Routing-Based Continual Learning for Multimodal Large Language Models

Jay Mohta*, **Kenan Emir Ak***, **Gwang Gook Lee**,
Dimitrios Dimitriadis, Yan Xu, Mingwei Shen
Amazon.com
{jaymoht, kenanea, gglee, dbdim, yanxuml, mingweis}@amazon.com

Abstract

Multimodal Large Language Models (MLLMs) struggle with continual learning, often suffering from catastrophic forgetting when adapting to sequential tasks. We introduce a routing-based architecture that integrates new capabilities while robustly preserving foundational knowledge. While Multi-Task Learning (MTL) is a strong performance reference, its training cost grows linearly with the number of tasks. In contrast, our method maintains fixed per-task training data and compute requirements regardless of the task sequence length. Across models ranging from 2B to 8B parameters, we demonstrate that our routing approach performs on par with MTL while retaining the training efficiency of sequential fine-tuning. Beyond merely mitigating forgetting, we observe that token-level routing facilitates cross-modal transfer, leveraging knowledge from one modality to bolster performance in another. We also show that routing remains robust even with large expert pools and effectively capitalizes on task relatedness. Finally, we show that our method scales favorably, with larger models exhibiting minimal degradation compared to fully specialized fine-tuning.

1 Introduction

Large Language Models (LLMs) have fundamentally transformed Natural Language Processing (NLP) (Brown et al., 2020), a success now mirrored by Multimodal Large Language Models (MLLMs) capable of reasoning leveraging visual and textual inputs (Radford et al., 2021; Alayrac et al., 2022; Liu et al., 2023). However, adapting these general-purpose models to specialized domains introduces a critical dilemma: optimizing performance on target tasks via fine-tuning frequently triggers catastrophic forgetting (CF), severely degrading the model's previously acquired knowledge (McCloskey and Cohen, 1989). This challenge is most severe in continual learning settings, where models must learn a sequence of tasks without accessing historical training data (Kemker et al., 2018). Given the massive scale of MLLMs, traditional solutions such as experience replay (Rolnick et al., 2019; Rebuffi et al., 2017) or parameter regularization (Kirkpatrick et al., 2017) are often computationally prohibitive or ineffective (Wu et al., 2024). Because general linguistic and visual knowledge is distributed across billions of parameters, task-specific updates may lead to catastrophic forgetting, where the optimization for new objectives inadvertently overwrites the representations necessary for earlier tasks (Kotha et al., 2023).

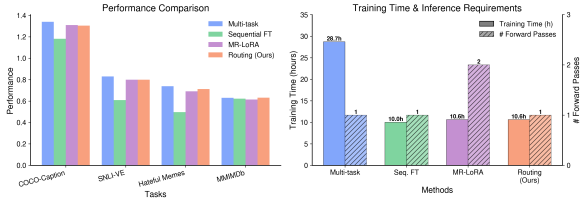


Figure 1: Comparison of continual learning methods. **Left:** Task performance across four vision-language benchmarks. **Right:** Training time and inference requirements. Our routing-based approach performs on par with Multi-Task Learning while requiring 2.7× less training time and only a single forward pass at inference, unlike MR-LoRA which requires two.

In the multimodal context, this interference is particularly damaging as it risks the cross-modal alignment established during pretraining (Zhai et al., 2023). Fine-tuning can decouple visual embeddings from their textual counterparts, breaking the model's reasoning capabilities (Zhu et al., 2024). Furthermore, visual encoders and language decoders often suffer from asymmetric forgetting rates, further destabilizing the model (Ni et al., 2023). Consequently, there is a critical need for architectures that allow plasticity for new tasks while enforcing stability for foundational cross-modal alignments.

Figure 1 illustrates the trade-offs between these

*Equal contribution

Proceedings of the 15th International Joint Conference on Natural Language Processing and the 5th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 2177–2192
November 6-10, 2026 ©2026 Association for Computational Linguistics

paradigms. While Sequential Fine-Tuning is data-efficient, it suffers from a sharp decline in performance on prior tasks. Conversely, Multi-Task Learning (MTL) can improve stability by retraining on the cumulative dataset, but its training cost grows linearly with the number of tasks, which is unsustainable for large-scale MLLMs. Furthermore, while modular frameworks like MR-LoRA (Zhao et al., 2025) reduce interference, they often require two forward passes at inference to resolve task-specific outputs. Our approach aims to occupy a middle ground wherein it achieves the stability of MTL and the efficiency of sequential methods while maintaining only a single-pass inference.

In this work, we adapt token-level routing over a library of LoRA experts (Muqeeth et al., 2024; Ostapenko et al., 2024) to continual learning in MLLMs. We do not claim the routing primitive itself as new. Instead, we show that this modular design isolates task-specific adaptations to prevent the overwriting of shared knowledge, and we study the properties it exhibits in the multimodal setting. Our contributions are fourfold:

1. We show that routing-based continual learning matches MTL using only the current task's data, eliminating costly retraining.
2. We find that token-level routing enables cross-modal transfer, allowing unimodal expert improvements (e.g., text-only) to bolster multimodal performance.
3. We provide extensive evaluations across sequential tasks, showing robustness to expert pool size and positive utilization of task relatedness.
4. We identify a positive scaling trend where as model size grows, our method approaches fully specialized fine-tuning, suggesting favorable scaling for larger backbones.

2 Related Work

2.1 Multimodal Large Language Models

MLLMs extend the capabilities of LLMs by processing visual inputs alongside text, typically comprising a pretrained vision encoder, a language backbone, and a connector module that projects visual features into the language embedding space (Yin et al., 2024). Early architectures, such as Flamingo (Alayrac et al., 2022) and BLIP-2 (Li et al., 2023), established foundational design principles by bridging frozen vision and language models via modular components.

Subsequent research has prioritized scaling model capacity and refining instruction tuning. LLaVA (Liu et al., 2023, 2024a) demonstrated the efficacy of visual instruction tuning using simple projection layers, while models such as Qwen-VL (Bai et al., 2023) and InternVL (Chen et al., 2024b,a) introduced multi-stage training paradigms incorporating diverse data mixtures and dynamic resolution processing. However, these architectures are typically optimized for static training pipelines under the assumption of i.i.d. data distributions. Consequently, how to effectively incorporate continual adaptation into these frameworks remains unclear; without explicit mechanisms to manage sequential learning, they remain susceptible to catastrophic forgetting.

2.2 Continual Learning in Large Models

Catastrophic forgetting remains a persistent challenge in deep learning, exhibiting a relationship with model scale that is both complex and task-dependent (Ramasesh et al., 2021; Kalajdzievski, 2024). In MLLMs, it becomes more difficult due to cross-modal interference where sequential adaptation often disrupts the alignment established during pretraining, causing fine-tuning in one modality to degrade representations in the other (Zhai et al., 2023).

Traditional continual learning strategies face severe scalability bottlenecks when applied to billion-parameter models. Replay-based methods (Rolnick et al., 2019; Chaudhry et al., 2019) require the retention of historical data, raising privacy and storage concerns, while regularization techniques (Kirkpatrick et al., 2017; Zenke et al., 2017) frequently demand computationally prohibitive importance scoring across the entire network. Although multi-task learning is a strong performance reference, it necessitates simultaneous access to all datasets, rendering it impractical for dynamic data streams (Ruder, 2017).

