How To Teach Large Multimodal Models New Skills?

Zhen Zhu[1,2*], Yiming Gong[1,3*], Yao Xiao[1], Yaoyao Liu[1], and Derek Hoiem[1]

[1] University of Illinois Urbana-Champaign, IL, USA
[2] Google DeepMind, CA, USA
[3] Carnegie Mellon University, PA, USA
{zhenzhu4,yimingg8,yaox11,lyy,dhoiem}@illinois.edu
[*] Work done at the University of Illinois Urbana-Champaign.

Abstract. How can we teach large multimodal models (LMMs) new skills without erasing prior abilities? We study sequential fine-tuning on five target skills while monitoring general ability on eight held-out benchmarks across three model families. Surprisingly, we find that performance lost on held-out tasks after fine-tuning on one skill can partly recover when the model is subsequently tuned on a different skill. We trace this behavior to a measurable shift in the output token distribution, manifested through a simple counting-bias probe that shows the shift co-varies with forgetting. Guided by this insight, we identify two simple, robust tuning recipes that learn strongly while limiting drift: (i) updating only the self-attention projection layers (SA Proj., Δ learning +24.9 / Δ held-out forgetting -0.6), and (ii) updating only the MLP Gate&Up while freezing the Down projection (+30.5 / -2.1). Both substantially outperform full-LLM tuning (+31.8 / -23.3) in the learning–forgetting trade-off. We also compare against common forgetting mitigation methods—Learning without Forgetting (LwF), LoRA, Mixture-of-Experts, and weight-space interpolation (WiSE-FT)—and find that our selective tuning recipes match or exceed their learning–stability balance while remaining simpler, requiring no replay, auxiliary parameters, or per-stage tuning. These results hold across LLaVA-OneVision, LLaVA-NeXT, and Qwen2.5-VL, confirming that the key to teaching LMMs new skills without forgetting lies in controlling output distribution shift by choosing which components to tune. Code will be made available.

Keywords: Large Multimodal Models · Continual Learning · Selective Tuning · Learning without Forgetting

1 Introduction

Large multimodal models (LMMs), such as LLaVA [33] and Qwen2.5-VL [2], are trained to generate natural language answers based on image(s) and natural language instruction. As such, these models can perform a wide range of tasks.

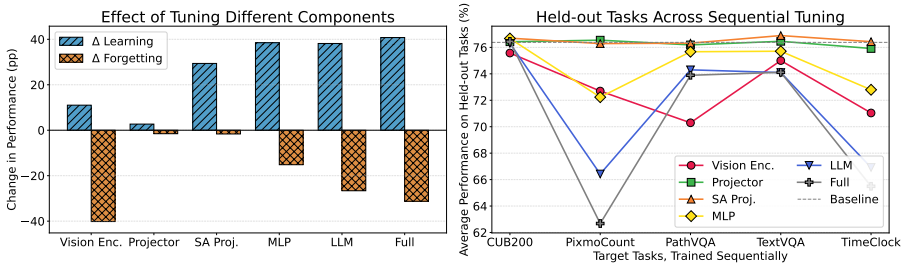

Fig. 1: Surprising Forgetting Behavior in LMMs: **Left**: When fine-tuning most components on one target task, we see major improvement in that task ("Learning") but a substantial drop in performance of other tasks ("Forgetting", total across tasks shown here), as expected. But if we only tune self-attention projection layers (SA Proj.) in the language model, we still get substantial learning on the target task with minimal forgetting. **Right**: Even fine-tuning SA Proj. for multiple tasks sequentially, we see no forgetting. For others, we see large forgetting on the PixmoCount task, but the models somehow partly recover what they "forgot" in learning the next specialized task. Our paper documents and analyzes these and other interesting phenomena of learning and forgetting in LMMs, leading to simple and effective ways to teach LMMs new skills.

However, for many special domains, such as medical images, or skills, such as counting, the models do not perform as desired. How can we teach LMMs something new without degrading existing capabilities?

Training a new LMM can cost millions of dollars, weeks of time, and emit hundreds of tons of CO_2; accordingly, practitioners often adapt existing models via fine-tuning rather than retraining from scratch. Since some LMMs are trained with only a single pass over their data [26], it is natural to keep adapting them as new data and use cases emerge. However, naïve task-by-task fine-tuning can degrade broad vision–language ability ("catastrophic forgetting"), as reported in prior work [6,70,75]. However, we show that forgetting is not uniform: it depends strongly on *which parameters are updated*, and some performance drops after a narrow adaptation stage can partially reverse later.

To study the forgetting dynamics of LMMs systematically, we develop a controlled evaluation suite. We sequentially fine-tune on five practical target skills spanning different answer formats (fine-grained bird classification, counting, medical VQA, OCR reading, and time reading), while monitoring general vision–language ability on eight widely used held-out benchmarks. We report *learning* as improvement on the current target task and *forgetting* as the average drop on the held-out suite. This protocol lets us compare tuning strategies on equal footing and directly measure the trade-off between acquiring new skills and preserving broad capabilities.

Our first goal is to identify tunable parts that deliver high target performance with minimal forgetting. We compare full-model fine-tuning to tuning each major component (vision encoder, projector, LLM) and then open the LLM into its two essential blocks—self-attention projections (SA Proj.) and the feed-forward network (MLP). Early experiments on LLaVA-OneVision (Fig. 1) reveal two surprising results: 1) tuning SA Proj. learns with little or no measurable forgetting

across a five-task sequence; and 2) what appears forgotten after one stage can be recovered by tuning another specialized task.

These results lead us to ponder: why is SA Proj. tuning so robust to forgetting, and how is forgotten knowledge recovered without rehearsing? Consider the roles of the two essential components in the transformer decoder: self-attention projections primarily route and combine information from the context [13, 51], while MLPs act like key–value memories that project activated features back into the residual stream and thereby influence the next-token distribution [14]. We thus hypothesize that what appears to be forgetting after fine-tuning on a narrow target task is often a bias in the output distribution induced by the task's answer format. We test this on the *counting* task: tuning the MLP improves the target performance but also increases the probability of emitting numeric tokens on a general *captioning* task, and held-out task accuracy drops accordingly. In contrast, tuning SA Proj. learns the counting skill without much bias to numeric tokens and without losing held-out performance (Sec. 5.2).

Guided by this result, we explore tuning recipes that preserve learning while limiting output shift. To avoid biasing the output distribution, we tune the MLP up/gating projections while keeping the down projection frozen, and find that it achieves similar learning to full MLP tuning with little forgetting. We experiment on LLaVA-OneVision [26] by training five target tasks sequentially, averaging over three sequence orders, measuring the learning and forgetting in target tasks and held-out tasks (Sec. 5.1). We then confirm that similar trends hold for LLaVA-NeXT [25] and Qwen2.5-VL [2] (Sec. 5.3).

