M^{2}Note: Continual Evolution of Vision Language Models via Mistake Notebook Learning

Haiwen Li[*] **Jing Tang**[†] **Rui Chen** **Lei Sun** **Zining Chen** **Xiangxiang Chu**
AMAP, Alibaba Group

Abstract

Vision Language Models (VLMs) have demonstrated remarkable capabilities in multimodal reasoning tasks, yet they still suffer from recurring failures, such as skipping key visual checks, misapplying domain rules, and hallucinating unsupported concepts. Most existing solutions rely on supervised fine-tuning (SFT) and reinforcement learning (RL), which are expensive to iterate and can be brittle under distribution shift. To this end, we propose **M**ultimodal **M**istake **Note**book Learning (M^{2}Note), a training-free continual evolution framework that externalizes learning into an editable memory. M^{2}Note transforms failed trajectories into compact subject–guidance notes: the subject summarizes the underlying domain and concept, while the guidance provides actionable verification steps that can be reused in future inference. At test time, M^{2}Note retrieves relevant notes via multimodal retrieval-augmented generation (RAG) and appends them to the model context, steering reasoning away from previously observed pitfalls. To stabilize continual evolution, we adopt batch-level post-verification with rollback, which commits notebook edits only if they improve performance on the same batch, reducing noisy updates and preventing regressions. M^{2}Note supports both *self-evolving*, where the same VLM acts as solver and supervisor, and *cross-model evolving*, where a stronger supervisor guides a weaker solver, enabling capability transfer without weight updates. Experiments on six multimodal reasoning benchmarks show consistent improvements across domains and backbones, while achieving strong cost and sample efficiency and remaining complementary to Chain-of-Thought (CoT) prompting. The code is available at *https://github.com/AMAP-ML/M2Note*.

[*]Work done during the internship at AMAP, Alibaba.
[†]Project Lead.

1 Introduction

Vision Language Models (VLMs) (Bordes et al., 2024; Hurst et al., 2024; Bai et al., 2025a; Guo et al., 2025; Liu et al., 2023) have become a general interface for multimodal tasks, including STEM reasoning (Yue et al., 2024; Lu et al., 2023; Wang et al., 2024; Qiao et al., 2025), chart and diagram understanding (Masry et al., 2022; Kembhavi et al., 2016), document OCR (Liu et al., 2024c; Yang et al., 2025), visual question answering (Liu et al., 2024b; Chen et al., 2024; xAI; Bai et al., 2025b), and video understanding (Li et al., 2024; Zhao et al., 2025; Yuan et al., 2026). Despite strong progress, VLMs in realistic settings still exhibit repeatable failure modes, such as missing key visual evidence or over-relying on superficial cues. Correcting such behaviors efficiently, without sacrificing robustness or requiring costly retraining, remains a practical challenge.

The dominant adaptation paradigm is parameter-based post-training, such as supervised fine-tuning (SFT) (Liu et al., 2023; Wei et al., 2022a) and reinforcement learning (RL) (Schulman et al., 2017; Rafailov et al., 2023; Shao et al., 2024; Yu et al., 2025). Although effective, these methods are expensive, slow to iterate, and prone to regression. More importantly, once model weights are updated, test-time behavior becomes fixed, making continual improvement difficult in dynamic environments. This motivates growing interest in training-free adaptation through in-context steering.

Existing training-free approaches mainly fall into two categories: ❶ prompt optimization methods (Zhou et al., 2022; Yang et al., 2023; Pryzant et al., 2023) refine global instructions, but often provide advice that is too coarse to target recurring errors across diverse multimodal tasks; ❷ memory-based methods (Zhao et al., 2024; Zhang et al., 2024a; Shinn et al., 2023; Zhang et al., 2025a) store instance-level experiences for retrieval, but

Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing, pages 16611–16629
October 24-29, 2026 ©2026 Association for Computational Linguistics

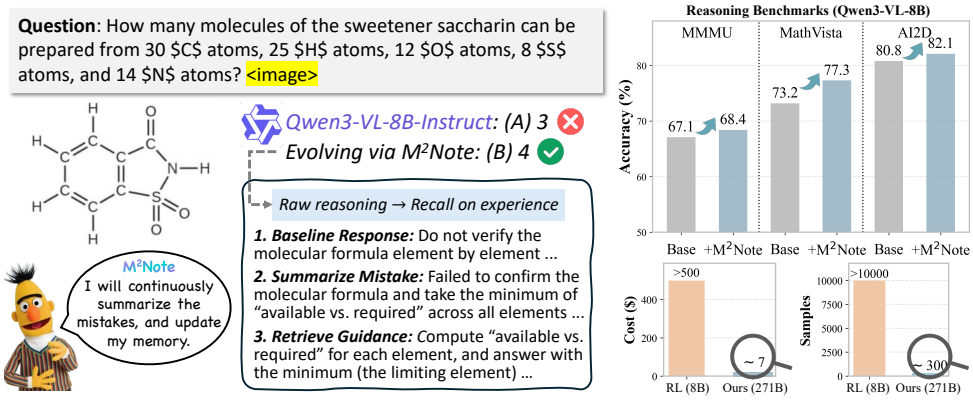


Figure 1: **Overview of M²Note**. (Left) The system updates an external mistake notebook from incorrect responses and retrieves task-relevant guidance at inference time to refine reasoning, for instance, limiting-reagent counting for saccharin. (Right) Accuracy gains on different benchmarks (e.g., MMMU (Yue et al., 2024), MathVista (Lu et al., 2023), and AI2D (Kembhavi et al., 2016)) via M²Note, together with cost- and sample-efficiency comparisons.

they often lack abstraction, leading to redundant memories and limited generalization beyond superficially similar cases. These limitations are especially pronounced in multimodal settings, where visual inputs are highly diverse and sparse.

