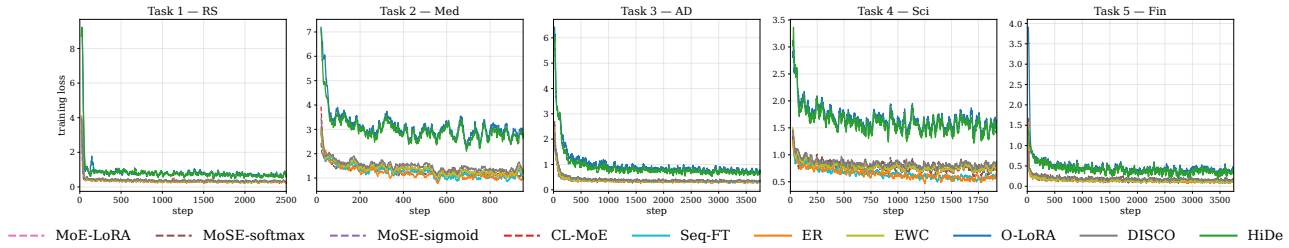

Figure 2. **Shared-MoE models optimize normally despite poor final accuracy.** Training-loss curves across the five DCL tasks. MoSE-softmax, MoSE-sigmoid, MoELoRA, Seq-FT, ER, and EWC exhibit similar convergence patterns within each task, ruling out failed optimization as the primary explanation for the shared-expert performance plateau. O-LoRA and HiDe-LoRA are the main exceptions: their auxiliary objectives keep the reported training loss elevated on several tasks, consistent with their weak DCL accuracy.

change the reported training objective rather than optimizing the language-modeling loss alone. Their curves remain substantially higher on several domains, especially Med and Sci, and these are also among the weakest methods in Tab. 1. Thus, unlike MoSE and MoELoRA, their poor benchmark performance is at least partly consistent with objective-level optimization pressure from the auxiliary terms.

Table 6. Task orders evaluated in the order-sensitivity ablation. Difficulty is anchored to the zero-shot accuracy of the un-adapted backbone on each domain.

Order	Sequence	Rationale
`canonical`	RS → Med → AD → Sci → Fin	Original order from the MLLM-CL benchmark.
`hard-to-easy`	AD → Med → RS → Sci → Fin	Anti-curriculum: hardest first.
`easy-to-hard`	Fin → Sci → RS → Med → AD	Curriculum: easiest first.
`random`	Sci → Fin → AD → RS → Med	Random permutation

D.4. Task-Order Sensitivity Ablation

The main DCL protocol evaluates all methods on the canonical MLLM-CL order, RS → Med → AD → Sci → Fin. For shared-pool MoE tuners, this single order can confound method quality with the position at which each domain is encountered: early domains are exposed to all subsequent overwriting updates, while late domains are evaluated shortly after training. We therefore repeated the DCL protocol under the four permutations in Table 6 and measured

$$\Delta_{\text{order}}(m) = \max_{\pi \in \Pi} \bar{a}_5(m, \pi) - \min_{\pi \in \Pi} \bar{a}_5(m, \pi), \qquad (13)$$

where $\bar{a}_5(m, \pi)$ is the mean final-task accuracy across RS, Med, AD, Sci, and Fin after method m is trained under order π.

All task-order runs use LLaVA-1.5-7B with the CLIP-ViT-L/14-336 as the vision feature extractor, $E = 5$ LoRA experts, LoRA rank $r = 30$ ($r/E = 6$ per expert for split-rank CoIN-style MoE variants), LoRA $\alpha = 60$, one epoch per task, AdamW with a 10^{-4} learning rate and cosine schedule, a uniform training fraction $\rho = 0.1$, and an evaluation cap of 1,000 examples per domain. Sequential training warm-starts task t from the adapter saved after task $t - 1$ and then evaluates the final task-5 adapter on all five domains.

Insights. Table 3 shows that task order dominates the gating function: all three evaluated methods follow the same ranking, `easy-to-hard` > `random` > `canonical` ≥ `hard-to-easy`. The within-method order spread is 6.36–8.21 pp, while the largest between-method spread within a fixed order is only 4.73 pp. Thus, for shared-pool MoE tuners on DCL, which permutation is used matters more than whether the router is top-k noisy softmax, dense softmax, or dense sigmoid.

The main driver is late-task forgetting of the hardest domain. AD has the lowest zero-shot accuracy (15.58%) and the largest order-induced swing: CL-MoE, MoELoRA, and MoSE-sigmoid reach 46.75%, 45.05%, and 45.70% when AD is task 5 in `easy-to-hard`, but only 24.45%, 18.35%, and 20.65% when AD is task 1 in `hard-to-easy`. In contrast, Finance, the easiest domain (62.57% zero-shot), varies by less than 4 pp across all completed orders. Canonical DCL is therefore a near-worst case at $\rho = 0.1$: it is the worst order for MoELoRA and the second-worst order for CL-MoE and MoSE-sigmoid.

MoSE-softmax is omitted from Tables 3 and 7 because it shares the same CoIN-style softmax forward pass and parameter shape as MoELoRA; the task-order orchestrators intentionally skip it as a parity duplicate rather than substitute full-fraction legacy numbers into the $\rho = 0.1$ matrix.

Table 7. Final accuracy (%) under task-order ablations. Can.=canonical, H→E=hard-to-easy, E→H=easy-to-hard.

Method	**Order**	**RS**	**Med**	**AD**	**Sci**	**Fin**	**Mean**
CL-MoE	Can.	69.42	34.08	31.12	38.25	86.00	51.77
	H→E	65.65	31.70	24.45	37.22	85.60	48.92
	E→H	69.00	33.25	46.75	43.24	84.15	55.28
	Rand.	63.45	34.30	32.15	37.15	85.55	50.52
MoELoRA	Can.	57.45	30.55	27.85	36.93	82.40	47.04
	H→E	62.40	30.70	18.35	36.16	83.15	46.15
	E→H	66.20	35.75	45.05	42.50	82.30	54.36
	Rand.	59.00	31.95	32.65	36.64	84.65	48.98
MoSE-sigmoid	Can.	68.15	31.20	26.30	37.17	85.70	49.70
	H→E	63.45	31.05	20.65	37.63	86.25	47.81
	E→H	67.85	35.10	45.70	41.63	84.90	55.04
	Rand.	65.15	34.80	33.00	37.10	85.95	51.20