To address these limitations, recent Parameter-Efficient Fine-Tuning (PEFT) methods have been adapted for the continual learning setting. Approaches such as Progressive Prompts (Razdaibiedina et al., 2023) utilize task-specific tokens, while O-LoRA (Wang et al., 2023) constrains adaptation to orthogonal subspaces. Similarly, SEMA (Wang et al., 2025) introduces adapter-based sequential training with automatic expansion

for vision-based learning. However, the MLLM-CL benchmark (Zhao et al., 2025) indicates that while these approaches mitigate forgetting, they struggle with complex, domain-specific shifts. Unlike these adapter-centric strategies, our method operates at *token level*, enabling fine-grained routing that better accommodates vision-language tasks.

2.3 Model Merging and Modular Routing

Modular architectures offer a promising alternative to rigid end-to-end training. Model merging, for instance, combines the weights of separately fine-tuned models without requiring concurrent data access (Matena and Raffel, 2022; Ilharco et al., 2022; Yadav et al., 2023). While computationally efficient, merging is static; it often underperforms when tasks are semantically distant or require conflicting reasoning patterns, leading to interference in the final averaged weights.

Routing-based methods address these challenges by dynamically selecting specialized sub-networks based on input semantics. While frameworks like PHATGOOSE (Muqeeth et al., 2024) and Arrow (Ostapenko et al., 2024) utilize modular routing for zero-shot generalization, and Mixture-of-Experts (MoE) architectures (Lin et al., 2024; Riquelme et al., 2021) employ it for capacity scaling, their application in continual learning remains underexplored. Specifically, standard MoEs often lack mechanisms to isolate task-specific knowledge or handle the sequential arrival of data, and we detail how our approach differs from them in Appendix E. Our approach addresses this by leveraging routing to manage an ever-growing library of Low-Rank Adapters (LoRA). Rather than updating a static set of parameters, we instantiate a new LoRA 'specialist' for each incremental task. The router then learns to steer inputs toward the appropriate module, allowing the MLLM to broaden its capabilities while maintaining strict functional isolation to prevent catastrophic forgetting.

3 Proposed Method

In this paper, we introduce a modular routing framework designed to address the stability-plasticity dilemma in MLLMs (Shi et al., 2024). Our routing mechanism follows PHATGOOSE (Muqeeth et al., 2024), which demonstrated modular routing for zero-shot generalization in text-only LLMs, and we adapt it to replay-free multimodal continual learning. Our core insight is the decoupling of *representation learning (skill acquisition)* from *task identification (routing)*. By isolating these processes, we ensure that new capabilities are integrated without affecting the model's existing cross-modal alignment.

As illustrated in Figure 2, our framework operates in three phases: (1) Expert Specialization via LoRA, (2) Router Vector Training, and (3) Dynamic Inference.

3.1 Architecture Overview

We utilize a standard MLLM architecture composed of a vision encoder, a projection layer, and an LLM. To isolate the impact of our routing mechanism and ensure a controlled evaluation of catastrophic forgetting, we keep the vision encoder and projection layer frozen in our main experiments. We implement task-specific LoRA modules within all linear layers of the LLM (query, key, value, output, gate, up, and down projections). The same mechanism also applies when the vision encoder and projector are adapted, as shown in Appendix G.

3.2 Training Strategy

Our training strategy separates the acquisition of task knowledge from the task selection. This separation is critical for preventing the gradient conflict typically observed in sequential fine-tuning.

Step 1: Expert Specialization (LoRA). For each arriving task T_i, we specialize the LLM by training a dedicated LoRA adapter. Let $W_0 \in \mathbb{R}^{d\times d_{in}}$ denote the frozen weights of a linear layer in the base LLM. For an input activation $u_t \in \mathbb{R}^{d_{in}}$ at token t, the output h_t is computed as:

$$h_t = W_0 u_t + B_i A_i u_t, \tag{1}$$

where $A_i \in \mathbb{R}^{r\times d_{in}}$ and $B_i \in \mathbb{R}^{d\times r}$ are the trainable LoRA parameters with rank $r \ll d$. These parameters are optimized using only the dataset for task T_i, ensuring the expert captures task-specific nuances without being regularized by prior tasks.

Step 2: Routing Vector (RV) Training. Once the expert for task T_i is trained, we freeze parameters A_i and B_i and add a lightweight routing vector $v_i \in \mathbb{R}^{d_{in}}$. This vector serves as a soft gate, learning to identify the activation subspace associated with task T_i.

The routing vector is initialized to zero and optimized using the same next-token prediction objective as Step 1. During this phase, the layer output

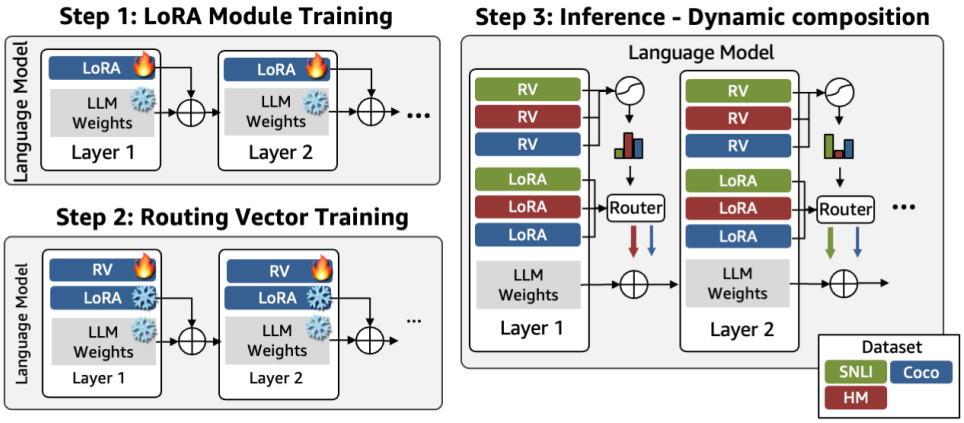


Figure 2: **The proposed method.** The process is divided into three steps: (1) Training task-specific LoRA adapters; (2) Training lightweight Routing Vectors (RV) to recognize task-specific activation patterns; (3) Inference using dynamic composition to select the top-k experts per token.

is governed by:

$$h_t = W_0 u_t + \sigma(v_i^T u_t) \cdot (B_i A_i u_t), \quad (2)$$

where $\sigma(\cdot)$ is the sigmoid function. By optimizing v_i while the expert is active, the router learns a latent "signature" of the task within the activation space. Crucially, this step requires no access to data or LoRA modules from prior tasks.

3.3 Inference

During inference, we utilize a bank of trained experts and their corresponding routing vectors $\mathcal{V} = \{v_1, \ldots, v_N\}$ for N accumulated tasks. As depicted in Step 3 of Figure 2, routing decisions are computed dynamically at the token level, allowing the model to switch contexts mid-generation.

For an input activation u_t, we compute an affinity score with each routing vector:

$$s_{t,j} = v_j^T u_t, \quad j \in \{1, \ldots, N\}. \quad (3)$$

We then identify the set of indices $\mathcal{E}_t$ corresponding to the top-k affinity scores. To effectively mix the selected experts, we normalize these scores using a softmax function with a scaling temperature τ:

$$w_{t,j} = \frac{\exp(s_{t,j}/\tau)}{\sum_{m \in \mathcal{E}_t} \exp(s_{t,m}/\tau)}, \quad \forall j \in \mathcal{E}_t. \quad (4)$$

The final output is the superposition of the base model and the weighted experts:

$$h_t = W_0 u_t + \sum_{j \in \mathcal{E}_t} w_{t,j} (B_j A_j u_t). \quad (5)$$

This design ensures that the model dynamically retrieves the most relevant expert for each token without requiring explicit task identifiers. This mitigates catastrophic forgetting while allowing for positive transfer, as vision-centric tokens can route to vision experts and logic-centric tokens to reasoning experts.