To address this, we propose **M**ultimodal **M**istake **Note**book Learning (M²Note), a tuning-free framework, which extends the learning-from-mistakes paradigm (Zhang et al., 2024a; Su et al., 2025) to VLMs. M²Note maintains an external mistake notebook that stores structured *subject-guidance* pairs and retrieves relevant notes via multimodal embeddings (Li et al., 2026; Zhang et al., 2024b) to augment the model context. Given a multimodal query, the VLM first recalls relevant notes and produces an answer. If the answer is incorrect, a reflective supervisor summarizes the failure into a high-level subject and concise actionable guidance, then writes or merges the resulting note into the notebook. In this way, recurring errors are distilled into compact, reusable experience, shifting adaptation from model parameter updates to semantic updates of an external notebook.

Beyond being training-free, M²Note is designed as a closed-loop evolution protocol: *generate* → *reflect* → *update* → *verify*. Whenever notebook updates are proposed, the system re-evaluates the same batch with the updated notebook and accepts the change only if performance improves; otherwise, it rolls back. This conservative *accept-if-improves* mechanism stabilizes continual evolution and suppresses noisy updates. The framework supports both *self-evolving*, where the same VLM acts as solver and supervisor, and *cross-model evolving*, where a stronger VLM supervises a weaker one to transfer knowledge without any parameter updates.

Experiments on six multimodal reasoning benchmarks show that M²Note consistently improves VLM performance. Compared with RL-based self-evolving methods such as VisPlay (He et al., 2025), EvoLMM (Thawakar et al., 2025), and Vision-Zero (Wang et al., 2025a), M²Note achieves strong gains with substantially lower training cost and fewer samples, since it evolves only a compact external notebook rather than model parameters. We introduce the following technical components:

- We present M²Note, a multimodal mistake notebook learning framework that improves VLM reasoning by storing and retrieving subject-level guidance in an external memory.
- We propose a stable closed-loop evolution protocol with batch-level verification, supporting both self-evolving and cross-model evolving settings while remaining tuning-free.
- We deliver consistent gains across six multimodal reasoning benchmarks, with strong cost and sample efficiency compared with SFT- and RL-based methods.

2 Related Work

VLMs and Multimodal Reasoning. Multimodal large language models (MLLMs) (Radford et al., 2021; Bai et al., 2025a; Liu et al., 2024a; Wang et al., 2025b; Hurst et al., 2024; Comanici et al., 2025; Chen et al., 2025; Sun et al., 2026) extend LLMs with visual perception and have achieved strong performance on multimodal understanding and reasoning tasks.

Benchmarks such as MMMU (Yue et al., 2024), MathVista (Lu et al., 2023), AI2D (Kembhavi et al., 2016), ChartQA (Masry et al., 2022), MM-Star (Chen et al., 2024), and RealworldQA (xAI) evaluate challenging abilities including STEM reasoning, diagram understanding, and real-world visual reasoning.

Self-Evolving and Training-Free Adaptation. Recent self-evolving methods improve models via closed-loop interaction, often by assigning co-evolving roles (e.g., a challenger and a solver) to the same base model. Representative examples (Huang et al., 2025; Wang et al., 2025a; He et al., 2025; Xu et al., 2026; Jia et al., 2026) use self-play to generate increasingly informative training signals and typically internalize the resulting improvements through parameter updates, such as supervised fine-tuning (Wei et al., 2022a; Zhang et al., 2026) and reinforcement learning (Schulman et al., 2017; Shao et al., 2024; Dai et al., 2026; Ji et al., 2025). While effective, these methods are costly and can be brittle under distribution shift (Chen et al., 2023; Zeng et al., 2024; Zhai et al., 2025). Training-free alternatives instead adapt models through inference-time context, primarily including prompt optimization (Wei et al., 2022b; Pryzant et al., 2023; Zhou et al., 2022; Yang et al., 2023) and memory-augmented inference (Shinn et al., 2023; Zhao et al., 2024; Zhou et al., 2025; Zhang et al., 2025a; Cai et al., 2025; Ouyang et al., 2025). Within the latter, learning-from-mistakes methods (Madaan et al., 2023; Zhang et al., 2024a; Zhao et al., 2024; Su et al., 2025) convert failures into reusable vexperience for future inference. Our work extends this paradigm to VLMs by distilling multimodal failures into compact subject-level guidance and updating memory with batch-level verification.

3 Methodology

3.1 Overview

We propose **M**ultimodal **M**istake **Note**book Learning (M^{2}Note), a training-free framework that improves VLMs by maintaining an external mistake notebook. Unlike prior context-optimization methods mainly developed for text-only settings (Su et al., 2025; Zheng et al.; Wang et al., 2025c; Zhang et al., 2024a; Zhou et al., 2025), M^{2}Note extends the *learning-from-mistakes* (Su et al., 2025; Zhang et al., 2024a) paradigm to multimodal tasks.

As illustrated in Figure 2, M^{2}Note involves two roles: a **Tuning Model** (π_θ), which solves multimodal problems with retrieved notebook guidance, and a **Tuner Model** (π_{tuner}), which analyzes the Tuning Model's errors and writes reusable guidance back into the notebook. The notebook is initialized as an external memory $\mathcal{M}$ that stores structured triplets (s, g, e), where s denotes the subject, summarizing the underlying task and domain, g denotes the corresponding guidance, i.e., concise recommendations associated with that subject, and $e \in \mathbb{R}^d$ represents the embedding produced by a multimodal embedding model for retrieval. Examples of the notes are provided in Appendix B.

