Dynamic Routing Between Experts: A Data-Efficient Approach to Continual Learning in Vision-Language Models

Jay Mohta*, Kenan Emir Ak*, Dimitrios Dimitriadis, Yan Xu, Mingwei Shen

Amazon.com

{*jaymoht, kenanea, dbdim, yanxuml, mingweis*}@amazon.com

* equal contribution

Abstract

Vision-Language Models (VLMs) suffer from catastrophic forgetting when sequentially fine-tuned on new tasks, degrading performance on previously learned foundational and task-specific capabilities. While multi-task learning can mitigate forgetting, it requires simultaneous access to all datasets and imposes computational overhead that scales linearly with the number of tasks. In this work, we introduce a routing-based approach that enables the integration of new tasks while preserving the foundational knowledge acquired during pretraining. We evaluate our method using InternVL-2 models (2B and 8B parameters) and demonstrate that routing preserves the model's foundational capabilities by maintaining performance on general-purpose benchmarks such as ChartQA, MMBench, and DocVQA, while simultaneously improving accuracy on specialized tasks. Importantly, our approach achieves this without requiring concurrent access to data from all tasks, avoiding the significant computational and data overhead associated with traditional multi-task learning. We further conduct extensive ablation studies to evaluate the scalability and robustness of routing-based learning, showing that the approach is resilient to a growing number of tasks and performs particularly well when new tasks are semantically related. Finally, we show that the routing mechanism enables superior cross-modal transfer between language and vision capabilities, allowing knowledge learned in one modality to enhance performance in another capability not achieved by existing continual learning methods.

Introduction

Large Language Models (LLMs) have advanced the state-of-the-art in Natural Language Processing (NLP), demonstrating capabilities in tasks such as machine translation, sentiment analysis, and question answering (Brown et al. 2020; Chowdhery et al. 2023). Building on advances in language modeling and computer vision, Vision-Language Models (VLMs) have emerged as multimodal systems capable of joint reasoning over visual and textual inputs (Radford et al. 2021; Lu et al. 2019), enabling applications from image captioning to visual question answering (Chen et al. 2020).

Although pretrained VLMs offer strong general-purpose capabilities, fine-tuning them on specialized-domain tasks (such as medical imagery, legal documents) is often required to achieve top performance on those specialized domains. However, this fine-tuning frequently triggers Catastrophic Forgetting (CF)—a phenomenon where the model starts to loose general purpose capabilities acquired from large-scale multimodal corpora (Kirkpatrick et al. 2017; Kemker et al. 2018; Goodfellow et al. 2013).

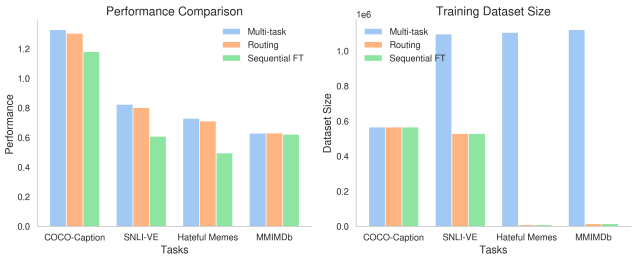

Figure 1: Comparison of Multi-Task Learning, Routing-Based Methods and Sequential FT in mitigating catastrophic forgetting. Routing-based methods match multi-task learning performance while requiring only new task data, avoiding costly retraining on previous tasks. Tasks are introduced sequentially from left to right, with the multi-task model trained on the cumulative data from all prior tasks. In contrast, sequential fine-tuning only sees new task data, leading to strong performance on the most recent task but potential forgetting of earlier ones.

Recent work shows that CF severity increases with model scale (Luo et al. 2023), making this particularly challenging for LLMs. Kalajdzievski (2024) et al. further establishes precise scaling laws: the extent of forgetting scales predictably with both the number of fine-tuned parameters and the number of update steps, even under parameter-efficient paradigms such as Low-Rank Adaptation (LoRA) (Hu et al. 2022). In multimodal settings, (Zhai et al. 2023) demonstrate that fine-tuning large VLMs on one modality (e.g., visual tasks) degrades performance in the other (e.g., language tasks), reinforcing the interference between modalities during sequential adaptation.

A common strategy to mitigate CF is multi-task learning, where all tasks are trained jointly to preserve shared representations (Ruder 2017). While this setup often serves as an upper bound for continual learning performance—since it assumes access to all task data—it is computationally expensive and infeasible in many real-world settings due to the

need for concurrent dataset access and large-scale infrastructure.

Continual learning offers a practical alternative to multi-task learning and has been explored through several strategies, including replay-based methods (Rolnick et al. 2019; Shin et al. 2017), regularization techniques (Kirkpatrick et al. 2017; Zenke, Poole, and Ganguli 2017), knowledge distillation (Li and Hoiem 2017; Rebuffi et al. 2017), and parameter-isolation approaches (Rusu et al. 2016; Mallya and Lazebnik 2018; Wortsman et al. 2020; Mallya, Davis, and Lazebnik 2018; Mallya and Lazebnik 2022). While effective in smaller-scale models or unimodal settings, these methods often require access to previous task data or model checkpoints, or introduce architectural changes that hinder scalability to billion-parameter vision-language models (VLMs) (Chen et al. 2024b; Liu et al. 2023). To address these scalability challenges, recent work SEMA (Wang et al. 2024a) has turned to adapter-based techniques, which offer a modular and lightweight alternative by enabling task-specific tuning without altering the core model. While SEMA supports sequential training without requiring access to all task data simultaneously, it remains limited to single-modality settings and coarse layer-level adaptation.

To address the scalability and data access limitations of traditional continual learning methods, model merging has emerged as a promising modular alternative. These techniques combine the weights of separately fine-tuned models to integrate new capabilities into a base model—without requiring concurrent access to all task data (Matena and Raffel 2022; Douillard et al. 2021; Ilharco et al. 2023; Merlingot et al. 2024; Cha et al. 2020). By operating directly on model parameters, merging offers an efficient way to incorporate task-specific knowledge while avoiding catastrophic forgetting. This makes it particularly attractive for large-scale vision-language models, where full retraining or replay is infeasible. However, merging often underperforms compared to joint multi-task training, especially when tasks are semantically distant or involve modality-specific reasoning, leading to interference and degraded performance in the merged model.

Another family of modular approaches—routing-based methods focuses on dynamically selecting specialized sub-networks based on input characteristics (Li et al. 2018; Ostapenko et al. 2024; Muqeeth et al. 2024). While conceptually related to Mixture of Experts (MoE) architectures, these methods differ in both design goals and constraints. MoE models for vision-language tasks, such as MoE-LLaVA (Lin et al. 2024) and V-MoE (Riquelme et al. 2021), primarily aim to enhance model capacity and computational efficiency through parallel expert networks activated by learned gating mechanisms (Shazeer et al. 2017; Fedus, Zoph, and Shazeer 2021). However, MoE architectures typically require simultaneous access to all datasets during training and focus on improving zero-shot generalization rather than preserving previously acquired knowledge. Similarly, existing routing-based approaches have predominantly been developed to improve generalization to novel inputs, leaving a significant gap in their application to continual learning scenarios where maintaining performance on previous tasks is essential.