4 Experiments

This section details our experimental setup, including model architectures, datasets, and training hyperparameters. We evaluate our routing framework on InternVL-2 (Chen et al., 2024b) (2B and 8B variants) and LLaVA-v1.5-7B (Liu et al., 2023) to assess scalability and robustness across diverse model families. Appendix G extends the evaluation to Qwen3-VL-4B (Bai et al., 2025), including a setting where the vision encoder and projector are also adapted.

4.1 Datasets and Evaluation Metrics

To evaluate the stability-plasticity trade-off, we categorize our evaluation into *Specialization Tasks* and *General Benchmarks*.

Specialization Tasks. We employ a diverse set of multimodal tasks covering: (1) **COCO-Caption** (COCO) (Lin et al., 2014) for image captioning (CIDEr); (2) **SNLI-VE** (Xie et al., 2019) for visual entailment (Accuracy); (3) **Hateful Memes** (HM) (Kiela et al., 2020) for toxicity detection (Accuracy); and (4) **MMIMDb** (Arevalo et al., 2017) for movie genre classification (Precision).

General Benchmarks. We monitor retention of foundational knowledge using **MMBench (MMB)** (Liu et al., 2024b) for multi-dimensional reasoning, **ChartQA** (Masry et al., 2022) for visual

Table 1: Comparison of continual learning methods on InternVL2-2B and 8B (best in **bold**, second best underlined); Performance is reported across four sequential specialization tasks and three general-purpose benchmarks.

Size	Method	COCO	SNLI	HM	MMIMDb	MMB	ChartQA	DocVQA
2B	Zero-shot	0.795	52.9	60.2	46.6	**79.1**	**59.5**	**84.7**
	Seq-FT	1.181	61.0	49.7	62.3	67.1	41.2	70.0
	Arith. Merge	1.171	73.3	64.5	**64.9**	78.6	59.2	84.1
	MR-LoRA	1.310	80.0	69.1	61.4	-	-	-
	MTL	**1.340**	**83.6**	**73.9**	63.0	76.5	55.7	77.9
	Routing (Ours)	1.305	80.3	71.2	63.2	78.8	59.2	84.3
8B	Zero-shot	0.889	70.2	65.4	54.5	85.6	72.6	**89.7**
	Seq-FT	1.258	65.4	52.1	64.8	71.2	48.5	74.3
	Arith. Merge	1.272	71.3	68.7	66.9	86.0	70.6	87.2
	MR-LoRA	1.351	81.0	73.4	66.7	-	-	-
	MTL	**1.374**	**84.7**	**78.3**	65.4	85.7	68.2	86.5
	Routing (Ours)	1.350	83.3	76.8	**67.4**	**86.3**	**73.1**	89.3

data interpretation, and **DocVQA** (Mathew et al., 2021) for document understanding.

MLLM-CL benchmark. (Zhao et al., 2025) This benchmark spans two settings, Domain-CL covers five disparate specialized domains (Remote Sensing, Medical imaging, Art and Design, Science, and Finance), and Ability-CL targets three transferable skills (OCR, Math & Logic (M&L), and Visual Perception).

Cross-Modal Transfer. To investigate the cross-modal transfer capabilities of the proposed method, we train two distinct experts: a text-only math expert (Orca-Math (Mitra et al., 2024)) and a multimodal math expert (Multi-math (Peng et al., 2024)). We then evaluate zero-shot performance on the MGSM framework (Shi et al., 2022) in both English and Chinese.

4.2 Training Hyperparameters

Baseline Configurations. For Sequential Fine-tuning (Seq-FT), we use a learning rate of 1×10^{-5} for three epochs on each task. For Multi-Task Learning (MTL), all tasks are co-trained for three epochs. All training experiments are conducted on 32 A10 GPUs with a global batch size of 512. Arithmetic Merging (Arith. Merge) (Ilharco et al., 2022) computes a weighted average of fine-tuned weights to create a single model.

Modular Routing Training:

1. **Expert Specialization:** Task-specific LoRA modules of rank 128 trained (lr=4×10^{-5}, cosine decay).
2. **Gate Optimization:** Routing vector v trained (100 steps, batch size 32) with LoRA weights frozen.

The full set of hyperparameters, including routing temperature and decoding settings, is listed in Appendix D.

Task Sequence. We evaluate continual learning using the sequence: COCO $\rightarrow$ SNLI $\rightarrow$ HM $\rightarrow$ MMIMDb. During inference, we employ top-k routing with $k = 1$, prioritizing the most relevant expert for each token to maintain computational efficiency. We follow the suggested tasks sequence for MLLM-CL benchmark (Zhao et al., 2025).

5 Results and Analysis

5.1 Performance Comparison and Stability

As demonstrated in Table 1, the proposed routing method (**Routing**) consistently achieves the optimal balance between task-specific specialization and the preservation of foundational capabilities.

Routing vs. Multi-Task Learning. Notably, our routing approach achieves performance comparably with the MTL model. On MMIMDb, routing surpasses MTL in both the 2B (63.2 vs. 63.0) and 8B (67.4 vs. 65.4) variants. This suggests that routing effectively mitigates *negative task interference* (Wang et al., 2020). While MTL forces a shared weight space to accommodate contrasting data distributions, our approach isolates task-specific features, preventing destructive gradient interference.

Furthermore, routing demonstrates superior stability. MTL exhibits performance degradation on MMBench and DocVQA compared to the zero-shot baseline, indicating a forgetting of foundational knowledge. In contrast, routing preserves these scores near zero-shot levels.

Table 2: Comparison of routing vs. LoRA-specialized models. Routing achieves competitive or superior performance on all tasks using a single model.

Model	Specialized Tasks				Benchmark Tasks		
	COCO	SNLI	HM	MMIMDb	MMB	ChartQA	DocVQA
Zero-Shot	0.889	70.2	65.4	54.2	85.6	72.7	**89.7**
LoRA-COCO	1.348	71.8	63.2	47.8	**86.3**	71.5	89.4
LoRA-SNLI	1.084	**84.2**	63.0	53.2	85.6	72.4	89.6
LoRA-HM	1.004	74.0	76.5	52.6	85.7	71.9	84.0
LoRA-MMIMDb	0.991	69.3	64.5	65.8	85.8	72.1	89.4
Routing (Ours)	**1.350**	83.3	**76.8**	**67.4**	**86.3**	**73.1**	89.4

Table 3: Performance on MLLM-CL (LLaVA) across Domain-CL and Ability-CL. **Bold**/underline = best/2nd among practical methods (Oracle excluded).

Method	Domain-CL					Ability-CL		
	RS	Med	AD	Sci	Fin	OCR	M&L	VP
Zero-shot	32.3	28.3	15.6	35.6	62.6	31.2	30.2	60.8
Oracle	81.1	65.8	54.2	56.9	91.1	33.6	36.5	65.1
LoRA-FT	69.7	41.6	25.4	40.9	87.5	21.8	32.7	58.4
O-LoRA	74.6	44.4	30.0	41.5	87.2	29.6	31.3	60.8
MoELoRA	77.5	41.9	27.6	40.1	86.8	19.8	32.2	54.2
CL-MoE	71.3	46.8	26.3	41.2	88.7	25.4	31.8	60.9
HiDe	74.3	49.0	33.2	38.5	81.6	24.6	28.4	30.7
SEFE	77.3	50.4	37.2	40.9	86.8	25.6	34.8	57.6
DISCO	76.0	45.2	43.8	42.3	89.0	**34.2**	35.0	61.6
ModalPrompt	53.6	45.7	40.8	41.8	87.8	–	–	–
L2P	63.8	34.6	23.0	38.6	**93.0**	–	–	–
MR-LoRA	80.9	65.3	**54.1**	**56.7**	91.1	33.7	36.2	**65.1**
Routing (Ours)	**81.0**	**66.6**	53.4	56.4	91.1	33.5	**36.5**	64.9

Routing vs. Arith. Merge. Arithmetic merging produces a *static* weighted average of expert parameters, forcing all tokens through the same blended representation and causing destructive interference when tasks require conflicting weight directions. As shown in Table 1, our token-level routing instead selects experts per-token based on activation patterns, yielding a 12.0% gain on 8B SNLI (83.3 vs. 71.3) while preserving general-benchmark performance.