The framework runs in a closed loop with three steps: ❶ the Tuning Model retrieves relevant notes from $\mathcal{M}$ and generates baseline responses; ❷ the Tuner Model summarizes failed cases into new subject-guidance notes, and decides whether to add or merge them; ❸ the system reruns the same batch with the updated memory and accepts the update only if the new memory improves batch-level performance; otherwise, it rolls back.

3.2 Problem Formulation

M^{2}Note can be formulated as a context optimization problem: instead of updating the parameters of π_θ, we refine only the external memory to maximize the expected reward, ensuring that the retrieved guidance provides effective assistance. Formally, let $\mathcal{D}$ denote a data distribution over multimodal queries (q, i), optionally paired with y (when the training split is available), where q denotes the text question, i the associated image, and y the optional ground-truth answer when available. We seek an optimal memory $\mathcal{M}^*$:

$$\begin{aligned} \mathcal{M}^* &= \arg\max_{\mathcal{M}} \ \mathbb{E}_{(q,i)\sim\mathcal{D}}\Big[R\big(\pi_\theta(z)\big)\Big], \\ &\text{where } z = (q, i) \oplus \mathrm{Ret}((q, i), \mathcal{M}), \end{aligned} \tag{1}$$

where $\oplus$ is context concatenation, and $\mathrm{Ret}(\cdot, \cdot)$ represents retrieving guidance from the notebook. We consider two instantiations of the reward function $R(\cdot)$: ❶ **Supervised setting.** If ground-truth labels y are available, the model response is first parsed into a final prediction, and $R(\cdot)$ is instantiated as a binary reward by comparing the parsed prediction against the ground truth, i.e., it returns 1 if they match and 0 otherwise; ❷ **Label-free setting.** If labels are unavailable, $R(\cdot)$ adopts an LLM-as-a-Judge (Gu et al., 2024), which evaluates whether the response is correct given the input (and task

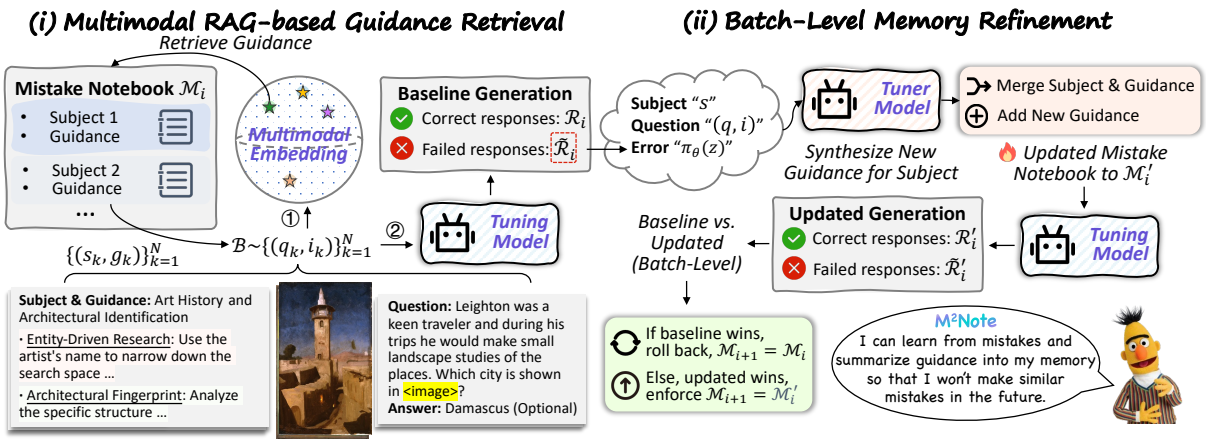


Figure 2: **The M²Note evolving protocol.** M²Note improves a VLM through a closed loop with two roles: a Tuning Model for solving multimodal queries with retrieved notebook guidance, and a Tuner Model for analyzing mistakes and refining the notebook. The process consists of (i) **Multimodal RAG-based Guidance Retrieval**, where the Tuning Model retrieves relevant subject-guidance notes from memory to generate responses, and (ii) **Batch-level Memory Refinement**, where the Tuner Model summarizes failures into new or merged notes, after which the update is verified and accepted only if it improves batch-level performance.

instruction if available), and outputs a binary judgment as reward. In both cases, the optimization process is delegated to π_{tuner}, which identifies failures, summarizes guidance, and updates the memory.

3.3 The M²Note Evolution Protocol

As illustrated in Figure 2, M²Note follows a closed-loop protocol that iteratively retrieves guidance, updates memory from mistakes, and verifies whether the update should be kept.

Multimodal RAG-based Guidance Retrieval. The Tuning Model π_θ uses the current memory $\mathcal{M}_i$ in a multimodal RAG manner (Lewis et al., 2020; Mei et al., 2025). For the i-th iteration, given an input batch $\mathcal{B}_i = \{(q_k, t_k)\}_{k=1}^N$, each query (q_k, t_k) is fed into a multimodal embedding model $\mathcal{E}$ to obtain an embedding $e_k = \mathcal{E}(q_k, i_k) \in \mathbb{R}^d$. The system then computes the cosine similarity between e_k and each note embedding in memory, ranks all notes by cosine similarity, and returns the top-K results (the following shows $K = 1$):

$$(s_k, g_k) = \arg\max_{(s,g)} \left\{ \frac{\langle e_k, e \rangle}{\|e_k\| \, \|e\|} \;\middle|\; (s, g, e) \in \mathcal{M}_i, \; \frac{\langle e_k, e \rangle}{\|e_k\| \, \|e\|} > \alpha \right\} \tag{2}$$

where $\langle e_k, e \rangle$ denotes the inner product; $\|\cdot\|$ denotes the ℓ_2 normalization; and α controls the relevance of the retrieved content.