In this work, we adapt routing-based modularity specifically to address catastrophic forgetting in large-scale vision-language models. We systematically investigate token-level routing within large-scale vision-language models as a strategy to overcome forgetting. Unlike traditional multi-task learning, which relies on joint training over all datasets, our approach supports incremental learning using only the current task's data—while achieving performance comparable to multi-task baselines, illustrated in Figure 1. We evaluate the approach across multiple sequential tasks and analyze its scalability with increasing model size. Our experiments demonstrate that as model size grows, the performance gap between our routing method and fully specialized models narrows. Furthermore, we highlight the unique ability of token-level routing to facilitate cross-modal transfer, enabling knowledge gained in one modality to enhance performance in another—a critical capability not achieved by existing continual learning methods.

Preliminaries

We first review foundational concepts relevant to continual learning in vision-language models, including the underlying model architecture and common continual learning strategies such as multi-task learning, sequential fine-tuning and expert merging.

Vision Language Models

Vision-Language Models (VLMs) process multimodal inputs by encoding visual content through a visual encoder V and projecting these features into a shared embedding space via projector F, where visual features and textual embeddings are fused and jointly processed by a transformer-based language model for multimodal reasoning.

Multi-task Learning

Multi-task learning (MTL) mitigates catastrophic forgetting by training on all specialized tasks simultaneously, enabling shared representations that generalize across tasks (Caruana 1997). While MTL serves as a performance upper bound for CL, it is often computationally prohibitive due to simultaneous access and training on all task data.

Sequential Fine-tuning

Sequential fine-tuning adapts pre-trained models to new tasks incrementally, making it computationally efficient but highly susceptible to catastrophic forgetting (Goodfellow et al. 2013). Techniques like experience replay (Rolnick et al. 2019) improve knowledge retention but require storing previous data, limiting their practicality for large-scale applications. Other approaches to mitigate catastrophic forgetting include regularization-based methods (Kirkpatrick et al. 2017; Zenke, Poole, and Ganguli 2017) and parameter isolation techniques (Rusu et al. 2016; Mallya and Lazebnik 2018), but these often involve architectural modifications or require access to prior task data, making them challenging to scale to billion-parameter vision-language models.

Expert Merging

Model merging combines multiple task-specific models into a single aggregate model. Starting with common architecture and initial weights θ_0, merging integrates specialized knowledge from models with weights $\theta_1, \theta_2, \ldots, \theta_n$. We explore two strategies:

Model averaging (McMahan et al. 2017) uses an element-wise average of model parameters: $\theta_m = \frac{1}{n}\sum_{i=0}^{n}\theta_i$

Task Arithmetic (Ilharco et al. 2023) constructs task vectors and combines them with the base model: $\theta_m = \theta_0 + \sum_{i=1}^{n}\lambda_i t_i$ where $t_i = \theta_i - \theta_0$

These methods require only the dataset of the new specialized task, making them cost-effective alternatives for mitigating catastrophic forgetting.

Approach

Routing between experts is a modular machine learning paradigm that leverages a routing function to dynamically direct inputs to specialized expert modules based on input embeddings. Prior works such as PHATGOOSE (Muqeeth et al. 2024) and Arrow (Ostapenko et al. 2024) have primarily explored routing to enable zero-shot generalization in large language models by dynamically selecting expert modules at inference.

While most routing algorithms have been explored in the context of LLMs, we adapt them for VLMs with a different objective: mitigating CF rather than enabling zero-shot generalization. Building on the superior performance of PHATGOOSE (Muqeeth et al. 2024), we extend its design by incorporating specialized LoRA modules tailored to VLM architectures, and implement a routing function that dynamically selects task-specific adapters during inference. Below, we detail how PHATGOOSE's routing mechanism is applied within the vision-language domain.

PHATGOOSE for VLMs. Building on the VLM architecture described previously, we implement task-specific LoRA adapters within the language model component while keeping the vision encoder V and projection layer F frozen. This design choice enables a more controlled evaluation of catastrophic forgetting, isolating the impact of adaptation within the language model and minimizing confounding factors.

Setup. In this setup, LoRA modifies the output of each linear layer in the language model while keeping the base parameters frozen. Specifically, for a linear layer with weights $W \in \mathbb{R}^{d\times n}$, given an input activation $u_t \in \mathbb{R}^n$, the output is computed as:

$$Wu_t + BAu_t,$$

where $A \in \mathbb{R}^{r\times n}$ and $B \in \mathbb{R}^{d\times r}$ are the trainable LoRA parameters, while W remains frozen during fine-tuning. Here, u_t denotes the input activation corresponding to the t-th token in the sequence. The dimensions d and n represent the output and input sizes of the linear layer, respectively, and $r \ll \min(d, n)$ is a small rank hyperparameter that controls the number of trainable parameters.

Training. Our training procedure consists of two sequential stages designed to enable modular adaptation without modifying the base model or requiring access to previous tasks. First, we train task-specific LoRA modules that specialize the language model for each new task independently. During this stage, all base model parameters, including the vision encoder and core language model weights, remain frozen. Only the LoRA parameters are updated using data exclusively from the current task. This isolates task adaptation and prevents interference.

After the LoRA modules are trained, we introduce lightweight routing vectors—one per task—that serve as gating mechanisms to dynamically control the influence of each LoRA module during inference. Crucially, each routing vector is trained independently using only the task-specific data and the corresponding LoRA module, without accessing other tasks' data or parameters. Initialized to zero, the routing vector is optimized with the same next-token prediction objective used for LoRA training, while keeping all other parameters—including the base model and all LoRA modules—frozen. Through this process, the routing vector learns to identify input activation patterns unique to its task and applies gating on a token-level basis via a sigmoid function, effectively controlling the task-specific adaptation dynamically.

Specifically, the output of a linear layer augmented with routing and LoRA modules is computed as:

$$Wu_t + BAu_t\sigma(v^T u_t),$$

where W is the frozen base weight matrix, A and B are the LoRA parameters trained per task, u_t is the input activation at token t, and $\sigma(v^T u_t)$ is the sigmoid gating function controlled by the routing vector v.

We assume tasks arrive sequentially. For each new task, only the corresponding LoRA module and routing vector are trained using data from that task alone. During specialization, no data or parameters from previous tasks are accessed or modified, thereby enforcing continual learning constraints and enabling efficient adaptation without catastrophic forgetting.