Routing vs. Seq-FT. Sequential FT exhibits the most severe catastrophic forgetting, with performance on general benchmarks like MMBench dropping sharply (from 79.1 to 67.1 in the 2B model). While merging and MTL offer varying degrees of stability, they either sacrifice peak task performance or require simultaneous access to all training data. Routing achieves a "best-of-both-worlds" outcome: it doesn't require access to past training dataset while matching or exceeding the stability of global multi-task training.

Comparison with MR-LoRA. Across the four specialization tasks in Table 1, routing improves on MR-LoRA in SNLI, HM, and MMIMDb while remaining within 0.005 CIDEr on COCO, in both the 2B and 8B settings. The 8B gains are substantial, reaching +2.3 points on SNLI (83.3 vs. 81.0), +3.4 on HM (76.8 vs. 73.4), and +0.7 on MMIMDb (67.4 vs. 66.7). Routing also retains MMB, ChartQA, and DocVQA close to the zero-shot baseline, whereas MR-LoRA leaves these columns empty because its router selects a single LoRA per query and cannot route benchmark inputs that fall outside the trained expert set. MR-LoRA further requires dual-pass inference, with an external router consuming one forward pass to select an expert and the chosen expert consuming a second pass to generate the response, while our routing vectors operate inside the LLM activation space and complete inference in a single decoder pass. Its router is also trained on stored samples from previously seen tasks, whereas our routing vectors are trained in isolation on each task with no access to prior data. In all comparisons, both methods use the same base model, the same per-task training data, and LoRA experts of rank 128, so the methods differ only in the routing and inference regime. As shown in Appendix G, the same pattern holds on the more recent Qwen3-VL-4B backbone, where single-pass routing achieves the best score on three of the four specialization tasks and training remains stable when the vision encoder and projector are also adapted.

5.2 Specialization vs. Generalization

A key question in modular architectures is whether a single routed model can match the performance of individual specialized models. Table 2 compares our framework against single-task LoRA experts.

The results indicate that routing matches or exceeds specialist performance on 3 of 4 tasks. For example, the routed model achieves a score of 76.8 on Hateful Memes, surpassing the 76.5 achieved by the standalone LoRA-HM model. This suggests that the routing mechanism can perform "cross-expert utilization," where tokens within a single input benefit from the specialized features of multiple expert modules.

5.3 Evaluation on MLLM-CL Benchmark

We evaluate our framework on the MLLM-CL benchmark across both Domain-CL and Ability-CL settings using LLaVA-v1.5-7B as base model. As summarized in Table 3, our routing mechanism

Table 4: Routing performance with incremental expert inclusion.

Routing Setup	COCO	SNLI	HM	MMIMDb
Zero-Shot	0.795	52.9	60.2	46.6
SNLI, COCO	1.306	80.0	61.9	48.3
+ HM	1.309	80.7	**71.6**	45.3
+ MMIMDb	1.305	80.3	71.2	**63.2**
+ ChartGemma	**1.311**	80.3	70.8	62.8
+ XNLI	1.308	**81.2**	71.2	62.9

effectively bridges the gap between zero-shot baselines and oracle.

The results highlight two primary strengths of our approach: First, by isolating domain-specific features into dedicated expert subspaces, our method achieves scores of 81.0 in Remote Sensing and 66.6 in Medical imaging, matching or slightly exceeding oracle performance. This indicates that modular specialization can be more effective than joint training, which often suffers from destructive gradient interference. Second, our approach maintains high efficiency by delivering competitive results, such as 91.1 in Finance and 36.5 on Math & Logic (M&L) using a single-pass inference. This offers a significant computational advantage over state-of-the-art baselines like MR-LoRA, which require expensive dual-inference passes to achieve similar performance.

5.4 Ablation Studies

Scalability and Expert Interference. We first evaluate the stability of the routing mechanism as the expert pool grows. A critical challenge in modular architectures is ensuring that the addition of new experts does not induce performance degradation.

Starting with two experts (SNLI and COCO), Table 4 shows that routing progressively acquires strong domain-specific knowledge while preserving previously learned capabilities. Adding new experts consistently improves performance on their corresponding domains without introducing noticeable interference across earlier tasks.

To assess resilience against irrelevant expertise, we introduced a LoRA module trained on *ChartGemma* (Masry et al., 2025), a dataset unrelated to the core evaluation suite. As shown in Table 4, the inclusion of this unrelated expert does not degrade performance across existing tasks, reinforcing the framework's scalability and its ability to ignore irrelevant modules.

Table 5: Computational efficiency comparison across methods. Training time measured in GPU hours on 32 A10 GPUs. Data requirements shown as relative factors.

Method	**Training Time (hrs)**		**Data Requirements**	
	First Task	All Tasks	First Task	All Tasks
Seq-FT	2.5	10.0	1×	1×
Seq-FT+ER	2.5	13.5	1×	2.2×
MTL	2.5	28.7	1×	4×
Routing (Ours)	2.5	10.6	1×	1×

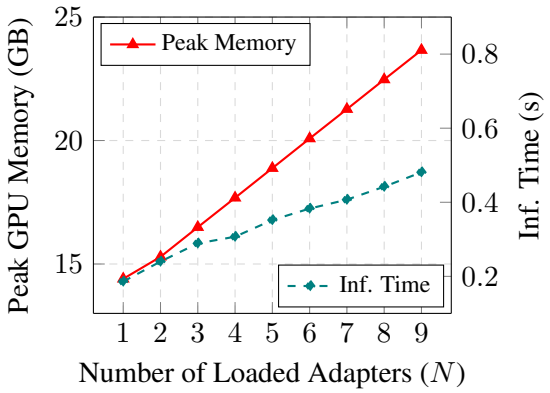


Figure 3: Peak GPU memory (left axis) and median inference time (right axis) as the # of loaded LoRA adapters grows. LLaVA-v1.5-7B on one A100-40GB, 500 samples per N, `max_new_tokens`=1.

Finally, adding XNLI (Conneau et al., 2018) a text-only natural language inference task closely related to the multimodal SNLI. Interestingly, the addition of XNLI not only boosted performance on its own task but also improved scores on related image-text tasks (**80.3** → **81.2**). This suggests a *positive transfer* effect, where the routing mechanism effectively leverages cross-modal similarities to enhance reasoning across related domains.

Beyond positive transfer, we verify in Appendix F that routing vectors are discriminative across tasks, with each router occupying a distinct region of the activation space, and that performance remains stable under L2-normalized routing, indicating robustness to score-scale bias.

Computational Efficiency. We analyze the computational advantages of our routing based approach compared to standard continual learning methods and existing routing baselines. Table 5 summarizes training time and data requirements for InternVL2-2B across a four-task sequence.

As shown in Table 5, our framework is significantly more efficient than MTL and Experience Replay (ER). Beyond these standard baselines, our method offers two distinct advantages over recent

modular approaches that utilize external router modules. First, our method is data-independent. Existing routing frameworks store past task samples (e.g., 20 per task) (Zhao et al., 2025) to train a centralized router, while our routing vectors are trained in isolation per task with zero access to prior data. Second, our method requires only single-pass inference, whereas methods such as MR-LoRA separate routing and generation, using an initial forward pass to determine the appropriate expert before a second pass produces the response. Our routing mechanism allows expert selection and decoding to occur within the same computational graph, so inference requires only a single decoder pass.