To our knowledge, we are the **first** to systematically study rehearsal-free continual learning in large multimodal models under sequential skill adaptation while monitoring performance on broad vision–language benchmarks, and to show that apparent forgetting can be partially recoverable across stages. **Our main contribution** is a mechanistically motivated view of *where* to update an LMM to acquire new skills while *limiting output-distribution drift*, together with a thorough evaluation suite spanning **five** practical target skills, **eight** held-out benchmarks, and **three** model families. Our key findings are:

- **Tuning the LLM (Δ learning +31.8/Δ forgetting -23.3) is critical for learning new tasks**, while tuning the vision encoder (+9.6/-10.8) brings little gain and harms general ability.
- **Tuning only the self-attention projection weights (+24.9/-0.6) or the up&gate layers of the MLP (+30.5/-2.1) provides excellent learning with limited forgetting**, evaluated on a five-target task sequence, eight held-out benchmarks, and three model families.
- **Forgetting is largely a manifestation of output distribution shift.** We use a counting-bias probe to show that the rise in number-token likelihood grows with MLP tuning and remains near baseline for self-attention tuning; the magnitude of this shift covaries with performance drop in held-out tasks. This explains why some methods – tuning only SA Proj. or MLP Up&Gate, or full tuning with distillation loss on the output tokens – can learn with minimal forgetting.

2 Related work

Continual learning [1,5,8] aims to train models on a sequence of tasks or continuous data streams while preserving previously acquired knowledge. Much of the foundational work has been developed in closed-vocabulary image classification, where approaches can be broadly organized into three families: (1) *regularization* methods preserve knowledge from earlier model states by matching logits [30,55], feature maps [12], or other representational statistics [20, 37, 52, 58, 61, 65]; (2) *exemplar replay* methods maintain a reservoir of samples from previous training stages [3, 38, 40, 43, 53, 55, 57] and revisit them during later phases to reinforce past knowledge; and (3) *network expansion* methods [39, 65] increase model capacity for new target data while freezing a subset of parameters to retain prior knowledge. More recently, prompt tuning has emerged as an effective continual learning strategy [19,22,23,60], keeping all model weights frozen while introducing learnable prompts to accommodate new tasks.

Large language models (LLMs). A growing body of work evaluates the learning-forgetting trade-off across various fine-tuning strategies for LLMs with billions of parameters, including full-model tuning, adapters, LoRA, and prompt tuning. Luo et al. [42] find that decoder-only models exhibit greater robustness to forgetting than encoder-decoder models. Lin et al. [31] observe that models fine-tuned for narrow domains can lose general-task ability, though weight interpolation (WiSE-FT) [68] helps maintain a balance between specialization and retention. Biderman et al. [4] show that LoRA reduces both learning and forgetting relative to full fine-tuning. Li et al. [28] propose a dual-memory replay framework with interpolated LoRA to further manage this balance. Huang et al. [18] generate pseudo-data from the model itself to mitigate forgetting without requiring access to original training data. Xiang et al. [69] apply regularization strategies such as Elastic Weight Consolidation (EWC) [24] and hierarchical importance-based penalties to preserve general knowledge by constraining updates to important parameters. Wang et al. [66] learn orthogonal LoRA weights for new tasks, reducing interference with previously learned representations. Concurrent with our work, Shenfeld et al. [56] report that the degree of forgetting correlates with the distributional shift between the base and tuned models (measured by KL divergence), and use this finding to explain why on-policy RL training is more robust to forgetting than supervised fine-tuning.

Roles of attention and FFN in LLMs. Mechanistic studies of transformer blocks reveal a division of labor. Attention heads act primarily as *routing* and retrieval mechanisms: they select *where* to read from using query–key patterns and then mix the corresponding values; this view is formalized in the Transformer Circuits framework and supported by analyses of "induction heads," which implement a simple copying algorithm and closely track the emergence of in-context learning during training [51]. Complementing this routing role, feed-forward (FFN/MLP) blocks behave like *key–value memories*: learned keys detect input patterns while values write features that align with groups of vocabulary items, thereby shifting the model's output preferences [14]. Meng et al. [48] show that

directly modifying MLP weights updates specific facts while preserving unrelated behavior, providing further evidence that FFN layers serve as a principal site where factual associations are stored. Earlier analyses in BERT and NMT have also found that a minority of attention heads specialize in linguistically interpretable roles (e.g., syntax, coreference) while many heads are prunable with little loss, reinforcing the view of attention as selective routing rather than the main repository of lexical knowledge [9,63]. Our findings are consistent with this literature: self-attention updates tend to preserve global behavior, while MLP updates are the main driver of output-distribution shift.

Vision-text contrastive models such as CLIP [54], are trained to align images and texts for open-vocabulary image classification and retrieval. While CLIP achieves strong zero-shot performance, it may underperform on fine-grained or specialized tasks [54, 77], motivating the need for reliable continual learning approaches. Zhu et al. [76, 77] propose learning to blend predictions from original and tuned image encoders, enabling fast online learning without forgetting for open-vocabulary classification. Yu et al. [70] add parameter-efficient adapters to a mixture-of-experts on a frozen CLIP model to prevent forgetting. Zhou et al. [74] design task-specific projection layers and cross-modal fusion modules for vision-language models in class-incremental learning. Liu et al. [36] incorporate continual low-rank adaptation and knowledge consolidation to prevent forgetting. Zheng et al. [73] use knowledge distillation [30] on CLIP to maintain zero-shot performance. These methods focus on classification with frozen or lightly adapted encoders; our work instead addresses generative LMMs and investigates which decoder components to update.

Large multimodal models (LMMs). Although continual learning in generative instruction-following LMMs remains relatively underexplored, interest is growing rapidly. Chen et al. [6] find that LMMs suffer from catastrophic forgetting when learning a sequence of new tasks. Much of the existing work focuses on visual question answering (VQA) [11, 32, 44, 50, 71] or image captioning [49]. Zhang et al. [71] leverage both sample-specific and sample-invariant features to learn representations that are both discriminative and generalizable for VQA tasks. Nikandrou et al. [50] distill knowledge separately for each modality, ensuring that both image features and question features retain their relevant information when new tasks arrive. Lin et al. [32] combine selective memory replay and knowledge distillation for VQA. Marouf et al. [44] store only past questions from previous tasks as memory for rehearsal. More recent approaches rely on architectural expansions and dynamic routing, such as decoupling modalities via mixture-of-experts [67] or using dynamic MLLM-based routing mechanisms [72]. In earlier model generations, Nguyen et al. [49] integrate continual learning techniques, such as finetuning schemas and regularization, into the captioning pipeline to address forgetting, and Del Chiaro et al. [11] introduce an attention-based LSTM architecture.

In comparison, our study (i) evaluates rehearsal-free sequential skill adaptation on five diverse target skills while monitoring eight broad held-out bench-

marks across three model families; (ii) identifies two simple, widely applicable tuning recipes, SA Proj. and MLP Gate&Up (with Down frozen), that provide strong learning with limited forgetting; and (iii) provides a mechanistic analysis showing that much of the observed forgetting is tied to output-token distribution drift, which can be limited (Sec. 5.2) or regularized (Sec. 5.4) without adding new modules or storing rehearsal data.