Inference. To enable dynamic routing during inference, we load all the LoRA modules and gating vectors trained for different tasks. The gating vectors $v_1, v_2, \ldots, v_n$ correspond to tasks $\{1, 2, \ldots, n\}$. All routing decisions and module activations are computed independently for each token in the input sequence, ensuring fine-grained token-level routing. At each layer, routing proceeds as follows:

- Compute the affinity score between the input activation u_t and each gating vector v_i via dot product:

$$\alpha_{t,i} = v_i^T u_t.$$

- Select the top-k modules with the highest affinity scores, where k is a user-defined hyperparameter.
- Normalize these scores using a softmax scaled by $\sqrt{n}$, where n is the input dimension, to obtain weights:

$$w_t = \text{softmax}\left(\left\{\frac{\alpha_{t,i}}{\sqrt{n}} : i \in E_t\right\}\right),$$

where E_t is the set of selected top-k modules.

The output of the routed linear layer at token position t is then computed as:

$$Wu_t + \sum_{i \in E_t} w_{t,i} B_i A_i u_t,$$

where W is the frozen base weight matrix, and A_i, B_i are the LoRA parameters for module i.

Crucially, this inference design respects continual learning constraints: it does not require access to any previous task data or task identifiers. New tasks can be added simply by training their respective LoRA modules and routing vectors without altering existing modules. The routing vectors implicitly learn to recognize input patterns and gate module activation, thus preventing interference between tasks and mitigating catastrophic forgetting while preserving the base model's foundational capabilities.

Experiment Design

In this section, we present the models, datasets, training hyperparameters, and strategies employed to evaluate our continual learning framework using routing-based approaches.

Models

We use InternVL-2 (Chen et al. 2024b), one of the most powerful open-source VLM in our experiments. The InternVL-2 family includes models ranging from 1B to 108B parameters. The model supports multimodal input, handling text, images, video. We utilize 2B and 8B versions of InternVL-2 for our experiments on CF.

Datasets and Evaluations

To assess CF across different methods, we utilize a combination of specialization tasks and benchmark tasks. Specialization tasks involve fine-tuning the model to enhance performance on specific tasks, while benchmark tasks evaluate the model's original proficiency, providing a measure of its general-purpose capability. Achieving strong performance on both specialization and benchmark tasks is critical for any method because that would mean the model gains new knowledge while preserving the older information.

For specializing tasks, we utilize several datasets that cover a diverse range of multimodal applications such as image captioning, visual entailment, hateful content detection, and multimodal classification, offering a comprehensive framework for evaluating the effectiveness and robustness of our models. The COCO-Caption (COCO) dataset (Lin et al. 2015), focused on image captioning, is evaluated using the CIDEr metric. SNLI-VE (SNLI) (Xie et al. 2018), a text-image entailment task, is assessed based on accuracy. The Hateful Memes (HM) dataset (Kiela et al. 2021), which focuses on detecting hateful memes, is also evaluated using accuracy. Finally, we use the MMIMDb dataset (Arevalo et al. 2017) for movie genre classification, with performance measured by precision.

To quantify the model's retained capabilities post-specialization, we utilize three benchmark tasks. MMBench (MMB) (Liu et al. 2024) evaluates preservation of fine-grained perception and reasoning abilities across 20 dimensions, ChartQA (Masry et al. 2022a) tests retention of data visualization comprehension skills, and DocVQA (Mathew, Karatzas, and Jawahar 2021b) assesses document understanding capabilities. This evaluation strategy enables us to assess both the model's ability to acquire new specialized knowledge and its capacity to retain previously learned skills.

As our routing-based framework enables zero-shot cross-modal transfer, we conduct experiments training two expert models: one specialized on the text modality using the Orca-Math dataset (Mitra et al. 2024), and another trained on the combined image+text modality using the Multi-math dataset (Peng et al. 2024). For zero-shot evaluation, we employ the MGSM framework (Shi et al. 2022) on both English and Chinese subsets to measure cross-lingual and cross-modal generalization.

Training

For the full finetuning, we use a learning rate of $1 \times 10^{-}5$ and train the models for a single epoch, unless stated otherwise. We employ total of 32 A10 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.

The models are trained using a cosine learning rate scheduler to adjust the learning rate over the course of training. For specializing LoRA module, we employ a learning rate of 4×10^{-5}. LoRA modules are added exclusively to the LLM, ensuring that only the LoRA weights of the LLM are updated during training. To train the gating module we freeze all the parameters and only train the gate vector v for 100 steps on the specialized task.

For sequential learning, we follow the task progression: COCO-Caption $\rightarrow$ SNLI-VE $\rightarrow$ Hateful Memes $\rightarrow$ MMIMDb. This sequence allows us to analyze the impact of each approach across diverse datasets and tasks. During inference we set the top-k hyperparameter in $k = 1$ for all of our experiments.

Results

In this section, we present the results of our study organized into two main categories: (1) comparative evaluations of different continual learning methods, and (2) ablation studies examining the properties and capabilities of our routing-based approach.

Continual Learning Comparison

InternVL2-2B Results. We evaluate various continual learning strategies on the InternVL2-2B model across multiple tasks. The approaches include MTL training, sequential fine-tuning, experience replay, model merging, and routing.

As shown in Table 1, InternVL2-2B demonstrates strong zero-shot performance on benchmark tasks, but performs relatively poorly on specialized tasks, highlighting the need for task-specific adaptation. Traditional sequential fine-tuning improves performance on the most recent task but suffers from severe CF, leading to substantial degradation on earlier tasks and benchmarks, consistent with prior findings (Goodfellow et al. 2013; Wang et al. 2024b).

Method	Specialized Tasks				Benchmark Tasks		
	COCO	SNLI	HM	MMIMDb	MMB	ChartQA	DocVQA
Zero-shot	0.795	52.9	60.2	46.6	**79.1**	**59.5**	**84.7**
Sequential FT	1.181	61.0	49.7	62.3	67.1	41.2	70.0
Sequential FT w/ ER (20%)	1.280	58.1	73.8	62.3	70.9	46.6	73.9
Merging: Model Avg.	1.148	72.4	64.1	62.3	78.8	59.2	83.6
Merging: Task Arith.	1.171	73.3	64.5	64.9	78.6	59.2	84.1
Routing	1.305	80.3	71.2	**63.2**	78.8	59.2	84.3
MTL	**1.340**	**83.6**	**73.9**	63.0	76.5	55.7	77.9

Table 1: Comparison of continual learning methods on specialized and benchmark tasks using InternVL2-2B.

To address this limitation, we incorporate experience replay by retaining 20% of data from previous tasks. This approach improves stability over naive sequential fine-tuning but still results in performance drops—particularly on SNLI-VE—and requires storing historical data, which increases computational overhead as tasks accumulate. Both sequential fine-tuning strategies show significant degradation on benchmark tasks, highlighting the model's inability to retain general-purpose knowledge.

Next, we explore an efficient modular approach: model merging. We evaluate two strategies: model weight averaging and task vector arithmetic. These methods outperform traditional sequential fine-tuning while eliminating the dependency on historical data, requiring only the adaptation of the base model to new tasks. These methods require only task-specific adaptation of the base model and effectively preserve the VLM's general-purpose capabilities, with performance closely matching the zero-shot baseline. However, despite these advantages, a notable performance gap remains when compared to MTL on specialized tasks.