The retrieved notes are concatenated as advisory context to the VLM input, i.e., $z_k = (q_k, i_k) \oplus (s_k, g_k)$. In addition, the Tuning Model is instructed to incorporate the extra context critically rather than follow it blindly, which helps to reduce hallucinations (Appendix A.2, Table 5 and Figure 5). The initial response, denoted as $\pi_\theta(z_k)$, is fed into the reward function $R(\cdot)$. Aggregating and partitioning the responses within the batch yields two mutually exclusive sets: correct responses are collected into $\mathcal{R}_i = \{k \mid R(\pi_\theta(z_k)) = 1\}$, while failed trajectories form $\tilde{\mathcal{R}}_i = \{k \mid R(\pi_\theta(z_k)) = 0\}$. The batch accuracy is then computed as the performance baseline for the current iteration.

Batch-Level Memory Refinement. Given the failure set $\tilde{\mathcal{R}}_i$, the Tuner Model π_{tuner} summarizes reusable guidance and proposes notebook updates. An update is accepted only if it improves batch-level performance; otherwise, it is rolled back.

(1) *Subject Extraction* σ. For each failed sample index $k \in \tilde{\mathcal{R}}_i$, the system infers a *subject* that summarizes the underlying concept of the query. Concretely, as shown below, we apply a prompted operator $\sigma(\cdot)$ implemented by π_{tuner}:

$$s_k' = \pi_{\text{tuner}}(\sigma(q_k, i_k)), \quad k \in \tilde{\mathcal{R}}_i. \tag{3}$$

The subject is expected to be domain-aware (e.g., math, chemistry, or diagram understanding), specific enough to capture recurring error patterns, but not tied to instance-specific details (e.g., exact entities or numbers).

(2) *Guidance Synthesis* ϕ. Given a failed trajectory, π_{tuner} further generates a guidance that is directly actionable at inference time. The guidance

is distilled from the failed response $\pi_\theta(z_k)$ and the available feedback, abstracting the cause of failure into verification-oriented steps:

$$g'_k = \pi_{\text{tuner}}(\phi((q_k, i_k), \pi_\theta(z_k))),\ k \in \tilde{\mathcal{R}}_i, \quad (4)$$

where $\phi(\cdot)$ is a prompted "mistake-to-guidance" operator. In practice, g'_k is constrained to be concise, structured, and oriented toward checks and invariants (e.g., "use the artist's name to narrow down the search space" in Figure 2).

(3) *Subject Merging and Memory Update* μ. To avoid fragmentation, M^{2}Note optionally merges new notes with existing ones. For each candidate note (s'_k, g'_k), the tuner predicts an edit action:

$$a_k = \pi_{\text{tuner}}(\mu(s'_k, s_k)) \in \{\texttt{add}, \texttt{merge}\}. \quad (5)$$

If $a_k = \texttt{Merge}$, π_{tuner} synthesizes the merged subject s^*_k and the merged guidance g^*_k; otherwise, they are set to s'_k and g'_k, respectively. For each newly created or merged note, we compute its embedding as $e^*_k = \mathcal{E}(s^*_k \oplus g^*_k)$. The updated memory $\mathcal{M}'_i$ is obtained by incorporating all entries $\{(s^*_k, g^*_k, e^*_k)\}_{k=1}^N$ derived from this batch of data.

(4) *Batch-Level Post-Verification.* After proposing $\mathcal{M}'_i$, we rerun the same batch using the updated memory. The batch-level accuracy before and after the update is computed by the formula:

$$\text{Acc}(\mathcal{B}_i|\mathcal{M}) = \frac{1}{N}\sum_{k=1}^{N} R(\pi_\theta(z_k|\mathcal{M})). \quad (6)$$

We accept the update if it improves performance:

$$\mathcal{M}_{i+1} = \begin{cases} \mathcal{M}'_i, & \text{if } \text{Acc}(\mathcal{B}_i|\mathcal{M}'_i) \geq \text{Acc}(\mathcal{B}_i|\mathcal{M}_i), \\ \mathcal{M}_i, & \text{otherwise (rollback).} \end{cases} \quad (7)$$

This "accept-if-improves" rule enforces monotonic non-degradation at the batch level, making notebook evolution robust to occasional low-quality reflections. It also acts as an implicit regularizer that curbs uncontrolled notebook growth, since only useful notes survive. The pseudo-code of the overall procedure is shown in Algorithm 1, and all prompt templates are provided in Appendix A.