3 Method

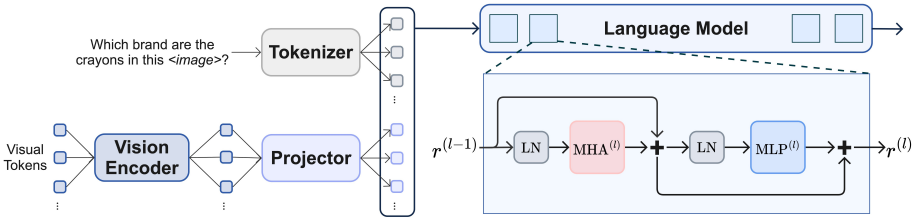

Fig. 2: Architecture of our evaluated LMMs. The input contains visual inputs such as images or videos, which are converted to visual tokens by the vision encoder, and text input is processed by a tokenizer containing a visual placeholder token `<image>`. Visual tokens are converted by the projector and concatenated with text tokens as input for the language model. We visualize the architecture of the transformer decoder layer of the language model. "LN", "MHA", and "MLP" represent layer norm, multi-head attention, and multi-layer perceptron, respectively. $r^{(l)}$ is the final output of layer l.

Setting. We adapt a pretrained large multimodal model on either a *single-target* task or a *sequential* stream of tasks. In the single-target case, given a target dataset $\mathcal{D}_{\mathrm{T}}$ and a held-out suite $\mathcal{D}_{\mathrm{H}}$, the goal is to improve performance on $\mathcal{D}_{\mathrm{T}}$ while preserving performance on $\mathcal{D}_{\mathrm{H}}$. In the sequential case, tasks $\{\mathcal{D}_{\mathrm{T}}^{(1)}, \ldots, \mathcal{D}_{\mathrm{T}}^{(K)}\}$ arrive in stages; unless noted, we update at each stage without rehearsal (no mixing of earlier tasks) and assess both the current target and the aggregated held-out suite after every stage.

3.1 Model

Overview of the LMM. Our evaluated LMMs have three major parts (Fig. 2): a vision encoder that turns an image into visual tokens, a projector that maps those tokens to the language width d, and a decoder-only transformer language model that produces next-token logits given the visual and text tokens.

Vision encoder and projector. The vision encoder produces $v = f_{\mathrm{vis}}(I) \in \mathbb{R}^{S_v \times d_v}$. The projector maps to the language representation width,

$$x_{\mathrm{vis}} = g_\psi(v) \in \mathbb{R}^{S_v \times d},$$

where ψ are the projector's trainable parameters.

Language model. The language model is a pre-norm, decoder-only transformer with L identical blocks. As illustrated in Fig. 2, the sublayer outputs and residual update at block l are

$$a^{(l)} = \mathrm{MHA}^{(l)}\big(\mathrm{LN}(r^{(l-1)})\big), \quad f^{(l)} = \mathrm{MLP}^{(l)}\big(\mathrm{LN}(r^{(l-1)} + a^{(l)})\big), \quad r^{(l)} = r^{(l-1)} + a^{(l)} + f^{(l)}, \tag{1}$$

where MHA and MLP denote the **multi-head self-attention** and **feed-forward network**, respectively, and LN is layer normalization.
Self-attention. With input x and per-head key width d_k,

$$Q = xW_Q,\ K = xW_K,\ V = xW_V,\ A = \mathrm{softmax}\Big(\tfrac{QK^\top}{\sqrt{d_k}}\Big),\ \mathrm{MHA}(x) = (AV)W_O \tag{2}$$

Here W_Q, W_K determine *where* to attend (routing), W_V selects what content is mixed, and W_O projects the mixed value features back into the residual stream at model width d.
Feed-forward. With input x and gating nonlinearity $\phi = \mathrm{SiLU}$,

$$\mathrm{MLP}(x) = W_{\mathrm{down}}(\phi(xW_{\mathrm{gate}}) \odot (xW_{\mathrm{up}}))\,, \tag{3}$$

where $\odot$ denotes the elementwise (Hadamard) product. Matrices $W_{\mathrm{gate}}, W_{\mathrm{up}}$ detect features (key-like pattern match), and W_{down} projects the activated features back to the residual stream at width d. We use $U \in \mathbb{R}^{d\times|V|}$ for the LM head (unembeddings) and denote the final block output as $r^{(L)}$, so the output logits are $z = U^\top r^{(L)}$.

Residual stream. Let $x_{\mathrm{text}} \in \mathbb{R}^{S_t\times d}$ be the text embeddings and $x_{\mathrm{vis}} \in \mathbb{R}^{S_v\times d}$ the projected visual tokens. The transformer input stacks them along the sequence: $r^{(0)} = \big[x_{\mathrm{text}}; x_{\mathrm{vis}}\big]$. Because the model relies on additive residual connections, the final output state $r^{(L)}$ is exactly the sum of this initial state and the accumulated updates from all subsequent layers:

$$r^{(L)} = r^{(0)} + \textstyle\sum_{l=1}^{L}\big(a^{(l)} + f^{(l)}\big). \tag{4}$$

This formulation highlights two critical points. First, the initial multimodal representation $r^{(0)}$ is a direct contributor to the final output logits z. Second, the sublayers are highly coupled; the key write-back paths—W_O (attention output) and W_{down} (MLP output)—inject new content into this shared stream, layer by layer, directly shifting the final predictions.

3.2 Which parts to tune?

When keeping the token embeddings, the LM head U, and layer-norm parameters frozen, Eq. 4 highlights the importance of two primary components: the initial state $r^{(0)}$ (driven by the **vision encoder** and **projector**) and the **language model** submodules (which accumulate into $r^{(L)}$).

Within the language model submodules, our selection of tunable parameters is guided by mechanistic studies of transformer components [14, 51]. Prior work

suggests that self-attention layers act primarily as routers that associate and organize information from the context, while MLPs behave like key–value memories that project activated features back into the residual stream, strongly influencing the next-token distribution. We hypothesize that much of rehearsal-free forgetting is driven by unconstrained write-back that biases this output distribution. Therefore, updating components that mainly change routing (SA Proj.) or updating only the MLP's feature detectors while freezing its write-back projection (W_{down}) ideally should help limit global output drift and reduce interference. Guided by this idea and combining Eqs. 2–4, we consider:

- **SA Proj.**: Update W_Q, W_K, W_V, W_O in all blocks (routing + write-back for attention).
- **SA Proj. (QKV)**: Freeze W_O to emphasize routing without directly modifying write-back.
- **MLP**: Update $W_{\text{gate}}, W_{\text{up}}, W_{\text{down}}$ (concept activation + write-back).
- **MLP (Gate & Up)**: Update $W_{\text{gate}}, W_{\text{up}}$ while freezing W_{down} to regulate write-back.

3.3 Training objective

Target task loss. We use next-token cross-entropy on the current target dataset with teacher forcing. For a batch $\mathcal{B} \subset \mathcal{D}_{\text{T}}^{(k)}$ at stage k,

$$\mathcal{L}_{\text{task}}(\theta) = \mathbb{E}_{(I,y)\sim\mathcal{B}}\Big[-\textstyle\sum_{t=1}^{|y|} \log p_\theta\big(y_t \mid y_{<t},\, x_{\text{vis}}\big)\Big], \qquad x_{\text{vis}} = g_\psi(f_{\text{vis}}(I)). \tag{5}$$

Learning-without-Forgetting (optional). To curb the output-distribution drift, we can enforce a KL-divergence constraint between the outputs of the current model at stage k with a frozen teacher model (checkpoint after stage $k-1$). Let θ_{k-1} be the frozen teacher and θ the current model tuned on $\mathcal{D}_{\text{T}}^{(k)}$. The objective is

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{task}}(\theta) \;+\; \lambda\, \mathcal{L}_{\text{distill}}(\theta; \theta_{k-1}), \tag{6}$$

with

$$\mathcal{L}_{\text{distill}}(\theta; \theta_{k-1}) = \mathbb{E}_{(I,y)\sim\tilde{\mathcal{B}}}\Big[\tfrac{\tau^2}{|\mathcal{S}(y)|} \textstyle\sum_{j\in\mathcal{S}(y)} \mathrm{KL}\Big(\mathrm{softmax}\big(\tfrac{z_{\theta_{k-1},j}}{\tau}\big) \,\big\|\, \mathrm{softmax}\big(\tfrac{z_{\theta,j}}{\tau}\big)\Big)\Big], \tag{7}$$

where $\tilde{\mathcal{B}} \subset \mathcal{D}_{\text{T}}^{(k)}$ is a target minibatch, τ is the distillation temperature, and $\mathcal{S}(y)$ is a uniformly random subset of positions with $|\mathcal{S}(y)| = \min(|y|, 1000)$ so we distill over many tokens while capping compute/memory. The coefficient λ balances fitting the new supervision against preserving the model's earlier behavior.