In contrast, our routing-based approach achieves performance on specialized tasks that is comparable to or better than MTL, while preserving the model's original benchmark performance. For example, on MMIMDb, routing even surpasses MTL, possibly due to reduced negative task interference (Wang, Lipton, and Tsvetkov 2020). Furthermore, while MTL suffers a substantial drop on benchmark tasks—indicating loss of general-purpose capabilities—routing maintains performance near the zero-shot baseline. This demonstrates that routing not only mitigates catastrophic forgetting, but also balances specialization and generalization without the computational and data burdens of multi-task training.

InternVL2-8B Results. Table 2 extends our evaluation to the larger InternVL2-8B model, focusing on the top-performing methods from the 2B setting: task arithmetic, routing, and MTL. At this scale, routing matches MTL in performance, further closing the gap observed with the smaller model, while maintaining benchmark performance without degradation. Notably, routing not only preserves general-purpose capabilities but also surpasses the zero-shot baseline—highlighting that its effectiveness scales with model size. These results suggest routing becomes increasingly advantageous as model capacity grows.

Method	Specialized Tasks				Benchmark Tasks		
	COCO	SNLI	HM	MMIMDb	MMB	ChartQA	DocVQA
Zero-shot	0.889	70.2	65.4	54.5	85.6	72.6	**89.7**
Merging: Task Arith.	1.272	71.3	68.7	66.9	86.0	70.6	87.2
Routing	1.350	83.3	76.8	**67.4**	**86.3**	**73.1**	89.3
MTL	**1.374**	**84.7**	**78.3**	65.4	85.7	68.2	86.5

Table 2: Comparison of top-performing continual learning approaches for InternVL2-8B on specialized and benchmark tasks.

Building on findings from Yadav et al. (2024), which show that model merging improves with scale, we also conducted model merging experiments using the 8B model. Compared to the 2B setting, merging indeed yielded better performance. However, it still lagged behind routing and MTL. One key limitation of model merging is its tendency to disproportionately favor tasks with more training data, allowing their influence to dominate the merged weights. For example, in our experiments, captioning performance was retained significantly better than that on SNLI-VE (SNLI) or Hateful Memes (HM), which had fewer examples.

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	**1.350**	83.3	**76.8**	**67.4**	**86.3**	**73.1**	89.4

Table 3: Comparison of routing vs. LoRA-specialized models. Routing achieves competitive or superior performance on all tasks using a single model. InternVL2-8B is used.

Routing Matches or Outperforms Specialized Models Table 3 compares routing against both zero-shot and task-specific LoRA fine-tuned models across specialized for InternVL-2 8B model. While LoRA models offer strong performance on the task they were fine-tuned for (e.g., LoRA-SNLI excels on SNLI), they often exhibit reduced performance on unrelated domains, revealing limited generalization.

In contrast, the routing model matches or surpasses each specialized LoRA variant on its corresponding task—achieving better scores on MMIMDb and Hateful Memes—and performs competitively on others. Notably, routing maintains strong results across benchmark tasks such as MMBench and ChartQA, rivaling or slightly exceeding even the zero-shot and LoRA baselines. This demonstrates its capacity to adaptively activate relevant modules while preserving the base model's general-purpose strengths.

By learning token-level gates that modulate each LoRA

module's contribution, routing avoids the interference typically caused by multi-task fine-tuning and mitigates catastrophic forgetting. Its selective activation mechanism enables it to leverage task-specific expertise without compromising robustness across unrelated domains—delivering the benefits of specialization and generalization in a single model.

Ablation Study

In this section we perform different ablations on routing based approach to demonstrate it's robustness and computational efficiency compared to other approaches.

Adding more LoRA modules doesn't degrade performance

In this section, we evaluate the scalability and robustness of the routing mechanism as more LoRA modules are incrementally added to the model. Our goal is to test whether introducing additional experts causes performance degradation or interference among tasks.

We begin by routing between two task-specific experts, SNLI and COCO. As shown in Table 4, this configuration yields strong performance on both SNLI and COCO while having nearly zero-shot performance on unrelated tasks like Hateful Memes and MMIMDb.

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

Table 4: Performance of routing models with incremental inclusion of modules across datasets using InternVL2-2B. Zero-shot baseline included for reference.

Adding a LoRA module for the Hateful Memes task leads to a substantial gain in that task's performance, demonstrating the model's capacity to correctly route samples to the appropriate expert. Extending this setup by including the MMIMDb LoRA module results in significant improvement on MMIMDb, while maintaining previously attained performance levels across other tasks.

To test the system's robustness to unrelated modules, we add a LoRA module trained on ChartGEMMA (Masry et al. 2024), a dataset unrelated to the core tasks. As seen in Table 4, the inclusion of this unrelated module does not degrade performance, reinforcing the routing system's resilience and scalability, even in the presence of unrelated modules.

Finally, we introduce a LoRA module for XNLI (Conneau et al. 2018), a text-to-text entailment task closely related to SNLI, which specializes in text-to-image entailment task. Interestingly, adding the XNLI module not only boosts performance on image-text tasks but also shows the model's ability to leverage cross-modal transfer, where improvements on text-based tasks benefit performance on related image-based tasks.

Task	Zero-shot	**Expert Models**		Routing
		orca-math (text)	multi-math (image+text)	
English	0.57	0.78	0.62	**0.82**
Chinese	0.57	0.62	0.58	**0.64**

Table 5: Cross-modal transfer experiments using routing with modality-specific experts on the MGSM dataset.

Routing Enables Cross-modal Transfer

Building on our findings from the XNLI and SNLI experiments, we further examine routing's effectiveness in enabling cross-modal transfer within VLMs. Specifically, we evaluate this on the multi-lingual math dataset MGSM (Shi et al. 2022), focusing on zero-shot performance.

As shown in Table 5, we employ two LoRA modules: Orca-Math, trained on text-based mathematical reasoning, and Multi-Math, trained on both image and text modalities. Both modules improve performance on MGSM, with Orca-Math outperforming Multi-Math, likely due to its alignment with the task structure of MGSM.

Crucially, routing achieves the strongest performance across both languages by dynamically integrating knowledge from multiple experts, surpassing individual LoRA modules. This highlights its effectiveness in cross-modal knowledge transfer, wherein model is able to use information from both text based orca-math expert as well as image and text based mutli-math expert allowing the model to leverage complementary information from diverse modalities. These results demonstrate that routing serves as a robust mechanism for improving multi-modal performance by effectively leveraging knowledge from unimodal tasks.

Impact of Model Size

In this section, we examine the effect of model size on the routing approach by comparing the performance of specialized models with that of a routing model. The routing model integrates SNLI-VE, COCO, Hateful Memes, and MMIMDb as LoRA modules within the routing pool. A smaller performance drop in the routing model, relative to specialized models, indicates better robustness.