VRAM and Latency Scalability. While per-task training cost is fixed, inference memory and latency grow with the number of loaded experts. To quantify this overhead, we profile GPU memory and inference latency on a single NVIDIA A100-40GB across 500 AD test samples per pool size with LLaVA-v1.5-7B (LoRA rank 128). We set `max_new_tokens` to 1 so that the measurement isolates routing and forward-pass overhead from generation-length variance, and we report the median per-sample time (P10 to P90 within 0.02s). As shown in Figure 3, peak memory scales linearly at approximately 1.16 GB per additional expert, utilizing 59% of the 40 GB capacity at $N{=}9$. We measured pool sizes up to $N{=}9$, and linearly extrapolating this slope suggests capacity for approximately 23 experts on an A100-40GB and 57 on an A100-80GB. The median per-sample time grows from 0.186s at $N{=}1$ to 0.481s at $N{=}9$, a $2.6\times$ increase for a $9\times$ larger expert pool. Because token-level routing activates only the top-k experts per token, offloading rarely used experts to CPU memory is a natural way to bound VRAM for longer task sequences, which we leave to future work.

5.5 Routing Enables Cross-modal Transfer

Building upon the positive transfer observed between the text-only XNLI and multimodal SNLI tasks in Table 4, we further investigate the effectiveness of routing in enabling cross-modal transfer. We evaluate this capability on the multi-lingual math dataset MGSM (Shi et al., 2022), specifically focusing on zero-shot reasoning performance.

As detailed in Table 6, our experimental setup utilizes two distinct LoRA modules: *orca-math*, which is specialized for text-based mathematical reasoning, and *multi-math*, which is trained on both image and text modalities. While the text-centric *orca-math* expert naturally outperforms the multimodal variant on this specific task, the routing mechanism achieves a superior aggregate performance (0.82 vs. 0.78 for English).

Table 6: Cross-modal transfer on MGSM. Routing leverages modality-specific experts to achieve superior performance over individual specialists.

Task	Zero-shot	Expert Models		Routing
		orca-math (T)	*multi-math* (V+T)	
English	0.57	<u>0.78</u>	0.62	**0.82**
Chinese	0.57	<u>0.62</u>	0.58	**0.64**

The routed model surpasses both individual experts on English and Chinese, indicating that routing does not collapse to selecting a single module but draws on the text-only expert for arithmetic reasoning and the multimodal expert for grounded inputs within the same generation. The two skills compose at the token level rather than competing for a shared parameter update, allowing the combined model to exceed either specialist. Composition can also hurt when a selected expert is itself worse than the base model, and we analyze such a case in Appendix C.

6 Conclusion

In this work, we address the challenge of catastrophic forgetting in MLLMs by adapting token-level routing to integrate new task-specific experts while preserving foundational knowledge. Our approach distinguishes itself from traditional multi-task learning and experience replay by maintaining fixed training cost per task and eliminating the dependency on historical data, thus avoiding the linear growth in training cost that limits large-scale continual learning. Through extensive evaluations, we demonstrate that routing performs on par with multi-task learning on specialized tasks while protecting general-purpose reasoning capabilities. We also show that this modular architecture facilitates cross-modal transfer, allowing unimodal expertise to bolster multimodal reasoning, and exhibits a positive scaling trend from 2B to 8B where larger models demonstrate better resistance to forgetting. Unlike existing routing baselines that require multi-pass inference and data replay, our mechanism operates in a single pass and trains each routing vector without access to prior data.

7 Limitations

While our routing-based framework offers a scalable solution for multimodal continual learning, some limitations merit further investigation. First, the current implementation necessitates the simultaneous loading of multiple LoRA adapters during inference to enable token-level dynamic composition. Although each adapter is parameter-efficient, the cumulative memory footprint scales linearly with the number of tasks, which may present deployment challenges on hardware with restricted VRAM. Our capacity estimates beyond $N{=}9$ are linear extrapolations rather than measurements. Because only the top-k experts are active per token, an LRU scheme that keeps rarely used adapters in CPU memory and pages them into GPU memory on demand could bound the active VRAM footprint independently of the task count. Other options include merging experts whose routing vectors are highly similar (e.g., Science and Medical in Table A.3), reducing the LoRA rank, pruning experts that routing statistics show are rarely selected, and hierarchical routing that first selects a task group and then an expert within it. We have not validated these mechanisms experimentally. Second, our training strategy prioritizes stability by isolating experts, which minimizes forward transfer during the training phase. Because each LoRA module is trained independently on its respective task without access to prior adapters, a new expert cannot be initialized from related prior experts. Transfer re-emerges only at inference time through composition, as when the router combines the SNLI and XNLI experts in Table 4. Third, due to compute constraints our experiments cap at 8B parameters, and whether the favorable 2B to 8B scaling trend extrapolates to 70B-scale models remains an open question. For the same reason, all results come from single training runs, and we do not report variance across seeds. Fourth, the router selects experts by relevance rather than quality, so an expert that is relevant to an input but worse than the base model on it can degrade the composed output (Appendix C). A quality-aware gate that down-weights such experts is a direction for future work, as is initializing new experts from existing ones to enable forward transfer without reintroducing catastrophic forgetting.

References

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, and 1 others. 2022. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736.

John Arevalo, Thamar Solorio, Manuel Montes-y Gómez, and Fabio A González. 2017. Gated multimodal units for information fusion. *arXiv preprint arXiv:1702.01992*.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *Preprint*, arXiv:2308.12966.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, and 45 others. 2025. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 632–642.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. 2019. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*.

Zhe Chen, Weiyun Wang, Hao Tian, Shenglong Ye, Zhangwei Gao, Erfei Cui, Wenwen Tong, Kongzhi Hu, Jiapeng Luo, Zheng Ma, and 1 others. 2024a. How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites. *Science China Information Sciences*, 67(12):220101.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, and 1 others. 2024b. 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.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk,

and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2475–2485.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*.

Damjan Kalajdzievski. 2024. Scaling laws for forgetting when fine-tuning large language models. *arXiv preprint arXiv:2401.05605*.

Ronald Kemker, Marc McClure, Angelina Abitino, Tyler Hayes, and Christopher Kanan. 2018. Measuring catastrophic forgetting in neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. *Advances in neural information processing systems*, 33:2611–2624.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Claudia Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwińska, and 1 others. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Suhas Kotha, Jacob Mitchell Springer, and Aditi Raghunathan. 2023. Understanding catastrophic forgetting in language models via implicit inference. *arXiv preprint arXiv:2309.10105*.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. 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.

Bin Lin, Zhenyu Tang, Yang Ye, Jiaxi Cui, Bin Zhu, Peng Jin, Jinfa Huang, Junwu Zhang, Yatian Pang, Munan Ning, and 1 others. 2024. Moe-llava: Mixture of experts for large vision-language models. *arXiv preprint arXiv:2401.15947*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024a. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 26296–26306.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, and 1 others. 2024b. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, pages 216–233. Springer.

Ahmed Masry, Xuan Long Do, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. 2022. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. In *Findings of the association for computational linguistics: ACL 2022*, pages 2263–2279.

Ahmed Masry, Megh Thakkar, Aayush Bajaj, Aaryaman Kartha, Enamul Hoque, and Shafiq Joty. 2025. Chartgemma: Visual instruction-tuning for chart reasoning in the wild. In *Proceedings of the 31st International Conference on Computational Linguistics: Industry Track*, pages 625–643.