4 Experiment Design

Our experiments are designed to answer four questions: 1) **Where to tune?**—which components of an LMM can be updated to learn new skills while preserving

prior abilities (Sec. 5.1); 2) **Why does forgetting occur?**—whether performance loss is tied to a shift in the model's output distribution (Sec. 5.2); 3) **How generalizable is our selective tuning strategy**—whether the same selective-tuning recipes (SA Proj., MLP Gate&Up) transfer across model families (Sec. 5.3); 4) **How does our selective tuning compare to other simple forgetting mitigation approaches?** (Sec. 5.4). Due to space limits, we provide additional task/implementation details, per-task tables, and extended analyses in the Appendix.

4.1 Tasks and evaluation suites

Target tasks. We select **5 target tasks** that are common in real-world usage and are typically challenging for LMMs. Then we create a **default sequential-tuning task curriculum**: **1) Bird classification** from the CUB dataset [64]. We reformat the dataset following the instructions of [35] for training and evaluating LMMs; **2) Counting** from the PixmoCount dataset [10]; **3) Medical VQA** from the PathVQA dataset [16]; **4) OCR reading** from the TextVQA dataset [59] which has 34,602 training samples; **5) Time reading** from the TimeClock dataset [15]. In total, the curriculum contains 107,910 training samples, providing a comprehensive stress test for forgetting and knowledge transfer.

Held-out suite. To measure generalization beyond the training stream, we evaluate on eight held-out benchmarks: AI2D [21], ChartQA [45], DocVQA [47], InfoVQA [46], RealWorldQA [62], SeedBench [27], ScienceQA [41], and MMStar [7]. Several of these are multi-skill evaluation suites (e.g., SeedBench, MMStar), so the held-out average reflects a broad collection of capabilities rather than a single narrow dataset. InfoVQA and DocVQA use ANLS; since ANLS is in $[0, 1]$, we average it with accuracies from the other held-out tasks when reporting the mean held-out score.

4.2 Sequence-level metrics

We summarize performance over the five-stage curriculum with four metrics computed for every method.

- **Target Learning.** At each stage, consider only the task being tuned and measure its improvement over the base model on that task. We then average these stage-wise gains across all stages. This captures how well a method learns the task it is currently trained on.
- **Target Forgetting.** To measure forgetting on target tasks trained earlier in the sequence, we report the average difference between their accuracy immediately after they were trained and their accuracy at the end of the sequence. More negative means more forgetting.
- **Target Overall.** After training the full sequence, we compute the average performance change vs. the base model across all target tasks. This yields the net end-of-sequence effect on the target suite, combining the learned task and the previously learned targets.

Table 1: Effect of tuning different components: learning, forgetting, and overall performance, averaged over three five-task sequences. Cells are colored using a blue-orange colormap to show performance changes. Blue indicates a positive change, where a darker shade is better. Orange indicates a negative change, where a lighter shade is better. We underline numbers that do not reflect a significantly different task-average distribution from the best, based on a two-sided paired sample t-test.

Method	Target Learning	Target Forgetting	Target Overall	Held-out Forgetting
Baseline	43.9	0.0	43.9	76.4
Full	+29.9	−25.9	+9.2	−27.4
|- Vision Encoder	+9.6	−12.7	−0.5	−10.8
|- Projector	+2.3	−0.8	+1.7	−1.3
|- LLM	+31.8	−23.5	+13.0	−23.3
|- SA Proj.	+24.9	−2.3	+23.1	−0.6
|- SA Proj. (QKV)	+14.9	−0.5	+14.5	+0.2
|- MLP	+31.1	−19.5	+15.5	−15.7
|- MLP (Gate&Up)	+30.5	−4.2	+27.1	−2.1

– **Held-out Forgetting.** After training the full sequence of target tasks, we measure the average performance across all eight held-out benchmarks, in comparison to the base model. Negative values indicate forgetting on general vision–language ability; positive values indicate positive transfer.

5 Results

5.1 Component tuning on LLaVA-OneVision

Figure 1 previews learning (single–task tuning, left) and forgetting (held-out along the default sequence, right). Table 1 summarizes the four sequence-level metrics on LLaVA-OneVision for each component-tuning configuration, averaged over three five-task curricula whose task sequences and detailed results are in the Appendix. Entries are percentage-point deltas from the base model; the baseline row reports absolute scores. For each column, we run paired sample t-tests on the per-sequence/per-task averages to test whether tuning different components leads to significantly different per-task learning, forgetting, or overall performance. We underline numbers that are *not* significantly different ($p > 0.1$) than the best. Detailed single-task and sequential results, together with per-task performance tables, are provided in the Appendix.

From the table, we find the following patterns:

– **Full-model tuning attains large learning but maximizes forgetting**: Target Learning +29.9 is coupled with the worst Target/Held-out Forgetting (−25.9/−27.4).
– **Vision-side updates are weak or near-neutral**: the vision encoder yields a modest Target Learning +9.6 with negative Target Overall −0.5 and Held-out Forgetting −10.8; projector-only updates result in little change.
– **Language-model tuning has the best learning**: LLM shows the strongest Target Learning +31.8 and a solid Target Overall +13.0, but still substantial Target/Held-out Forgetting (−23.5/−23.3).

– **Self-attention projection is the most stable among LLM choices**: SA Proj. achieves high Target Overall +23.1 with minimal forgetting (Target −2.3, Held-out −0.6); the conservative variant without W_O further reduces forgetting (Target −0.5, Held-out +0.2) at the cost of learning (+14.5 Target Overall), indicating over-regularization when the attention write-back is frozen.
– **Regulating the MLP write-back offers the best balance**: MLP (Gate&Up) delivers near-maximal Target Learning +30.5 and the highest Target Overall +27.1 while keeping forgetting small (Target −4.2, Held-out −2.1); by contrast, full-MLP pushes learning slightly higher on the current task +31.1 but increases forgetting (Target −19.5, Held-out −15.7).

Overall, methods that mainly modify routing/associations (SA Proj. and SA Proj. (QKV)) or regulate how activated features are projected back (MLP Gate&Up) provide the most favorable learning–stability trade-off on LLaVA-OneVision.