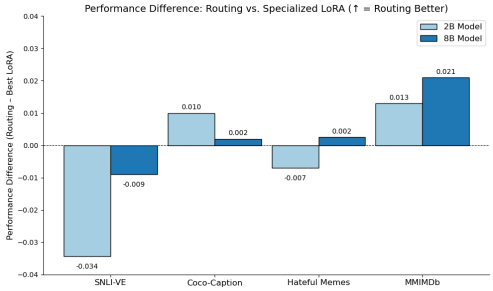

Figure 2: Performance drop in comparison to specialized model for 2B and 8B models.

In Figure 2, the 8B model exhibits superior robustness,

with minimal performance drops across all tasks compared to the 2B model. Notably, the 8B routing model even outperforms the specialized models on tasks such as COCO, Hateful Memes, and MMIMDb. In contrast, the 2B model experiences a more significant performance drop across all tasks, with SNLI-VE showing the largest gap.

These results indicate that larger models, such as the 8B variant, can effectively manage multiple tasks with minimal trade-offs, likely due to their increased capacity and improved generalization capabilities. This behavior is consistent with findings from prior research on model merging, as noted by Yadav et al. (2024).

Computational Efficiency Analysis

We analyze the computational advantages of our routing-based LoRA approach compared to standard continual learning methods. Table 6 reports training time and data requirements for InternVL2-2B across four specialized tasks.

Method	Training Time (hrs)		Data Requirements	
	First Task	All Tasks	First Task	All Tasks
Seq. FT	8	11.0	1×	1×
Seq. FT + ER	8	12.8	1×	2.2×
MTL	8	15.6	1×	4×
Routing	4	8.4	1×	1×

Table 6: Comparison of computational efficiency. Training time in GPU hours on 32×A10 GPUs; data usage is shown relative to the first task.

As shown in Table 6, our routing-based approach is substantially more efficient in both training time and data usage. This efficiency arises from two key design choices: (i) the use of lightweight LoRA adapters, and (ii) training exclusively on the current task's data, without revisiting prior datasets or retraining earlier components. While multi-task learning (MTL) can match routing in specialized task performance, it incurs significantly higher compute and data costs—approximately 2× training time and 4× data usage—as it requires joint access to all task data and, as shown in Table 1, suffers from degraded performance on benchmark tasks due to CF on general purpose knowledge. Experience replay (ER), although more efficient than MTL, still faces growing complexity as the number of tasks increases. We demonstrate that routing framework strikes a practical balance by maintaining strong performance, limiting computational overhead, and scaling easily to an increasing number of tasks—making it well-suited for real-world continual learning scenarios.

Discussion

Relationship to Mixture of Experts

While our routing approach shares architectural similarities with Mixture of Experts (MoE) systems like MoE-LLaVA (Chen et al. 2024a), it fundamentally differs in both purpose and implementation. It's important to clarify these distinctions as they represent different paradigms in modular machine learning.

Traditional MoE approaches are designed to improve model capacity and performance through joint training of specialized experts within a unified architecture. During both training and inference, a gating network distributes computation across experts based on input characteristics. This requires simultaneous access to all training data and joint optimization of all experts.

In contrast, our routing-based continual learning approach:

1. **Sequential vs. Joint Training:** Our experts (LoRA modules) are trained independently and sequentially, with each expert focusing on a specific task without access to other tasks' data. Traditional MoEs require joint training of all experts from the beginning.
2. **Architectural Separation:** Our approach explicitly decouples expert learning from routing. We first train task-specific LoRA modules independently, and then learn lightweight routing vectors while keeping the LoRA parameters frozen. This modular design supports continual adaptation, unlike traditional MoE systems where expert structure and routing are jointly optimized from the get-go on all tasks.
3. **Data Efficiency:** Our approach requires only new task data, eliminating the need to retain or synthesize previous task data—a key requirement for practical continual learning in data-constrained environments.

Overall, unlike traditional MoEs that route among fully joint-trained large experts, our approach routes among lightweight, task-specialized parameter-efficient modules optimized for scalable continual adaptation in large vision-language models. This makes our method particularly well-suited for settings where full retraining of large experts is computationally prohibitive or data reuse is restricted.

Conclusion

In this work, we address catastrophic forgetting in Vision-Language Models (VLMs) by introducing a routing-based continual learning approach that enables seamless integration of new task-specific experts while preserving foundational capabilities. Unlike multi-task learning and experience replay, which incur $\mathcal{O}(n)$ growth in data and computation, our method maintains constant $\mathcal{O}(1)$ resource requirements per task and eliminates the need to revisit prior data, making it highly efficient. Through extensive experiments with InternVL2 models (2B and 8B), we show that routing achieves performance comparable to multi-task learning—the theoretical upper bound—on specialized tasks, while preserving general capabilities on benchmarks such as ChartQA, DocVQA, and MMBench. Routing also enables cross-modal transfer, allowing improvements in one modality (e.g., vision) to enhance performance in another (e.g., language), a property not exhibited by existing continual learning methods. Furthermore, we find that larger models exhibit increased resistance to forgetting. In future work, we aim to extend this framework to support cross-lingual transfer in VLMs.

References

Arevalo, J.; Solorio, T.; y Gómez, M. M.; and González, F. A. 2017. Gated Multimodal Units for Information Fusion. arXiv:1702.01992.

Bowman, S. R.; Angeli, G.; Potts, C.; and Manning, C. D. 2015. A large annotated corpus for learning natural language inference. In Màrquez, L.; Callison-Burch, C.; and Su, J., eds., *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 632–642. Lisbon, Portugal: Association for Computational Linguistics.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neel, S.; Shinn, E.; Steinhardt, J.; Christian, G.; et al. 2020. Language Models are Few-Shot Learners. *arXiv preprint arXiv:2005.14165*.

Caruana, R. 1997. Multitask Learning. *Machine Learning*, 28(1): 41–75.

Cha, S.; Lee, H.; Shin, J.; and Shin, J. 2020. CPR: Classifier-projection regularization for continual learning. *arXiv preprint arXiv:2006.07326*.

Chaudhry, A.; et al. 2018. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*.

Chen, B.; Wang, H.; Du, T.; Yu, S.; An, R.; Gao, Q.; Lin, D.; and Wang, J. 2024a. MoE-LLaVA: Mixture of Experts for Large Vision-Language Models.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, 1597–1607. PMLR.

Chen, Z.; Wu, J.; Wang, W.; Su, W.; Chen, G.; Xing, S.; Zhong, M.; Zhang, Q.; Zhu, X.; Lu, L.; et al. 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*, 24185–24198.

Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H. W.; Sutton, C.; Gehrmann, S.; Schuh, P.; Shi, K.; Tsvyashchenko, S.; Maynez, J.; Rao, A.; Barnes, P.; Tay, Y.; Shazeer, N.; Prabhakaran, V.; Reif, E.; Du, N.; Hutchinson, B.; Pope, R.; Bradbury, J.; Austin, J.; Isard, M.; Gur-Ari, G.; Yin, P.; Duke, T.; Levskaya, A.; Ghemawat, S.; Dev, S.; Michalewski, H.; Garcia, X.; Misra, V.; Robinson, K.; Fedus, L.; Zhou, D.; Ippolito, D.; Luan, D.; Lim, H.; Zoph, B.; Spiridonov, A.; Sepassi, R.; Dohan, D.; Agrawal, S.; Omernick, M.; Dai, A. M.; Pillai, T. S.; Pellat, M.; Lewkowycz, A.; Moreira, E.; Child, R.; Polozov, O.; Lee, K.; Zhou, Z.; Wang, X.; Saeta, B.; Diaz, M.; Firat, O.; Catasta, M.; Wei, J.; Meier-Hellstern, K.; Eck, D.; Dean, J.; Petrov, S.; and Fiedel, N. 2023. PaLM: scaling language modeling with pathways. *J. Mach. Learn. Res.*, 24(1).

Conneau, A.; Rinott, R.; Lample, G.; Williams, A.; Bowman, S. R.; Schwenk, H.; and Stoyanov, V. 2018. XNLI: Evaluating Cross-lingual Sentence Representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Douillard, A.; et al. 2021. End-to-End Task-Specific Model Merging for Multi-Task Learning. *Proceedings of the International Conference on Machine Learning*, 139: 1380–1389.

Fedus, W.; Zoph, B.; and Shazeer, N. 2021. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23: 1–39.

Goodfellow, I. J.; Mirza, M.; Xiao, D.; Courville, A.; and Bengio, Y. 2013. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*.

Hinton, G.; et al. 2015. Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W.; et al. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2): 3.

Ilharco, G.; Ribeiro, M. T.; Wortsman, M.; Schmidt, L.; Hajishirzi, H.; and Farhadi, A. 2023. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*.

Kairouz, P.; McMahan, H. B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A. N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. 2021. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14(1–2): 1–210.

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

Kemker, R.; McClure, M.; Abitino, A.; Hayes, T.; and Kanan, C. 2018. Measuring catastrophic forgetting in neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Kiela, D.; Firooz, H.; Mohan, A.; Goswami, V.; Singh, A.; Ringshia, P.; and Testuggine, D. 2021. The Hateful Memes Challenge: Detecting Hate Speech in Multimodal Memes. arXiv:2005.04790.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, C.; Quan, J.; Ramalho, T.; Grabska-Barwińska, A.; et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13): 3521–3526.

Li, T.; Sahu, A.; Talwalkar, A.; and Smith, V. 2020. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems (MLSys)*.

Li, Y.; et al. 2018. Learning to route with neural modular networks. *arXiv preprint arXiv:1809.10778*.

Li, Z.; and Hoiem, D. 2017. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12): 2935–2947.

Lin, B.; Tang, Z.; Ye, Y.; Huang, J.; Zhang, J.; Pang, Y.; Jin, P.; Ning, M.; Luo, J.; and Yuan, L. 2024. MoE-LLaVA: Mixture of Experts for Large Vision-Language Models. arXiv:2401.15947.

Lin, T.-Y.; Maire, M.; Belongie, S.; Bourdev, L.; Girshick, R.; Hays, J.; Perona, P.; Ramanan, D.; Zitnick, C. L.; and

Dollár, P. 2015. Microsoft COCO: Common Objects in Context. arXiv:1405.0312.

Liu, H.; Li, C.; Wu, Q.; and Lee, Y. J. 2023. Visual Instruction Tuning. *arXiv preprint arXiv:2304.08485*.

Liu, Y.; Duan, H.; Zhang, Y.; Li, B.; Zhang, S.; Zhao, W.; Yuan, Y.; Wang, J.; He, C.; Liu, Z.; Chen, K.; and Lin, D. 2024. MMBench: Is Your Multi-modal Model an All-around Player? arXiv:2307.06281.

Liu, Y.; Duan, H.; Zhang, Y.; Li, B.; Zhang, S.; Zhao, W.; Yuan, Y.; Wang, J.; He, C.; Liu, Z.; et al. 2025. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, 216–233. Springer.

Lopez-Paz, D.; and Ranzato, M. 2017. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, volume 30.

Lu, J.; Batra, D.; Parikh, D.; and Lee, S. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in neural information processing systems*, volume 32.

Luo, Y.; et al. 2023. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *arXiv preprint arXiv:2308.08747*.

Mallya, A.; Davis, D.; and Lazebnik, S. 2018. Piggyback: Adding new tasks to a single network with weight transformations using binary masks. In *Proceedings of the European Conference on Computer Vision*, 72–87.

Mallya, A.; and Lazebnik, S. 2018. PackNet: Adding Multiple Tasks to a Single Network by Iterative Pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7765–7773.

Mallya, A.; and Lazebnik, S. 2022. Forget-free Continual Learning with Winning Subnetworks. In *International Conference on Machine Learning*, 15014–15024.

Masry, A.; Long, D.; Tan, J. Q.; Joty, S.; and Hoque, E. 2022a. ChartQA: A Benchmark for Question Answering about Charts with Visual and Logical Reasoning. In *Findings of the Association for Computational Linguistics: ACL 2022*, 2263–2279. Dublin, Ireland: Association for Computational Linguistics.

Masry, A.; Long, D. X.; Tan, J. Q.; Joty, S.; and Hoque, E. 2022b. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. *arXiv preprint arXiv:2203.10244*.

Masry, A.; Thakkar, M.; Bajaj, A.; Kartha, A.; Hoque, E.; and Joty, S. 2024. ChartGemma: Visual Instruction-tuning for Chart Reasoning in the Wild. arXiv:2407.04172.

Matena, M.; and Raffel, C. 2022. Merging Models with Fisher-Weighted Averaging. arXiv:2111.09832.

Mathew, M.; Karatzas, D.; and Jawahar, C. 2021a. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2200–2209.

Mathew, M.; Karatzas, D.; and Jawahar, C. V. 2021b. DocVQA: A Dataset for VQA on Document Images. arXiv:2007.00398.

McMahan, H. B.; Moore, E.; Ramage, D.; Hampson, S.; and Agüera y Arcas, B. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 1273–1282. PMLR.

Merlingot, S.; Gagnon-Audet, J.-C.; Kadoury, S.; and Pal, C. 2024. MagMax: Tackling continual learning with automated model merging. *arXiv preprint arXiv:2403.07505*.

Mitra, A.; Khanpour, H.; Rosset, C.; and Awadallah, A. 2024. Orca-Math: Unlocking the potential of SLMs in Grade School Math. arXiv:2402.14830.

Muennighoff, N.; Wang, T.; Sutawika, L.; Roberts, A.; Biderman, S.; Scao, T. L.; Bari, M. S.; Shen, S.; Yong, Z.-X.; Schoelkopf, H.; et al. 2022. Crosslingual generalization through multitask finetuning. *arXiv preprint arXiv:2211.01786*.