Michael S Matena and Colin A Raffel. 2022. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716.

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. 2021. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2200–2209.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Arindam Mitra, Hamed Khanpour, Corby Rosset, and Ahmed Awadallah. 2024. Orca-math: Unlocking the potential of slms in grade school math. *arXiv preprint arXiv:2402.14830*.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, and 1 others. 2023. Crosslingual generalization through multitask finetuning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15991–16111.

Mohammed Muqeeth, Haokun Liu, Yufan Liu, and Colin Raffel. 2024. Learning to route among specialized experts for zero-shot generalization. *arXiv preprint arXiv:2402.05859*.

Zixuan Ni, Longhui Wei, Siliang Tang, Yueting Zhuang, and Qi Tian. 2023. Continual vision-language representation learning with off-diagonal information. In *International Conference on Machine Learning*, pages 26129–26149. PMLR.

Oleksiy Ostapenko, Zhan Su, Edoardo Maria Ponti, Laurent Charlin, Nicolas Le Roux, Matheus Pereira, Lucas Caccia, and Alessandro Sordoni. 2024. Towards modular llms by building and reusing a library of loras. *arXiv preprint arXiv:2405.11157*.

Shuai Peng, Di Fu, Liangcai Gao, Xiuqin Zhong, Hongguang Fu, and Zhi Tang. 2024. Multimath: Bridging visual and mathematical reasoning for large language models. *arXiv preprint arXiv:2409.00147*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and 1 others. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Vinay Venkatesh Ramasesh, Aitor Lewkowycz, and Ethan Dyer. 2021. Effect of scale on catastrophic forgetting in neural networks. In *International conference on learning representations*.

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

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. 2017. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. 2021. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34:8583–8595.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. 2019. Experience replay for continual learning. *Advances in neural information processing systems*, 32.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, and 1 others. 2022. Language models are multilingual chain-of-thought reasoners. *arXiv preprint arXiv:2210.03057*.

Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, Zifeng Wang, Sayna Ebrahimi, and Hao Wang. 2024. Continual learning of large language models: A comprehensive survey. *ACM Computing Surveys*.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Huiyi Wang, Haodong Lu, Lina Yao, and Dong Gong. 2025. Self-expansion of pre-trained models with mixture of adapters for continual learning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10087–10098.

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

Zirui Wang, Zachary C Lipton, and Yulia Tsvetkov. 2020. On negative interference in multilingual models: Findings and a meta-learning treatment. *arXiv preprint arXiv:2010.03017*.

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

Ning Xie, Farley Lai, Derek Doran, and Asim Kadav. 2019. Visual entailment: A novel task for fine-grained image understanding. *arXiv preprint arXiv:1901.06706*.

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

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

Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual learning through synaptic intelligence. In *International conference on machine learning*, pages 3987–3995. PMLR.

Yuexiang Zhai, Shengbang Tong, Xiao Li, Mu Cai, Qing Qu, Yong Jae Lee, and Yi Ma. 2023. Investigating the catastrophic forgetting in multimodal large language models. *arXiv preprint arXiv:2309.10313*.

Hongbo Zhao, Fei Zhu, Haiyang Guo, Meng Wang, Rundong Wang, Gaofeng Meng, and Zhaoxiang Zhang. 2025. Mllm-cl: Continual learning for multimodal large language models. *Preprint*, arXiv:2506.05453.

Lei Zhu, Fangyun Wei, and Yanye Lu. 2024. Beyond text: Frozen large language models in visual signal comprehension. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27047–27057.

Appendix

A Dataset description and sizes

For our fine-tuning experiments, we have selected a diverse set of datasets that target distinct tasks and test various facets of multimodal learning. These datasets span a range of applications, including image captioning, visual entailment, hate speech detection, and multimodal classification, providing a comprehensive evaluation framework for our models. Below are the datasets used in our experiments:

COCO-Caption dataset consists of over 330,000 images, each annotated with five human-generated captions, originally designed for image captioning tasks. We use an adapted split, which includes 566,747 image-text pairs for training and 5,000 image-text pairs for testing. This provides a rich multimodal resource for exploring the relationships between visual content and natural language descriptions. The performance is reported using the CIDEr (Vedantam et al., 2015) metric, which evaluates the quality of generated captions based on their similarity to reference captions.

SNLI-VE is derived from the Stanford Natural Language Inference (SNLI) (Bowman et al., 2015) corpus, SNLI-VE (Visual Entailment) extends this dataset by including visual contexts. It consists of 529,527 image-text pairs for training, with a test set of 2,000 pairs. These pairs are categorized into three labels: (a) entailment, (b) neutral, and (c) contradiction, allowing models to evaluate their ability to reason about the relationships between visual and textual information. We use accuracy as the evaluation metric for this task.

Hateful Memes is multimodal dataset is specifically designed to evaluate models' ability to detect hateful content. It contains 8,500 training examples and 1,000 test examples, with each example combining text and images. The task requires models to interpret both modalities to accurately identify hate speech. The dataset includes two labels: (a) not-hateful and (b) hateful. Accuracy is used as the performance metric for this dataset.

MMIMDb is aimed at evaluating multimodal movie genre classification. It contains pairs of movie plot texts and posters, with 15,552 training examples and 2,592 testing examples. The dataset is similar to a multilabel classification task, with 25 possible genre labels. We evaluate the models using the precision metric, which measures the proportion of true positive predictions out of all positive predictions.

These datasets represent a broad spectrum of multimodal tasks, enabling us to assess our models' ability to handle diverse challenges that involve integrating both visual and textual information. We utilize a variety of performance metrics—such as CIDEr, accuracy, and precision—to comprehensively evaluate the models' capabilities across different tasks.

A.1 Benchmarks

In addition to evaluating metrics on fine-tuning tasks, we conduct an in-depth analysis of the fine-tuned model using well-established benchmarks to assess the extent of forgetting introduced during model training. For this analysis, we utilize the MMBench (Liu et al., 2024b), ChartQA (Masry et al., 2022), and DocVQA (Mathew et al., 2021) benchmarks, each targeting distinct evaluation objectives.

MMBench is designed to assess a model's performance on a set of fine-grained abilities, encompassing 20 ability dimensions under perception and reasoning. It uses multiple-choice questions that challenge the model to demonstrate both fine-grained understanding and reasoning skills.

ChartQA evaluates the model's ability to perform complex reasoning over data visualizations such as charts and graphs. The questions in this evaluation often require logical and arithmetic reasoning, as well as an understanding of the visual features of the chart, which tests a model's ability to interpret and reason about structured data.

DocVQA Focused on visual question answering (VQA) tasks involving document images, this dataset requires models to comprehend and reason over the structure and content of documents. It assesses the model's ability to interpret visual and textual information in the context of documents, a challenge unique to VQA in document-specific scenarios.

These benchmark datasets are crucial for evaluating the robustness and generalization capabilities of our models, particularly in scenarios that involve more complex reasoning or domain-specific tasks, such as visual question answering in documents and data visualization comprehension. Additionally, they provide a means of assessing the potential for CF in the fine-tuned models, ensuring their continued effectiveness across various domains.

B Routing patterns

In Figure A.1 (left), we examine the routing pattern for the SNLI task. The results clearly indicate that the SNLI module is correctly activated. However, we also observe the activation of the XNLI module, which highlights the ability of the routing vector to understand task similarity. This is intuitive, as XNLI is a text-text entailment task, while SNLI is a text-image entailment task. This pattern is consistent with the findings in Table 4, where we observed that adding the XNLI module enhanced the performance of the SNLI task, likely due to shared underlying features between the two tasks. In Figure A.1 (right), we present the routing pattern for the COCO task. Here, we observe that only the COCO module is activated, demonstrating that the routing vectors are correctly trained. This ensures that requests are routed automatically and accurately to the appropriate modules, validating the robustness of the routing.