Algorithm 1: M^{2}Note evolving protocol

Input: Tuning Model π_θ; Tuner Model π_{tuner}; Data Corpus $\mathcal{D}$; Reward function $R(\cdot)$; RAG top-K and α
Output: Mistake Notebook $\mathcal{M}$

Initialize $\mathcal{M}_0 \leftarrow \emptyset$;
for *each* $\mathcal{B}_i = \{(q_k, i_k)\}_{k=1}^N$ *sampled from* $\mathcal{D}$ **do**
  // (1) Retrieve guidance and run baseline
  for *each* $k \in \mathcal{B}_i$ **do**
    $(s_k, g_k) \leftarrow \text{Ret}((q_k, i_k), \mathcal{M}_i; K, \alpha)$;
    $z_k = (q_k, i_k) \oplus (s_k, g_k)$;
  $\text{Acc}_{base} \leftarrow \frac{1}{N}\sum_{k=1}^N R(\pi_\theta(z_k|\mathcal{M}_i))$;
  // (2) Propose notebook edits from failures
  Backup memory $\mathcal{M}'_i \leftarrow \mathcal{M}_i$;
  for *each* k *such that* $R(\pi_\theta(z_k|\mathcal{M}_i) = 0$ **do**
    π_{tuner} generates a subject-guidance pair (s'_k, g'_k) and compares s_k and s'_k to make an action a_k ;
    if $a_k = \texttt{add}$ **then**
      $\mathcal{M}'_i \leftarrow \mathcal{M}'_i \cup (s'_k, g'_k, \mathcal{E}(s'_k \oplus g'_k))$
    else
      π_{tuner} merges $s_k \oplus s'_k$ into s^*_k and $g_k \oplus g'_k$ into g^*_k ;
      $\mathcal{M}'_i \leftarrow \mathcal{M}'_i \cup (s^*_k, g^*_k, \mathcal{E}(s^*_k \oplus g^*_k))$
  // (3) Batch-level post-verification with rollback
  Rerun the batch $\mathcal{B}_i$ with $\mathcal{M}'_i$ and compute $\text{Acc}_{new} \leftarrow \frac{1}{N}\sum_{k=1}^N R(\pi_\theta(z_k|\mathcal{M}'_i))$;
  $\mathcal{M}_{i+1} \leftarrow \mathcal{M}'_i$ **if** $\text{Acc}_{new} \geq \text{Acc}_{base}$ **else** roll back $\mathcal{M}_{i+1} = \mathcal{M}_i$;

4 Experiments

Experimental Settings. We adopt two experimental settings: ❶ *Supervised manner.* We hold out the test set for evaluation, and use data sampled from the training split to update and save the external mistake notebook, which is then used during evaluation. ❷ *Test-time scaling (TTS).* We directly evaluate on the test set while continuously updating the external memory online, allowing the model to improve progressively as testing proceeds.

Evaluation Datasets and Metrics. We evaluate on six benchmarks covering three vision language domains: STEM/Math (MMMU$_{val}$ (Yue et al., 2024), MathVista (Lu et al., 2023)), General VQA (MMStar (Chen et al., 2024), RealworldQA (xAI)), and Document OCR (AI2D (Kembhavi et al., 2016), ChartQA (Masry et al., 2022)). MMMU and MathVista emphasize multi-discipline reasoning and mathematical problem solving; MMStar and RealworldQA focus on open-world visual understanding; AI2D and ChartQA evaluate diagram/chart comprehension and text-grounded reasoning. Following prior work, we report accuracy (%) as the primary evaluation metric for all datasets. Among them, MMMU and MathVista are trained in a supervised manner, while the other benchmarks are evaluated using test-time scaling (TTS); please refer to Appendix C.2 for details.

Implementation Details. Our M^{2}Note does not require training model parameters; it only updates a mistake notebook stored as external memory in the

Table 1: **Main results.** We report accuracy and average score, together with the estimated training/API cost, the average number of entries in the memory (**Mem**), and the average number of tokens in the guidance (**Len**). ‡ denotes results reproduced by us with a specific answer-parsing strategy, and the best performance is in **bold.**

Method	Cost	Mem	Len	STEM Puzzle		General VQA		Document OCR		Avg.
				MMMU_{val}	MathVista	MMStar	RealworldQA	AI2D	ChartQA	
Self-Evolving: Compare with SFT- and RL-based methods										
Vision-Zero (Wang et al.)	Basically $> \$500$	–	–	58.8	72.6	65.2	68.5	84.5	86.3	72.65
EvoLMM (Thawakar et al.)		–	–	52.0	70.5	–	–	83.4	86.7	–
iReasoner (Sunil et al.)		–	–	52.4	69.7	–	–	83.9	85.8	–
VisPlay (He et al.)		–	–	54.9	68.2	65.1	69.0	–	86.2	–
DPE (Jia et al.)		–	–	69.1	76.2	62.1	72.1	–	84.8	–
Self-Evolving: Qwen3-VL-8B-Instruct (Open-source)										
Vanilla†	–	–	–	67.1	73.2	62.1	73.2	80.8	89.6	74.3
w/ **M²Note**	$\approx \$3$	50	261	**68.4**(↑1.3)	**77.3**(↑4.1)	**63.9**(↑1.8)	**75.4**(↑2.2)	**82.1**(↑1.3)	**89.9**(↑0.3)	**76.2**(↑1.9)
w/ CoT	–	–	–	68.8	82.9	62.9	78.0	81.3	91.7	77.6
w/ CoT+**M²Note**	$\approx \$4$	32	315	**69.4**(↑0.6)	**83.9**(↑1.0)	**64.9**(↑2.0)	**78.8**(↑0.8)	**81.7**(↑0.4)	**91.7**	**78.4**(↑0.8)
Self-Evolving: Qwen3-VL-Plus (Proprietary)										
Vanilla†	–	–	–	77.9	78.0	68.3	78.2	85.9	88.5	79.5
w/ **M²Note**	$\approx \$7$	28	462	**79.3**(↑1.4)	**80.0**(↑2.0)	**69.8**(↑1.5)	**79.4**(↑1.2)	**86.7**(↑0.8)	**89.2**(↑0.7)	**80.7**(↑1.2)
Self-Evolving: GPT-5.4 (Proprietary)										
Vanilla†	–	–	–	77.1	77.2	66.5	80.8	85.6	85.6	78.8
w/ **M²Note**	$\approx \$7$	28	462	**77.9**(↑0.8)	**78.0**(↑0.8)	**67.8**(↑1.3)	**81.8**(↑1.0)	**86.1**(↑0.5)	**85.8**(↑0.2)	**79.6**(↑0.8)
Cross-Model Evolving: Qwen3-VL-8B-Instruct (Tuning) and Qwen3-VL-Plus (Tuner)										
Vanilla†	–	–	–	67.1	73.2	62.1	73.2	80.8	89.6	74.3
w/ **M²Note**	$\approx \$5$	43	445	**68.9**(↑1.8)	**78.3**(↑5.1)	**64.0**(↑1.9)	**77.1**(↑3.9)	**81.0**(↑0.2)	**90.1**(↑0.5)	**76.6**(↑2.3)