Muqeeth, M.; Liu, H.; Liu, Y.; and Raffel, C. 2024. Learning to route among specialized experts for zero-shot generalization. *arXiv preprint arXiv:2402.05859*.

Ostapenko, O.; Su, Z.; Ponti, E. M.; Charlin, L.; Roux, N. L.; Pereira, M.; Caccia, L.; and Sordoni, A. 2024. Towards Modular LLMs by Building and Reusing a Library of LoRAs. arXiv:2405.11157.

Peng, S.; Fu, D.; Gao, L.; Zhong, X.; Fu, H.; and Tang, Z. 2024. MultiMath: Bridging Visual and Mathematical Reasoning for Large Language Models. arXiv:2409.00147.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, 8748–8763. PMLR.

Rebuffi, S.-A.; Kolesnikov, A.; Sperl, G.; and Lampert, C. H. 2017. iCaRL: Incremental classifier and representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2001–2010.

Riquelme, C.; Puigcerver, J.; Mustafa, B.; Neumann, M.; Jenatton, R.; Pinto, A. S.; Keysers, D.; and Houlsby, N. 2021. Scaling vision with sparse mixture of experts. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21. Red Hook, NY, USA: Curran Associates Inc. ISBN 9781713845393.

Rolnick, D.; et al. 2019. Experience replay for continual learning. In *Advances in Neural Information Processing Systems*, volume 32.

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

Rusu, A. A.; et al. 2016. Progressive neural networks. *arXiv preprint arXiv:1606.04671*.

Shazeer, N.; Mirhoseini, A.; Maziarz, K.; Davis, A.; Le, Q.; Hinton, G.; and Dean, J. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

Shi, F.; Suzgun, M.; Freitag, M.; Wang, X.; Srivats, S.; Vosoughi, S.; Chung, H. W.; Tay, Y.; Ruder, S.; Zhou, D.; Das, D.; and Wei, J. 2022. Language Models are Multilingual Chain-of-Thought Reasoners. arXiv:2210.03057.

Shin, H.; Lee, J. K.; Kim, J.; and Kim, J. 2017. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30.

Shoham, C.; Rotem, O.; and Ben-Ari, R. 2022. Federated continual learning via experience replay. *Proceedings of the European Conference on Artificial Intelligence (ECAI)*.

T Dinh, C.; Tran, N.; and Nguyen, T. 2020. Personalized federated learning with adaptive clustering. In *Proceedings of the 39th IEEE International Conference on Distributed Computing Systems (ICDCS)*.

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

Wang, H.; Lu, H.; Yao, L.; and Gong, D. 2024a. Self-Expansion of Pre-trained Models with Mixture of Adapters for Continual Learning. *arXiv preprint arXiv:2403.18886*.

Wang, Y.; Liu, Y.; Shi, C.; Li, H.; Chen, C.; Lu, H.; and Yang, Y. 2024b. InsCL: A Data-efficient Continual Learning Paradigm for Fine-tuning Large Language Models with Instructions. arXiv:2403.11435.

Wang, Y.; et al. 2024c. Inscl: A data-efficient continual learning paradigm for fine-tuning large language models with instructions. *arXiv preprint arXiv:2403.11435*.

Wang, Z.; Lipton, Z. C.; and Tsvetkov, Y. 2020. On Negative Interference in Multilingual Models: Findings and A Meta-Learning Treatment. In Webber, B.; Cohn, T.; He, Y.; and Liu, Y., eds., *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 4438–4450. Online: Association for Computational Linguistics.

Wortsman, M.; Ramanujan, V.; Liu, R.; Kembhavi, A.; Rastegari, M.; Yosinski, J.; and Farhadi, A. 2020. Supermasks in Superposition. In *Advances in Neural Information Processing Systems*, volume 33, 15173–15184.

Xie, N.; Lai, F.; Doran, D.; and Kadav, A. 2018. Visual Entailment Task for Visually-Grounded Language Learning. *arXiv preprint arXiv:1811.10582*.

Yadav, P.; Vu, T.; Lai, J.; Chronopoulou, A.; Faruqui, M.; Bansal, M.; and Munkhdalai, T. 2024. What Matters for Model Merging at Scale? arXiv:2410.03617.

Yoon, J.; Yang, E.; and Hwang, S. J. 2021. Federated continual learning with a mixture of experts. *Advances in Neural Information Processing Systems (NeurIPS)*.

Zenke, F.; Poole, B.; and Ganguli, S. 2017. Continual Learning Through Synaptic Intelligence. In *International Conference on Machine Learning*, 3987–3995. PMLR.

Zhai, Y.; et al. 2023. Investigating the catastrophic forgetting in multimodal large language models. *arXiv preprint arXiv:2309.10313*.

Zhao, H.; Wang, X.; Sahu, A.; and Talwalkar, A. 2022. Federated continual learning with knowledge distillation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

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, Lawrence Zitnick, and Parikh 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.

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. 2025), ChartQA (Masry et al. 2022b), and DocVQA (Mathew, Karatzas, and Jawahar 2021a) 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.

Related work

Continual Learning

Experience replay methods address CF by maintaining a small buffer of previously observed data, which is reused to retrain the model when learning new tasks. Although these methods theoretically pose a risk of looser generalization bounds, they are widely valued for their simplicity, stability, and robust performance, even with limited episodic memory buffers (Chaudhry et al. 2018; Rolnick et al. 2019). This practicality has made replay-based approaches a popular choice in continual learning scenarios (Rebuffi et al. 2017; Lopez-Paz and Ranzato 2017). The Instruction-based Continual Learning (InsCL) method (Wang et al. 2024c) extended this idea by introducing dynamic replay, which selects prior data based on task similarity, measured using the Wasserstein Distance with task-specific instructions. However, our experiments demonstrate that this approach results in forgetting earlier tasks, highlighting its limitations in retaining comprehensive task knowledge.

Regularization methods tackle CF by adding a penalty term to the loss function, which discourages significant changes to model parameters that are critical for previously learned tasks. The key principle is to balance retaining prior knowledge

with acquiring new information, typically controlled by a regularization coefficient. Notable examples include Elastic Weight Consolidation (EWC) (Kirkpatrick et al. 2017) and Learning without Forgetting (LwF) (Li and Hoiem 2017). EWC constrains significant changes to parameters deemed crucial for earlier tasks, while LwF employs a knowledge distillation loss (Hinton et al. 2015) to preserve performance on previous tasks. Despite their effectiveness, these methods often increase memory consumption and can be challenging to train due to the difficulty in achieving an optimal balance between old and new tasks. A routing-based approach offers a promising alternative, mitigating these issues by providing greater flexibility in balancing tasks while optimizing memory efficiency.