In Figure A.1 (middle), we present the routing pattern for the benchmark task MMBENCH. As this is a general-purpose task, we observe that no LoRA module receives dominant activation affinity scores are approximately uniformly distributed across experts, causing the selected top-k expert to receive very low weights. Consequently, the base model output dominates the final prediction (Eq. 5), effectively defaulting to zero-shot behavior.

C Routing in cross-lingual setting

In this section, we investigate whether routing can enable cross-lingual transfer—specifically, whether a routing model can solve a task in one language by combining a LoRA module trained on that language with another LoRA module trained on a specialized task in English. To evaluate this, we focus on the MGSM dataset (Shi et al., 2022), a multilingual math-based reasoning benchmark. We construct two expert models: a Math Expert, trained on the OrcaMath dataset (Mitra et al., 2024), and a Chinese Language Expert, trained on the xP3mt dataset (Muennighoff et al., 2023).

Table A.1 presents results for InternVL2-2B and InternVL2-8B models. The 2B model benefits from routing, achieving the highest score when both the Math and Chinese experts are combined, demonstrating effective cross-lingual transfer. However, the 8B model struggles with routing, showing a significant performance drop when both experts are used together.

We attribute the 8B failure to expert quality rather than to routing. The 8B Chinese expert scores 0.53, below the 0.57 zero-shot baseline, so fine-tuning on xP3mt interferes with the model's already strong pretrained multilingual capability. Because this expert is relevant to Chinese inputs, the router selects it, and composing it with the Math expert (0.62) pulls the result down to 0.47. By contrast, when each component expert is individually beneficial, composition yields positive transfer, as in the 2B Chinese subset (0.22 vs. 0.16 and 0.20), English MGSM in Table 6, and SNLI with XNLI in Table 4.

This case shows that degradation requires two conditions to hold at once. The added expert must be relevant to the input, so that the router selects it, and it must be worse than the base model on that input. An irrelevant expert does not cause harm because its routing vector does not match the input activations, which is consistent with the stable performance after adding the unrelated ChartGemma expert in Table 4. The current router gates on relevance rather than quality, so a quality-aware gate that down-weights experts failing to beat the base model on their own domain would address this failure mode, and a better data mixture for the language expert may also help.

Figure A.2 reveals an interesting pattern in cross-lingual transfer. As expected, the language model is more frequently utilized in the initial and later layers, while the math expert plays a dominant role in performing computations. We believe this aligns with the expected behavior, where the initial layers transform embeddings from language to math, and the later layers map from the math space back to language.

Table A.1: Comparison of routing models for cross-lingual transfer on the MGSM 8-shot dataset, evaluating different scales of the InternVL2 model on the Chinese subset.

Model Scale	zero-shot	**Expert Models**		Routing
		Chinese	Math	
InternVL2-2B	0.15	0.16	<u>0.20</u>	**0.22**
InternVL2-8B	0.57	<u>0.53</u>	**0.62**	0.47

D Training details

For the full finetuning baselines, we use a learning rate of 1×10^{-5}. We employ a total of 32 A10

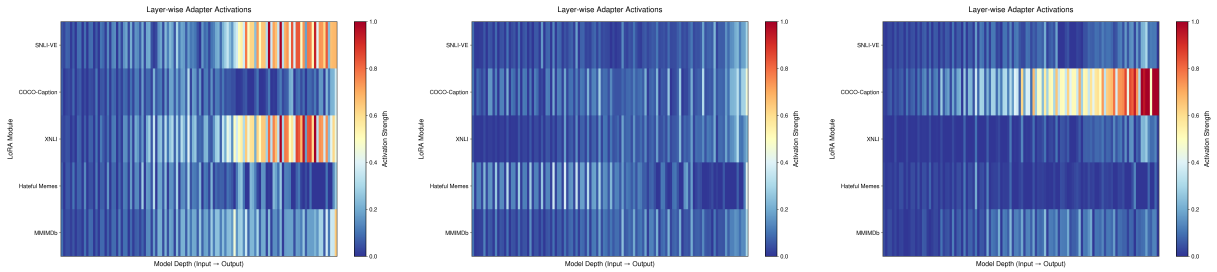


Figure A.1: Routing patterns for SNLI (left), MMBENCH (middle), and COCO (right). The figure demonstrates that the routing model correctly activates task-specific modules: SNLI and XNLI for SNLI, only COCO for COCO, and no module for MMBENCH, where the base model handles the task. These confirm the routing mechanism's effectiveness in selecting the appropriate module based on the task.

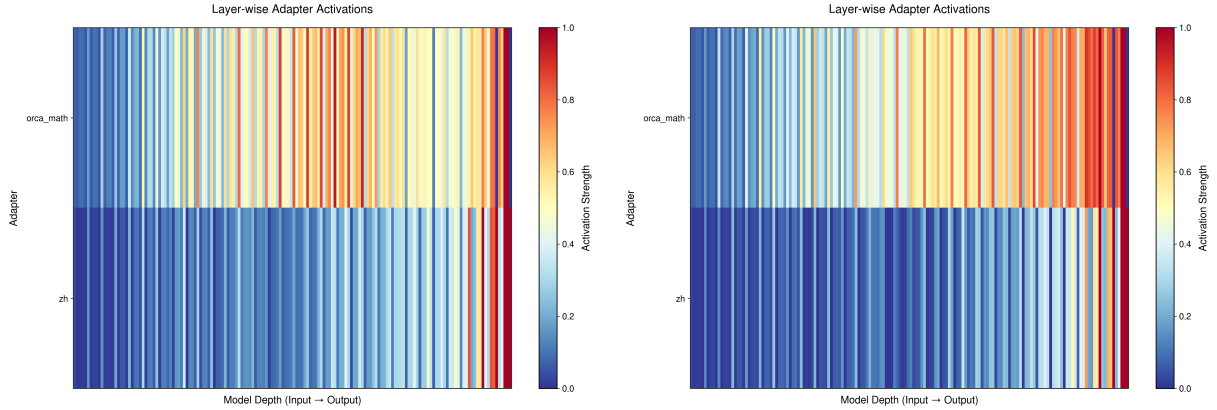


Figure A.2: Routing patterns for the MGSM dataset in multilingual transfer. Notably, the model leverages the Chinese expert in the early and later stages, while the math expert is primarily used for computation.

GPUs for training these models. The batch size for training is set to 512, and we freeze the ViT and projection layers, training only the language model's weights. All models are trained using a cosine learning rate scheduler, and we leverage the ZeRO-3 configuration to offload gradients, optimizer states, and parameters.

Table A.2 lists the settings for the routing framework, which are shared across InternVL2 and LLaVA-v1.5 and across all tables, including the MLLM-CL comparison. For that comparison, we re-implement MR-LoRA with rank-128 experts so that expert capacity is matched.

For sequential learning, we follow the task progression: COCO-Caption → SNLI-VE → Hateful Memes → MMIMDb. This sequence allows us to analyze the impact of each approach across diverse datasets and tasks.

Table A.2: Hyperparameters of the routing framework.