5.2 Output-distribution probe (counting bias)

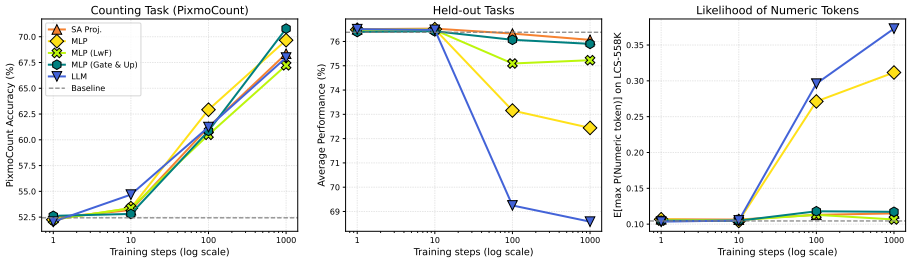

Fig. 3: Learning–forgetting tracks output–distribution shift. On LLaVA-OneVision tuned for counting, we plot five curves over log-spaced steps for LLM, SA Proj., MLP, MLP (Gate&Up) and MLP (LwF). The dashed line represents the base model. **Left:** PixmoCount accuracy rises for all methods. **Middle:** mean held-out performance drops sharply for LLM and MLP, remains nearly unchanged for SA Proj., and is preserved by MLP (LwF); **Right:** the expected likelihood of number tokens on non-counting captions (LCS-558K [34]) surges for LLM and MLP, stays near baseline for SA Proj., and has little changes for MLP (LwF).

Forgetting could be caused by interference or loss of access to memory (as commonly thought), or by biasing responses. We measure numeric-token bias (NTB) before, during, and after training on counting. Concretely, let C be a fixed set of numeric tokens (digits and common spelled numerals), and let $\mathcal{B} = \{(I, y)\}$ be a fixed held-out batch of captioning examples, where I is an image and y its reference caption. At training step s (with parameters θ_s), we generate a caption $\hat{y}$ for each image I using greedy decoding. At each decoding position j:

$$\mathrm{NTB}_s = \frac{1}{|\mathcal{B}|} \sum_{(I,y)\in\mathcal{B}} \left[\frac{1}{|\hat{y}|} \sum_{j=1}^{|\hat{y}|} \max_{v\in C} \; p_{\theta_s}\big(v \mid \hat{y}_{<j},\, x_{\mathrm{vis}}(I)\big) \right],$$

where $\hat{y}_{<j}$ denotes the generated prefix before position j and $p_{\theta_s}(\cdot \mid \hat{y}_{<j}, x_{\text{vis}}(I))$ is the next-token distribution.

Figure 3 shows that the likelihood of outputting numeric tokens for a captioning task is highly correlated with forgetting (Spearman $\rho = 0.84$; Supplementary Sec. B.4), over training steps and across methods. We further show the forgetting effect is also reversible with subsequent fine-tuning (Supplementary Sec. E.6), and layer-wise analysis attributes most of the induced logit shift to MLP blocks (Supplementary Fig. 5). These results support the hypothesis that response bias is a dominant cause of forgetting.

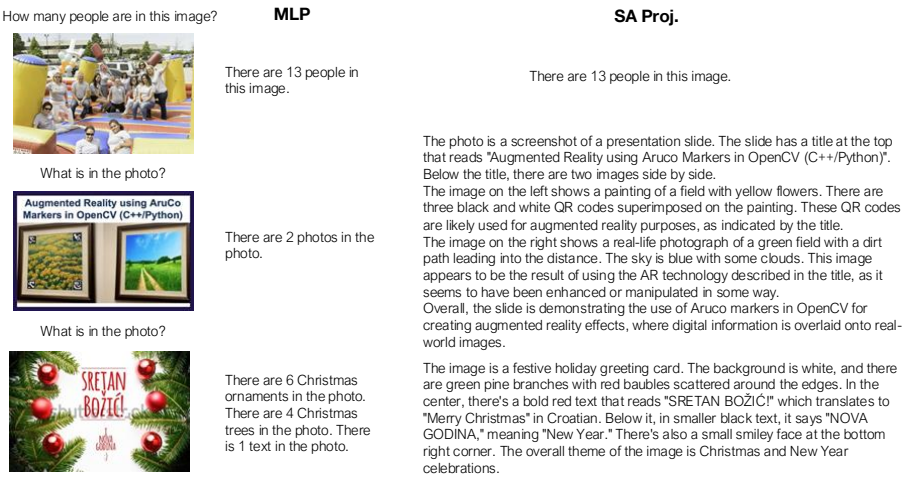

Fig. 4: Visualizations on counting and captioning examples after tuning MLP and SA Proj. on the counting task. The counting and examples are sampled from the PixmoCount dataset and the LCS-558K [34] dataset, respectively.

Qualitative results on counting and captioning. Figure 4 contrasts the output on counting and captioning tasks for models that have tuned MLP or SA Proj weights for 1K steps on PixmoCount. Both answer counting questions correctly, but the model with SA Proj tuning retains detailed captioning ability, while the MLP tuning leads to reframing captioning as counting statements, such as "There are 2 photos in the photo" (second row). This pattern suggests that MLP tuning does not erase visual understanding (objects and relations are still recognized) but biases the output distribution.

5.3 Beyond LLaVA-OneVision: generalization to other backbones

We further repeat the default five-task curriculum on two additional backbones, **LLaVA-NeXT** (LLaMA-3 8B) and **Qwen2.5-VL** (7B), using the same training protocol and sequence-level metrics as for LLaVA-OneVision. For vision side updates, we tune the vision encoder and projector jointly, since they form a single interface that produces the visual token sequence consumed by the language model.

Table 2: Component-level tuning experiments with **LLaVA-NeXT** and **Qwen2.5-VL**. "T" represents "Target" and "H" is for "Held-out". Underlined text of each column denotes the best method.

Method	LLaVA-NeXT (LLaMA-3 8B)				Qwen2.5-VL (7B)			
	T. Learn	T. Forget	T. Overall	H. Forget	T. Learn	T. Forget	T. Overall	H. Forget
Baseline	31.5	0.0	31.5	59.9	52.1	0.0	52.1	77.9
Full	+31.7	−20.3	+15.4	−32.0	+17.3	−5.2	+13.1	−17.5
|- Vision + Projector	+0.1	−1.8	−1.3	−13.4	+12.1	−9.1	+4.9	−6.2
|- LLM	+36.2	−21.2	+19.3	−35.9	+16.8	−5.9	+12.1	−24.6
|- SA Proj.	+28.3	−7.9	+21.9	−7.7	+16.1	−1.6	+14.9	+0.6
|- MLP	+34.9	−10.3	+26.6	−16.3	+17.7	−4.8	+13.9	−10.9
|- MLP ($W_{\text{gate}}, W_{\text{up}}$)	+28.0	−8.9	+20.9	−8.7	+16.8	+0.4	+17.1	−4.6

Across both backbones, the broad picture echoes LLaVA-OneVision: updating the language model is consistently effective for learning new skills; full-model and full-LLM tuning achieve large target task gains but come with the largest drops on held-out. Within the LM, two settings stand out as robust: self-attention projections deliver meaningful target learning with small held-out change, and MLP (Gate&Up) preserves most of the learning of full-MLP while limiting forgetting. There are, however, model-specific nuances worth noting. For example, on **LLaVA-NeXT**, MLP achieves the greatest target-overall improvement but incurs a noticeably larger held-out decrease than SA Proj. or Gate&Up, which remain the most stable choices. LLaVA-NeXT is much more susceptible to forgetting in general than the other models. On **Qwen2.5-VL**, SA Proj. is particularly stable: held-out performance is maintained or slightly improved; MLP (Gate&Up) attains the best target-overall score with near-zero target forgetting; vision + projector tuning also yields non-trivial target gains with moderate stability cost, in contrast to its weaker effect on LLaVA-NeXT. Though Qwen2.5-VL appears to learn less, it is worth noting that its baseline performance is much higher than that of other models. Overall, taking LLaVA-OneVision, LLaVA-NeXT, and Qwen2.5-VL together, the clearest cross-model takeaway is to prefer SA Proj. when stability on held-out is paramount and MLP (Gate&Up) when seeking near-maximal target learning with limited forgetting; projector-only updates are generally weak, and full-model / full-MLP tuning maximizes short-term gains at a clear stability cost.