Federated Learning

Federated Learning (FL) has become a pivotal paradigm in distributed machine learning, enabling multiple decentralized modules to collaboratively train a shared model while keeping raw data local. This framework addresses critical privacy and data sovereignty concerns by aggregating model parameters rather than exchanging sensitive data (McMahan et al. 2017). However, FL encounters several challenges, including handling non-IID (non-independent and identically distributed) data across modules, mitigating communication inefficiencies, and ensuring equitable performance across diverse modules (Kairouz et al. 2021; Li et al. 2020). Techniques like FedAvg (McMahan et al. 2017) and its adaptations have sought to address these issues by optimizing communication protocols and improving robustness to heterogeneous data distributions.

CL within the FL paradigm introduces further complexities due to the dynamic nature of non-stationary data distributions (Yoon, Yang, and Hwang 2021). CF becomes particularly pronounced as models need to adapt to new tasks while retaining prior knowledge without direct access to historical data. Federated continual learning (FCL) approaches tackle these challenges by leveraging methods such as knowledge distillation (Zhao et al. 2022), experience replay (Shoham, Rotem, and Ben-Ari 2022), and personalized model training (T Dinh, Tran, and Nguyen 2020). These strategies strive to balance global knowledge retention with local task adaptation, yet scalability and robustness remain open challenges. Modular machine learning offers a scalable solution to this interplay by enabling task-specific pathways, where only essential modular updates are shared rather than entire model parameters. This reduces communication overhead while preserving privacy. However, the primary challenge with routing lies in training these task-specific pathways effectively, which we identify as an important direction for future research.

Routing patterns

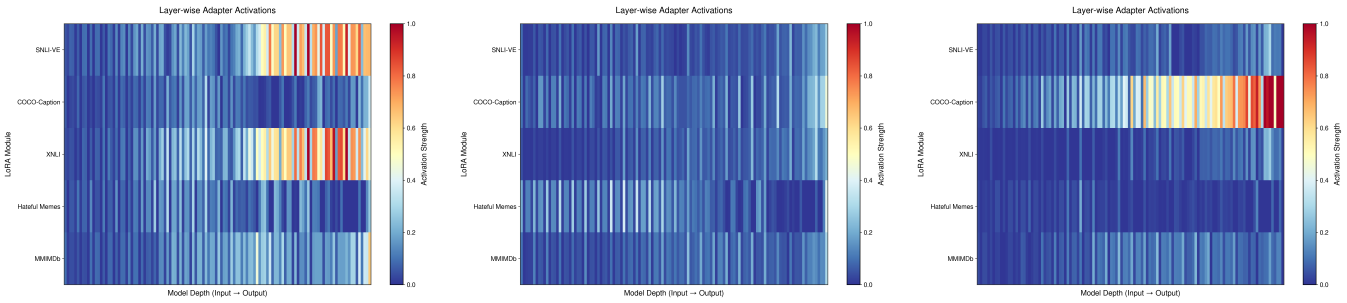

Figure 3: Routing patterns for SNLI (left), MMBENCH (middle), and COCO (right). The figure demonstrates that the routing model correctly activates task-specific modules: SNLI and XNLI for SNLI, only COCO for COCO, and no module for MMBENCH, where the base model handles the task. These confirm the routing mechanism's effectiveness in selecting the appropriate module based on the task.

In Figure 3 (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 discussed in section on routing vs specialized model, 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 3 (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 3 (middle), we present the routing pattern for the benchmark task MMBENCH. As this is a general-purpose task, we observe that no LoRA module is activated, and the model relies on the base model's knowledge to solve the task. This demonstrates the model's ability to bypass LoRA modules when the base model has sufficient expertise to address a specific task, showcasing the efficiency and adaptability of the routing mechanism.

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. 2022).

Table 7 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.

While the 8B model demonstrates strong zero-shot performance initially, the Chinese expert negatively affects the results, indicating potential interference with the model's pre-trained multilingual capabilities. The Math expert improves performance in isolation, but the routing mechanism fails to effectively combine both experts, causing a significant decline in performance. This suggests that the larger-scale language expert is underperforming, which in turn hampers the routing process. A more optimized data mixture for the language model may be necessary to address this issue.

Figure 4 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 7: 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

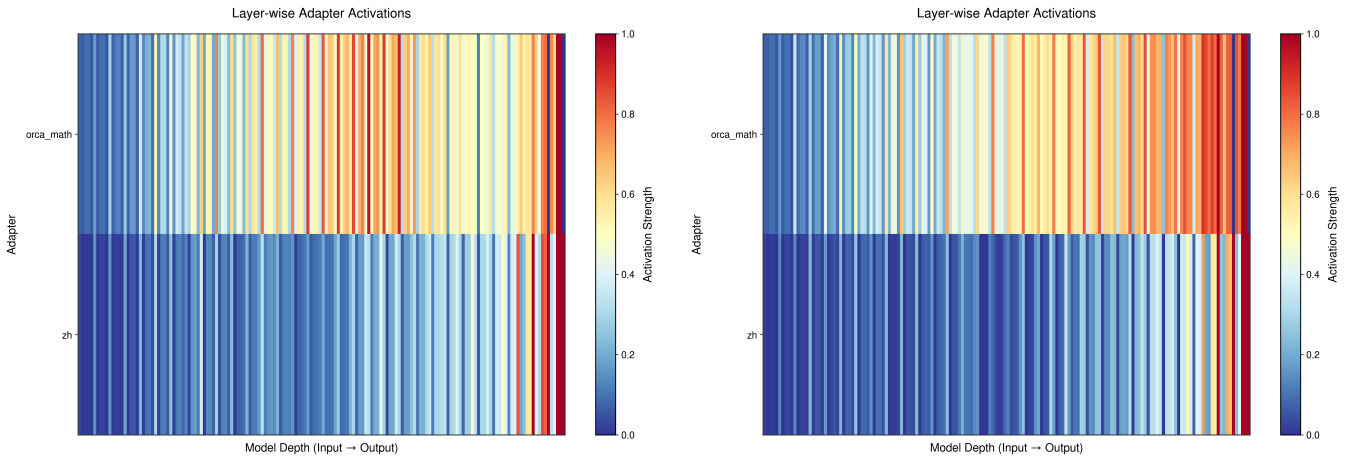

Figure 4: Routing patterns for the MGSM dataset in multilingual transfer. Notably, the model leverages the Chinese expert in the early and later stages, while the math expert is primarily used for computation.

Training details

For the full finetuning, we use a learning rate of 1×10^{-5} and train the models for a single epoch, unless stated otherwise. We employ total of 32 A10 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.

The models are trained using a cosine learning rate scheduler to adjust the learning rate over the course of training. We leverage the ZeRO-3 configuration to efficiently offload gradients, optimizer states, and parameters. The models are trained using a cosine learning rate scheduler to adjust the learning rate over the course of training. For specializing LoRA module, we employ a learning rate of $4 \times 10^{-}5$. LoRA modules are added exclusively to the LLM, ensuring that only the LoRA weights of the LLM are updated during training. To train the gating module we freeze all the parameters and only train the gate vector v for 100 steps on the specialized task.

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