Setting	**Value**
LoRA rank	128
LoRA insertion	All LLM linear layers (q, k, v, o, gate, up, down)
Frozen components	Vision encoder, projector, base LLM
Expert learning rate	4×10^{-5}, cosine decay, 1 epoch
Global batch size	512 (32 A10 GPUs, ZeRO-3)
Routing vector training	100 steps, batch size 32, LoRA frozen
Routing vector init	Zero
Top-k	1
Softmax temperature τ	0.1
Decoding	Greedy

E Comparison with Mixture-of-Experts Architectures

While our routing approach shares architectural similarities with Mixture of Experts (MoE) systems like MoE-LLaVA (Lin et al., 2024), it differs in both purpose and implementation. Traditional MoE approaches improve model capacity through joint training of specialized experts within a unified architecture, requiring simultaneous access to all

training data and joint optimization of experts and gating networks.

Our method diverges from this paradigm in three key aspects. First, our experts (LoRA modules) are trained independently and sequentially, with each expert specializing on a single task without access to other tasks' data. Second, we explicitly decouple expert learning from routing by first training task-specific LoRA modules, then learning lightweight routing vectors while keeping LoRA parameters frozen, unlike traditional MoEs where expert structure and routing are jointly optimized. Third, our approach requires only new task data, eliminating the need to retain or synthesize previous task data.

F Router Discriminability Analysis

A natural concern with independently trained routing vectors is whether they remain discriminative as the expert pool grows, and whether score-scale biases across tasks could corrupt top-k selection. Since each routing vector is trained only on positive samples from its own task (Eq. 2), it is important to verify that the resulting vectors encode task-specific structure rather than generic activation patterns, and that their magnitudes remain comparable enough for fair competition at inference time.

F.1 Pairwise similarity between routing vectors

To probe whether the routing vectors encode task-specific representations, we compute the pairwise cosine similarity and L2 distance between the learned routing vectors across the five domains of the MLLM-CL benchmark, averaged across all layers of the LLM backbone.

Table A.3 reports the cosine similarity matrix. Inter-task similarity remains consistently low, ranging from 0.113 to 0.282, while self-similarity is naturally 1.0. This indicates that each task's routing vector occupies a distinct region of the representation space rather than collapsing into a shared generic direction.

Table A.4 corroborates these findings with pairwise L2 distances, which show substantial separation between all task pairs.

The patterns observed are intuitive and semantically meaningful. Finance and Remote Sensing—two visually and semantically disparate domains (stock charts vs. satellite imagery) exhibit the lowest cosine similarity (0.113) and correspondingly smaller L2 distance is driven purely by magnitude rather than direction. Conversely, Science and Medical, which share visual characteristics such as scientific diagrams and structured visual reasoning, show relatively higher similarity (0.282). This suggests that the router intuitively encodes inter-domain semantic relationships, with closer vectors corresponding to tasks that share reasoning patterns or visual structure.

Table A.3: Pairwise cosine similarity between routing vectors across MLLM-CL domains, averaged over all layers. Low off-diagonal values confirm that each routing vector encodes task-specific structure.

	AD	**Fin**	**Med**	**RS**	**Sci**
AD	1.000	0.191	0.203	0.195	0.240
Fin	0.191	1.000	0.174	0.113	0.264
Med	0.203	0.174	1.000	0.174	0.282
RS	0.195	0.113	0.174	1.000	0.175
Sci	0.240	0.264	0.282	0.175	1.000

Table A.4: Pairwise L2 distance between routing vectors across MLLM-CL domains, averaged over all layers.

	AD	**Fin**	**Med**	**RS**	**Sci**
AD	0.000	0.217	0.308	0.252	0.329
Fin	0.217	0.000	0.265	0.178	0.297
Med	0.308	0.265	0.000	0.292	0.343
RS	0.252	0.178	0.292	0.000	0.323
Sci	0.329	0.297	0.343	0.323	0.000

F.2 Robustness to score-scale bias

A secondary concern is whether differences in the magnitudes of independently trained routing vectors could bias top-k selection toward high-norm experts, irrespective of semantic relevance. To directly eliminate this potential confound, we evaluate a variant (*Routing-norm*) in which all routing vectors are L2-normalized before computing affinity scores, removing any influence of magnitude on expert selection.

Table A.5: Comparison of standard routing against the L2-normalized variant (Routing-norm) on the MLLM-CL benchmark using LLaVA. Differences are at most 0.4 points across all domains with no consistent direction, indicating robustness to score-scale bias.

Method	**RS**	**Med**	**AD**	**Sci**	**Fin**
MR-LoRA	80.9	65.3	**54.1**	**56.7**	**91.1**
Routing (Ours)	**81.0**	**66.6**	53.4	56.4	**91.1**
Routing-norm	80.9	66.3	53.8	56.2	90.9

As shown in Table A.5, the performance differences between the two variants are at most 0.4 points across all five domains, with no consistent direction of change some domains marginally favor the unnormalized variant, others favor the normalized one. Both variants remain near Oracle-level performance. Since L2 normalization provably eliminates norm-induced bias by construction and introduces no performance degradation, it can be adopted as a strictly safer default for deployments where the expert pool is expected to grow to larger N, providing a guaranteed bound against score-scale miscalibration at zero computational or accuracy cost.

G Results on Qwen3-VL

To test whether routing generalizes beyond InternVL2 and LLaVA-v1.5, we repeat the four-task sequence on Qwen3-VL-4B (Bai et al., 2025), a more recent model family with a different vision encoder and projector design. All methods use the same training data and rank-128 LoRA experts inserted in the LLM, with the vision encoder and projector frozen.

Table A.6: Continual learning on Qwen3-VL-4B with LoRA in the LLM only. Passes is the number of forward passes per query at inference.

Method	Passes	COCO	SNLI	HM	MMIMDb
Zero-shot	1	0.285	75.2	66.9	34.6
Specialist	1	1.069	<u>83.0</u>	**76.3**	<u>76.7</u>
MTL	1	<u>1.258</u>	80.6	75.4	75.6
MR-LoRA	2	1.052	82.4	**76.3**	76.4
Routing (Ours)	1	**1.370**	**83.5**	75.0	**79.8**

As shown in Table A.6, routing achieves the best score on COCO, SNLI, and MMIMDb with a single forward pass, exceeding both the per-task specialist and the two-pass MR-LoRA baseline on these three tasks. On Hateful Memes, routing trails the specialist and MR-LoRA by 1.3 points.

Adapting the vision encoder and projector. Our main experiments freeze the vision encoder and projector to isolate the routing mechanism. To test whether the method depends on this choice, we additionally insert LoRA experts and routing vectors into the vision encoder and the visual-language projector of Qwen3-VL-4B, giving 356 adapted modules in total as listed in Table A.7. Each module has its own routing vector, trained as in Section 3.

Table A.7: Modules adapted when the vision encoder and projector of Qwen3-VL-4B are also tuned.

Component	Target layers	# modules
LLM	q/k/v/o, gate/up/down projections	252
Vision encoder	24 ViT blocks (qkv, attn.proj, fc1, fc2)	96
Projector	merger and deepstack merger (fc1, fc2)	8
Total		356

Table A.8: Qwen3-VL-4B with LoRA experts in the LLM, vision encoder, and projector.

Method	COCO	SNLI	HM	MMIMDb
Single expert	1.191	**80.9**	72.9	72.0
Routing (Ours)	**1.344**	78.8	**75.1**	**80.9**

As shown in Table A.8, training remains stable with vision-side experts, and routing improves over the single expert on COCO, Hateful Memes, and MMIMDb while trailing it by 2.1 points on SNLI. Relative to the LLM-only setting in Table A.6, routing stays within 0.03 CIDEr on COCO and within 1.1 points on Hateful Memes and MMIMDb, while SNLI drops by 4.7 points. The tasks used here consist of natural images that are well covered by the pretrained vision encoder, so large gains from vision-side adaptation are not expected. We expect vision-side experts to matter more on domains with a larger visual shift, such as medical imaging or remote sensing, and leave this to future work.