5.4 Mitigating forgetting

Comparisons to Continual Learning Baselines. Recent continual learning methods for multimodal models often rely on highly specialized architectures or are restricted to discriminative VQA settings [32, 50, 71]. Because our goal is to evaluate open-ended generative capabilities across diverse skills without storing rehearsal data, we benchmark against the fundamental, widely applicable pillars of continual learning: Regularization (LwF) [30], Parameter-Efficient Tuning (LoRA) [17], Model Expansion (MoE), and Weight Averaging (WiSE-FT) [68]. See Supplementary Sec. F for method details.

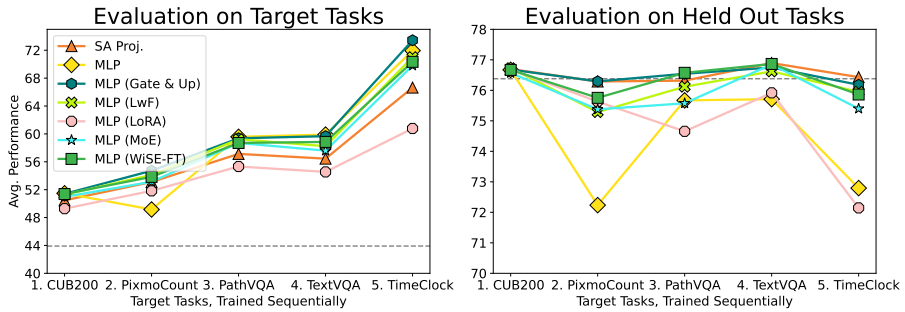

Fig. 5: Comparison of different continual learning techniques in the default sequential task curriculum. For LwF, WiSE-FT, only the MLP layers are tuned. LoRA adapters are wrapped only on the MLP layers. MoE is also applied to the MLP layers.

Figure 5 compares our selective tuning recipes to these baselines. Two patterns emerge. First, SA Proj. and MLP (Gate&Up) provide the best learning–stability trade-off: SA Proj. keeps held-out performance essentially flat while achieving meaningful target gains, and MLP (Gate&Up) delivers stronger target improvements with only a small held-out change, being substantially more stable than MLP. Second, among the compared methods, WiSE-FT can preserve held-out accuracy better than LwF but requires careful selection of task-dependent blending coefficients; LwF reliably curbs forgetting yet may impact target task gains; MoE and LoRA do not match the learning–stability balance of SA Proj. or MLP (Gate&Up), with LoRA often lagging behind on target performance. Also note that selective tuning is highly training-efficient. When evaluated on 4×H100 GPUs, our SA Proj. method achieves a throughput of 1.46 samples/sec, outperforming LoRA (1.27 samples/sec) and MoE (0.44 samples/sec), as it avoids the forward/backward pass overhead associated with adapter modules and routing logic (detailed in Supplementary Sec. E.5). Overall, selectively tuning SA Proj. or the MLP Gate&Up pair matches or exceeds these mitigation methods while remaining simple (no extra modules, no replay, no per-stage weight blending).

6 Conclusion

To answer our titular question, we can teach LMMs new skills by restricting tuning to SA Proj. or MLP Gate&Up layers or applying LwF-like knowledge distillation [29]. These approaches all limit forgetting by constraining output distribution shift, leading to strong performance in serial task learning while retaining existing capabilities. We demonstrate the strong relationship between measured forgetting and output shift with a counting bias probe and layerwise logit analysis, explaining how "forgetting" from fine-tuning on one task can be recovered by fine-tuning on another. These results and recommendations for how to manage the learning-forgetting trade-off hold across three model families, as validated with five target skills and eight broad benchmarks.

Limitations. A primary limitation of our work is that our evaluation does not extend to massive or commercial multimodal models. Furthermore, we have not tested whether the selective tuning recipes are still effective when scaling the training data to billions of samples.

Acknowledgment

This work is supported in part by ONR award N00014-23-1-2383, and U.S. DARPA ECOLE Program No. #HR00112390060. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of DARPA, ONR, or the U.S. Government. We used GPUs at NCSA Delta through allocation CIS230397 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by U.S. National Science Foundation grants 2138259, 2138286, 2138307, 2137603, and 2138296.

This research used both the DeltaAI advanced computing and data resource, which is supported by the National Science Foundation (award OAC 2320345) and the State of Illinois, and the Delta advanced computing and data resource which is supported by the National Science Foundation (award OAC 2005572) and the State of Illinois. Delta and DeltaAI are joint efforts of the University of Illinois Urbana-Champaign and its National Center for Supercomputing Applications.

References

1. Aljundi, R., Chakravarty, P., Tuytelaars, T.: Expert gate: Lifelong learning with a network of experts. In: CVPR. pp. 3366–3375 (2017)
2. Bai, S., Chen, K., Liu, X., Wang, J., Ge, W., Song, S., Dang, K., Wang, P., Wang, S., Tang, J., Zhong, H., Zhu, Y., Yang, M.H., Li, Z., Wan, J., Wang, P., Ding, W., Fu, Z., Xu, Y., Ye, J., Zhang, X., Xie, T., Cheng, Z., Zhang, H., Yang, Z., Xu, H., Lin, J.: Qwen2.5-VL Technical Report. arXiv preprint arXiv:2502.13923 (2025)
3. Bang, J., Kim, H., Yoo, Y., Ha, J.W., Choi, J.: Rainbow memory: Continual learning with a memory of diverse samples. In: CVPR. pp. 8218–8227 (2021)
4. Biderman, D., Portes, J., Ortiz, J.J.G., Paul, M., Greengard, P., Jennings, C., King, D., Havens, S., Chiley, V., Frankle, J., Blakeney, C., Cunningham, J.P.: LoRA learns less and forgets less. TMLR (2024)
5. Chaudhry, A., Ranzato, M., Rohrbach, M., Elhoseiny, M.: Efficient lifelong learning with A-GEM. In: ICLR (2019)
6. Chen, C., Zhu, J., Luo, X., Shen, H., Song, J., Gao, L.: CoIN: A Benchmark of Continual Instruction Tuning for Multimodel Large Language Models. In: NeurIPS (2024)
7. Chen, L., Li, J., Dong, X., Zhang, P., Zang, Y., Chen, Z., Duan, H., Wang, J., Qiao, Y., Lin, D., Zhao, F.: Are we on the right way for evaluating large vision-language models? In: Globersons, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J.M., Zhang, C. (eds.) NIPS (2024)
8. Chen, Z., Liu, B.: Lifelong machine learning. Synthesis Lectures on Artificial Intelligence and Machine Learning **12**(3), 1–207 (2018)