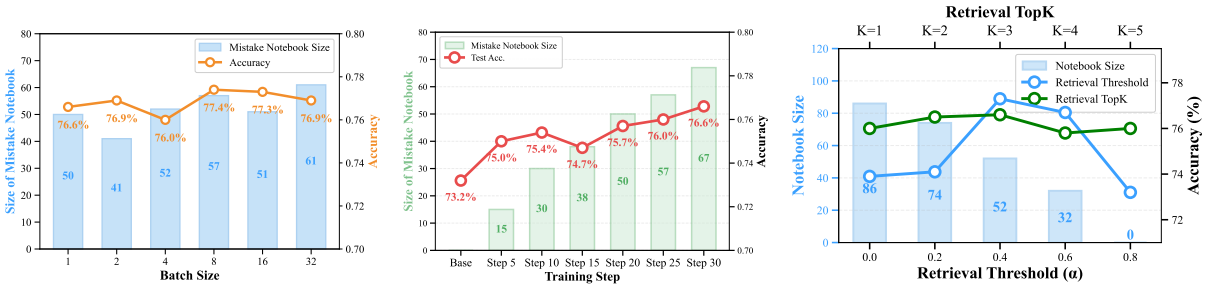
JSONL format. We evaluate three VLMs: the open-source *Qwen3-VL-8B-Instruct* (Bai et al., 2025a), the closed-source *Qwen3-VL-Plus* (Bai et al., 2025a) and *GPT-5.4* (Hurst et al., 2024). For multimodal RAG, we use *Qwen3-VL-Embedding* (Li et al., 2026) as the embedding model. In addition to these models, all ablation variants are also implemented through API requests. Unless otherwise specified, we set the RAG top-K to $K = 1$ and the RAG threshold to $\alpha = 0.4$. The batch size for both training and inference is 16. For supervised experiments on MMMU and MathVista, we run 10 and 20 training steps, respectively. More details can be found in Appendix C.1.

4.1 Main Results

Table 1 reports the main results of M²Note on six multimodal reasoning benchmarks: ❶ *Consistent gains across domains and backbones.* M²Note yields consistent improvements on different domains, and the gains hold for both an open-source backbone and a stronger proprietary backbone. ❷ *Cost- and sample-efficiency.* Compared to RL-based self-evolving approaches, M²Note achieves competitive or even better performance with dramatically lower cost via pure API requests. Under the same backbone model (*Qwen3-VL-8B-Instruct*), M²Note matches or slightly outperforms DPE (Jia et al., 2026) while avoiding RL-style training overhead. ❸ *Multiple evolving modes.* M²Note supports both self-evolving (single model acts as both the tuning and tuner models) and cross-model evolving (a stronger tuner model supervises a weaker tuning model), enabling capability transfer without any weight updates. ❹ *CoT compatibility.* M²Note can work together with Chain-of-Thought (CoT) prompting (Brown et al., 2020; Wei et al., 2022b), leading to further gains over either CoT or M²Note alone. ❺ *Compact notebook with interpretable scaling trends.* We observe that stronger backbones make fewer mistakes and require fewer notebook entries (**Mem**), while a stronger tuner (*Qwen3-VL-Plus*) tends to produce more detailed guidance, resulting in longer length (**Len**).

4.2 Ablation Study

Sensitivity Analysis of Hyperparameters. Results in Figure 3a and 3b reveal two trends: ❶ M²Note is relatively insensitive to batch size. Across a wide range (1–32), test accuracy remains stable at around 76%–77.4%, suggesting that the closed-loop update does not depend heavily on a specific batch size. Moderate batch sizes (e.g., 8 or 16) perform best, likely because they support more robust batch-level memory refinement and make the *verify* stage more reliable. Meanwhile, note-



(a) Batch size sensitivity. (b) Evolving as training progresses. (c) Ablation study of the RAG settings.

Figure 3: **Effects of different hyperparameters on performance and notebook size.** Experiments are conducted on MathVista with *Qwen3-VL-8B-Instruct*.

book size does not increase monotonically with batch size (about 41–61 entries), indicating that subject-level merging prevents memory from scaling directly with the batch. ❷ As training progresses, the notebook gradually expands (15 to 67 entries), while accuracy improves from 73.2% to 76.6%. Although there is some fluctuation around step 15, the overall upward trend suggests that continued evolution accumulates reusable guidance and improves test generalization.