9. Clark, K., Khandelwal, U., Levy, O., Manning, C.D.: What does bert look at? an analysis of bert's attention. arXiv preprint arXiv:1906.04341 (2019)
10. Deitke, M., Clark, C., Lee, S., Tripathi, R., Yang, Y., Park, J.S., Salehi, M., Muennighoff, N., Lo, K., Soldaini, L., Lu, J., Anderson, T., Bransom, E., Ehsani, K., Ngo, H., Chen, Y., Patel, A., Yatskar, M., Callison-Burch, C., Head, A., Hendrix, R., Bastani, F., VanderBilt, E., Lambert, N., Chou, Y., Chheda, A., Sparks, J., Skjonsberg, S., Schmitz, M., Sarnat, A., Bischoff, B., Walsh, P., Newell, C., Wolters, P., Gupta, T., Zeng, K.H., Borchardt, J., Groeneveld, D., Nam, C., Lebrecht, S., Wittlif, C., Schoenick, C., Michel, O., Krishna, R., Weihs, L., Smith, N.A., Hajishirzi, H., Girshick, R., Farhadi, A., Kembhavi, A.: Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models. In: CVPR. pp. 91–104 (2025)
11. Del Chiaro, R., Twardowski, B., Bagdanov, A., Van de Weijer, J.: Ratt: Recurrent attention to transient tasks for continual image captioning. In: NeurIPS. vol. 33, pp. 16736–16748 (2020)
12. Douillard, A., Cord, M., Ollion, C., Robert, T., Valle, E.: Podnet: Pooled outputs distillation for small-tasks incremental learning. In: ECCV. pp. 86–102 (2020)
13. Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., et al.: A mathematical framework for transformer circuits. Transformer Circuits Thread **1**(1), 12 (2021)
14. Geva, M., Schuster, R., Berant, J., Levy, O.: Transformer feed-forward layers are key-value memories. In: Moens, M., Huang, X., Specia, L., Yih, S.W. (eds.) EMNLP. pp. 5484–5495 (2021)
15. Gpiosenka: TIME - Image Dataset-Classification. https://www.kaggle.com/datasets/gpiosenka/time-image-datasetclassification, accessed: 29 June 2026
16. He, X., Zhang, Y., Mou, L., Xing, E.P., Xie, P.: Pathvqa: 30000+ questions for medical visual question answering. CoRR **abs/2003.10286** (2020)
17. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: LoRA: Low-Rank Adaptation of Large Language Models. In: ICLR (2022)
18. Huang, J., Cui, L., Wang, A., Yang, C., Liao, X., Song, L., Yao, J., Su, J.: Mitigating catastrophic forgetting in large language models with self-synthesized rehearsal. In: Ku, L., Martins, A., Srikumar, V. (eds.) ACL. pp. 1416–1428 (2024)
19. Jin, W., Cheng, Y., Shen, Y., Chen, W., Ren, X.: A good prompt is worth millions of parameters: Low-resource prompt-based learning for vision-language models. In: Muresan, S., Nakov, P., Villavicencio, A. (eds.) ACL. pp. 2763–2775 (2022)
20. Joseph, K.J., Khan, S., Khan, F.S., Anwer, R.M., Balasubramanian, V.N.: Energy-based latent aligner for incremental learning. In: CVPR. pp. 7452–7461 (2022)
21. Kembhavi, A., Salvato, M., Kolve, E., Seo, M.J., Hajishirzi, H., Farhadi, A.: A diagram is worth a dozen images. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV. vol. 9908, pp. 235–251. Springer (2016)
22. Khattak, M.U., Rasheed, H.A., Maaz, M., Khan, S.H., Khan, F.S.: Maple: Multi-modal prompt learning. In: CVPR (2023)
23. Khattak, M.U., Wasim, S.T., Naseer, M., Khan, S., Yang, M., Khan, F.S.: Self-regulating prompts: Foundational model adaptation without forgetting. In: ICCV (2023)
24. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al.: Overcoming catastrophic forgetting in neural networks. PNAS **114**(13), 3521–3526 (2017)

25. Li, B., Zhang, K., Zhang, H., Guo, D., Zhang, R., Li, F., Zhang, Y., Liu, Z., Li, C.: Llava-next: Stronger llms supercharge multimodal capabilities in the wild (May 2024), https://llava-vl.github.io/blog/2024-05-10-llava-next-stronger-llms/, accessed: 29 June 2026
26. Li, B., Zhang, Y., Guo, D., Zhang, R., Li, F., Zhang, H., Zhang, K., Li, Y., Liu, Z., Li, C.: LLaVA-OneVision: Easy Visual Task Transfer. arXiv preprint arXiv:2408.03326 (2024)
27. Li, B., Wang, R., Wang, G., Ge, Y., Ge, Y., Shan, Y.: Seed-bench: Benchmarking multimodal llms with generative comprehension. arXiv preprint arXiv:2307.16125 (2023)
28. Li, X., Ren, W., Qin, W., Wang, L., Zhao, T., Hong, R.: Analyzing and reducing catastrophic forgetting in parameter efficient tuning. In: ICASSP. pp. 1–5. IEEE (2025)
29. Li, Z., Hoiem, D.: Learning without forgetting. PAMI **40**(12), 2935–2947 (2018)
30. Li, Z., Hoiem, D.: Learning without forgetting. TPAMI (2017)
31. Lin, Y., Lin, H., Xiong, W., Diao, S., Liu, J., Zhang, J., Pan, R., Wang, H., Hu, W., Zhang, H., Dong, H., Pi, R., Zhao, H., Jiang, N., Ji, H., Yao, Y., Zhang, T.: Mitigating the Alignment Tax of RLHF. In: EMNLP (2024)
32. Lin, Y., Qi, M., Liu, L., Ma, H.: Vlm-assisted continual learning for visual question answering in self-driving. arXiv preprint arXiv:2502.00843 (2025)
33. Liu, H., Li, C., Wu, Q., Lee, Y.J.: Visual Instruction Tuning. In: NeurIPS (2023)
34. Liu, H., Li, C., Wu, Q., Lee, Y.J.: Visual instruction tuning. NeurIPS **36**, 34892–34916 (2023)
35. Liu, H., Xiao, L., Liu, J., Li, X., Feng, Z., Yang, S., Wang, J.: Revisiting mllms: An in-depth analysis of image classification abilities. arXiv preprint arXiv:2412.16418 (2024)
36. Liu, W., Zhu, F., Wei, L., Tian, Q.: C-clip: Multimodal continual learning for vision-language model. In: ICLR (2025)
37. Liu, Y., Li, Y., Schiele, B., Sun, Q.: Online hyperparameter optimization for class-incremental learning. In: AAAI (2023)
38. Liu, Y., Li, Y., Schiele, B., Sun, Q.: Wakening past concepts without past data: Class-incremental learning from online placebos. In: WACV. pp. 2226–2235 (2024)
39. Liu, Y., Schiele, B., Sun, Q.: Adaptive aggregation networks for class-incremental learning. In: CVPR. pp. 2544–2553 (2021)
40. Liu, Y., Su, Y., Liu, A., Schiele, B., Sun, Q.: Mnemonics training: Multi-class incremental learning without forgetting. In: CVPR. pp. 12245–12254 (2020)
41. Lu, P., Mishra, S., Xia, T., Qiu, L., Chang, K., Zhu, S., Tafjord, O., Clark, P., Kalyan, A.: Learn to explain: Multimodal reasoning via thought chains for science question answering. In: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (eds.) NIPS (2022)
42. Luo, Y., Yang, Z., Meng, F., Li, Y., Zhou, J., Zhang, Y.: An empirical study of catastrophic forgetting in large language models during continual fine-tuning. arXiv preprint arXiv:2308.08747 (2023)
43. Luo, Z., Liu, Y., Schiele, B., Sun, Q.: Class-incremental exemplar compression for class-incremental learning. In: CVPR. pp. 11371–11380 (2023)
44. Marouf, I.E., Tartaglione, E., Lathuiliere, S., van de Weijer, J.: No images, no problem: Retaining knowledge in continual vqa with questions-only memory. arXiv preprint arXiv:2502.04469 (2025)
45. Masry, A., Long, D.X., Tan, J.Q., Joty, S.R., Hoque, E.: Chartqa: A benchmark for question answering about charts with visual and logical reasoning. In: Muresan, S., Nakov, P., Villavicencio, A. (eds.) ACL. pp. 2263–2279 (2022)