Multimodal RAG Settings. Figure 3c studies the effect of the retrieval threshold α and the retrieval top-K on MathVista (*Qwen3-VL-8B-Instruct*). Increasing α makes the system more conservative, reducing the number of queries that trigger note insertion; as fewer queries retrieve any guidance, batch performance remains unchanged before and after the update, thus shrinking the final notebook size. In terms of accuracy, a moderate threshold performs best: α=0.4 achieves the highest accuracy, while overly strict retrieval (e.g., α=0.8) disables the notebook and degrades performance. We also observe that the system is robust to the retrieval top-K, with performance varying only slightly. Overall, we prefer a smaller value (K=1 by default) to reduce the risk of hallucinations.

Effectiveness of Batch-Level Post-Verification. Table 2 examines the batch-level post-verification, namely *accept-if-improves*. For *Qwen3-VL-8B-Instruct*, enabling post verification yields the best overall results, notably improving MathVista (73.2 → 77.3) and MMMU$_{val}$ (67.1 → 68.4), showing that filtering noisy updates stabilizes evolution and makes the accumulated notes more reliable. Without post verification, performance becomes less consistent (e.g., a drop on MMMU$_{val}$), indicating that blindly committing proposed notes may introduce harmful or noisy guidance. For the stronger

Table 2: **Effectiveness of batch-level post-verification.** We report results on three benchmarks across two backbones. Best results are marked in **bold**.

Method	MathVista	MMMU$_{val}$	MMStar
Self-Evolving: Qwen3-VL-8B-Instruct (Open-source)			
Vanilla	73.2	67.1	62.1
w/o Batch-Level Post-Verification	76.2	65.7	62.7
w/ Batch-Level Post-Verification	**77.3**	**68.4**	**63.9**
Self-Evolving: Qwen3-VL-Plus (Proprietary)			
Vanilla	78.0	77.9	68.3
w/o Batch-Level Post-Verification	**80.4**	77.0	**69.6**
w/ Batch-Level Post-Verification	80.1	**79.7**	69.4

Qwen3-VL-Plus backbone, the effect is smaller but still positive on average, suggesting that as the base model makes fewer mistakes, the notebook receives fewer high-impact updates, yet conservative verification remains useful for preventing degradation.

Ablation on retrieval embeddings. We study how the embedding model and input modality affect M^{2}Note as shown in Table 3. Using a text-only embedding model (*Qwen3-Embedding*) can improve MathVista, but tends to generalize worse on other benchmarks such as MMStar. In contrast, VLM-based embedding models (*Qwen3/2.5-VL-Embedding*) that encode both image and text consistently achieve better performance, suggesting that multimodal indexing better captures the visual cues needed to retrieve relevant notes. This is further supported by the modality ablation: using a single modality, either image or text, is inferior to using both, indicating that visual and textual signals are complementary for robust note retrieval.

Notebook transfer across models. We evaluate whether an evolved mistake notebook can transfer across different VLMs (Table 4). Specifically, we first self-evolve a notebook with one backbone (e.g., *Qwen3-VL-8B-Instruct* or *Qwen3-VL-Plus*), then freeze it and use it as external guidance for a

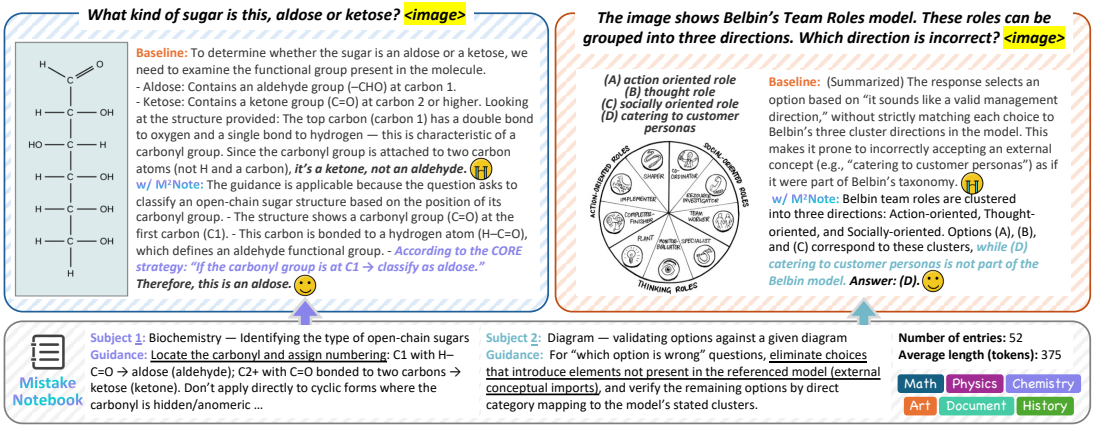


Figure 4: **Qualitative results.** For a STEM (biochemistry) question and a document understanding question, the baseline tends to rely on superficial cues and makes incorrect selections, while M^{2}Note retrieves subject-specific guidance from the external mistake notebook and corrects the reasoning by enforcing key structural checks (e.g., carbonyl position for aldose/ketose; option-to-cluster mapping for Belbin roles).

Table 3: **Impact of different embedding models and retrieval modes.** We report results using *Qwen3-VL-Instruct-8B*. Best results are marked in **bold**.

Embedding Model	Image	Text	MathVista	MMMU$_{val}$	MMStar
Self-Evolving: Qwen3-VL-8B-Instruct (Open-source)					
Vanilla	-	-	73.2	67.1	62.1
Qwen3-Embedding (Zhang et al.)	-	✓	75.5	66.6	59.1
Qwen2.5-VL-Embedding (Li et al.)	✓	✓	**77.3**	66.2	63.2
Qwen3-VL-Embedding (Li et al.)	✓	✗	76.3	66.0	63.0
Qwen3-VL-Embedding (Li et al.)	✗	✓	77.1	66.2	62.3
Qwen3-VL-Embedding (Li et al.)	✓	✓	**77.3**	**68.4**	**63.9**

Table 4: **Notebook transfer across different VLMs.** We report results using three different VLMs.