46. Mathew, M., Bagal, V., Tito, R., Karatzas, D., Valveny, E., Jawahar, C.V.: Infographicvqa. In: WACV. pp. 2582–2591. IEEE (2022)
47. Mathew, M., Karatzas, D., Jawahar, C.V.: Docvqa: A dataset for VQA on document images. In: WACV. pp. 2199–2208. IEEE (2021)
48. Meng, K., Bau, D., Andonian, A., Belinkov, Y.: Locating and editing factual associations in GPT. NeurIPS **36** (2022), arXiv:2202.05262, doi:10.48550/arXiv.2202.05262
49. Nguyen, G., Jun, T.J., Tran, T., Yalew, T., Kim, D.: Contcap: A scalable framework for continual image captioning. arXiv preprint arXiv:1909.08745 (2019)
50. Nikandrou, M., Pantazopoulos, G., Konstas, I., Suglia, A.: Enhancing continual learning in visual question answering with modality-aware feature distillation. arXiv preprint arXiv:2406.19297 (2024)
51. Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Johnston, S., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., Olah, C.: In-context learning and induction heads. arXiv preprint arXiv:2209.11895 (2022)
52. PourKeshavarzi, M., Zhao, G., Sabokrou, M.: Looking back on learned experiences for class/task incremental learning. In: ICLR (2022)
53. Prabhu, A., Torr, P.H., Dokania, P.K.: GDumb: A simple approach that questions our progress in continual learning. In: ECCV. pp. 524–540 (2020)
54. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. In: ICML (2021)
55. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: iCARL: Incremental classifier and representation learning. In: CVPR (2017)
56. Shenfeld, I., Pari, J., Agrawal, P.: Rl's razor: Why online reinforcement learning forgets less (2025), https://arxiv.org/abs/2509.04259, accessed: 29 June 2026
57. Shin, H., Lee, J.K., Kim, J., Kim, J.: Continual learning with deep generative replay. In: NeurIPS. pp. 2990–2999 (2017)
58. Simon, C., Koniusz, P., Harandi, M.: On learning the geodesic path for incremental learning. In: CVPR. pp. 1591–1600 (2021)
59. Singh, A., Natarajan, V., Shah, M., Jiang, Y., Chen, X., Batra, D., Parikh, D., Rohrbach, M.: Towards VQA models that can read. In: CVPR. pp. 8317–8326 (2019)
60. Smith, J.S., Karlinsky, L., Gutta, V., Cascante-Bonilla, P., Kim, D., Arbelle, A., Panda, R., Feris, R., Kira, Z.: Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In: CVPR (2023)
61. Tao, X., Chang, X., Hong, X., Wei, X., Gong, Y.: Topology-preserving class-incremental learning. In: ECCV. pp. 254–270 (2020)
62. visheratin: RealWorldQA: Benchmark for real-world spatial understanding (2024)
63. Voita, E., Talbot, D., Moiseev, F., Sennrich, R., Titov, I.: Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In: ACL. pp. 5797–5808 (2019). https://doi.org/10.18653/v1/P19-1580
64. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-ucsd birds-200-2011 dataset. Tech. Rep. CNS-TR-2011-001, California Institute of Technology (2011)
65. Wang, F.Y., Zhou, D.W., Ye, H.J., Zhan, D.C.: Foster: Feature boosting and compression for class-incremental learning. In: ECCV (2022)

66. Wang, X., Chen, T., Ge, Q., Xia, H., Bao, R., Zheng, R., Zhang, Q., Gui, T., Huang, X.: Orthogonal subspace learning for language model continual learning. In: Bouamor, H., Pino, J., Bali, K. (eds.) Findings of EMNLP (2023)
67. Wei, X., Munir, M., Marculescu, R.: Mitigating intra- and inter-modal forgetting in continual learning of unified multimodal models. In: NeurIPS (2025)
68. Wortsman, M., Ilharco, G., Kim, J.W., Li, M., Kornblith, S., Roelofs, R., Lopes, R.G., Hajishirzi, H., Farhadi, A., Namkoong, H., Schmidt, L.: Robust fine-tuning of zero-shot models. In: CVPR (2022)
69. Xiang, J., Tao, T., Gu, Y., Shu, T., Wang, Z., Yang, Z., Hu, Z.: Language models meet world models: Embodied experiences enhance language models. In: NeurIPS. vol. 36, pp. 75392–75412 (2023)
70. Yu, J., Zhuge, Y., Zhang, L., Hu, P., Wang, D., Lu, H., He, Y.: Boosting Continual Learning of Vision-Language Models via Mixture-of-Experts Adapters. In: CVPR (2024)
71. Zhang, X., Zhang, F., Xu, C.: Vqacl: A novel visual question answering continual learning setting. In: CVPR. pp. 19102–19112 (2023)
72. Zhao, H., Zhu, F., Wang, R., Meng, G., Zhang, Z.: MLLM-CL: Continual learning for multimodal large language models. In: NeurIPS (2025)
73. Zheng, Z., Ma, M., Wang, K., Qin, Z., Yue, X., You, Y.: Preventing zero-shot transfer degradation in continual learning of vision-language models. In: CVPR. pp. 19125–19136 (2023)
74. Zhou, D.W., Zhang, Y., Wang, Y., Ning, J., Ye, H.J., Zhan, D.C., Liu, Z.: Learning without forgetting for vision-language models. TPAMI (2025)
75. Zhu, D., Sun, Z., Li, Z., Shen, T., Yan, K., Ding, S., Wu, C., Kuang, K.: Model tailor: Mitigating catastrophic forgetting in multi-modal large language models. In: ICML (2024)
76. Zhu, Z., Gong, Y., Hoiem, D.: Anytime Continual Learning for Open Vocabulary Classification. In: ECCV. vol. 15064 (2024)
77. Zhu, Z., Lyu, W., Xiao, Y., Hoiem, D.: Continual learning in open-vocabulary classification with complementary memory systems. Trans. Mach. Learn. Res. **2024** (2024)