Tuning Model	Mem	Len	MathVista	MMMU$_{val}$	MMStar
Notebook obtained via self-evolving Qwen3-VL-8B-Instruct					
Qwen3-VL-Plus (Bai et al.)	–	–	78.0	77.9	68.3
w/ Specific Notebook	61	261	78.7(↑0.7)	76.8(↓1.1)	68.6(↑0.3)
Qwen3-VL-32B-Instruct (Bai et al.)	–	–	81.1	73.1	67.8
w/ Specific Notebook	61	261	81.4(↑0.4)	74.1(↑1.0)	68.1(↑0.3)
Notebook obtained via self-evolving Qwen3-VL-Plus					
Qwen3-VL-8B-Instruct (Bai et al.)	–	–	73.2	67.1	62.1
w/ Specific Notebook	33	469	74.7(↑1.5)	67.4(↑0.3)	62.7(↑0.6)

different tuning model, without further evolution or parameter updates. Overall, the transferred notebooks remain effective, indicating that M^{2}Note captures partially model-agnostic and reusable guidance. However, cross-model transfer is not always risk-free, and mismatched guidance can hurt performance in some cases. Moreover, compared with self-evolved notebooks in Table 1, transferred notebooks generally yield smaller gains, suggesting that the strongest improvements come from adapting to each model's own failure patterns.

4.3 Qualitative Results

Fig. 4 presents two representative cases illustrating how M^{2}Note improves multimodal reasoning. In the STEM (biochemistry) example on the left, the baseline gives a plausible explanation but misidentifies key structural cues, leading to an incorrect classification. With M^{2}Note, the model retrieves subject-level guidance from the external mistake notebook (e.g., locate the carbonyl and verify its position), which enforces essential structural checks and corrects the conclusion. In the diagram understanding example on the right, the baseline tends to pick an option that sounds reasonable without strictly aligning choices to the diagram's defined taxonomy, making it vulnerable to introducing hallucinations. M^{2}Note retrieves guidance that explicitly requires option-to-cluster mapping and eliminates elements not present in the reference model, yielding the correct answer. More visualizations, including case analyses and mistake-notebook visualizations, are provided in Appendix B (Figure 5).

5 Conclusion

We introduce M^{2}Note, a training-free framework for continually improving VLMs through multimodal mistake notebook learning. Instead of updating model weights, M^{2}Note distills model failures into reusable subject-specific guidance and retrieves relevant notes at inference time via multimodal RAG, helping the model verify its reasoning and avoid repeated errors. A key mechanism is batch-level *accept-if-improves* verification, which filters noisy notebook updates and stabilizes contin-

ual evolution by retaining only beneficial changes. Across six benchmarks covering STEM/math reasoning, general VQA, and document understanding, M^{2}Note yields consistent gains in both *self-evolving* and *cross-model evolving* settings, while also combining effectively with CoT prompting. Overall, these results suggest that reusing verified failure cases provides a practical and effective way to enhance VLM robustness at deployment.

6 Limitation

M^{2}Note evolves in a way that resembles how humans learn from experience: instead of rewriting one's "brain" (model weights), it repeatedly summarizes failures into reusable rules and consults them when facing similar situations. This mechanism is most effective when mistakes exhibit recurring structure, so that a distilled note can be reliably reused. As a result, M^{2}Note tends to work best in relatively narrow domains (e.g., math reasoning), where tasks share stable abstractions and retrieved guidance is more likely to transfer. In broad domains with long-tail visual diversity, relevant notes can be harder to retrieve, and mismatched guidance may introduce misleading context or amplify hallucinations, making continual evolution less stable. These limitations suggest two directions: ❶ building more generalizable notebooks through stronger abstraction and a more reliable post-update verification strategy, so that notes transfer beyond near-duplicate failures and harmful guidance is less likely to be retrieved; and ❷ enriching notes beyond text (e.g., reference images, tool-use traces, structured checklists) to provide more grounded and actionable guidance.

References

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. 2025a. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*.

Sule Bai, Mingxing Li, Yong Liu, Jing Tang, Haoji Zhang, Lei Sun, Xiangxiang Chu, and Yansong Tang. 2025b. Univg-r1: Reasoning guided universal visual grounding with reinforcement learning. *arXiv preprint arXiv:2505.14231*.

Florian Bordes, Richard Yuanzhe Pang, Anurag Ajay, et al. 2024. An introduction to vision-language modeling. *Preprint*, arXiv:2405.17247.

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

Yuzheng Cai, Siqi Cai, Yuchen Shi, Zihan Xu, Lichao Chen, Yulei Qin, Xiaoyu Tan, Gang Li, Zongyi Li, Haojia Lin, et al. 2025. Training-free group relative policy optimization. *arXiv preprint arXiv:2510.08191*.

Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. 2023. Fireact: Toward language agent fine-tuning. *arXiv preprint arXiv:2310.05915*.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, et al. 2024. Are we on the right way for evaluating large vision-language models? *Advances in Neural Information Processing Systems*, 37:27056–27087.

Rui Chen, Lei Sun, Jing Tang, Geng Li, and Xiangxiang Chu. 2025. Finger: Content aware fine-grained evaluation with reasoning for ai-generated videos. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 3517–3526.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Yanqi Dai, Yuxiang Ji, Xiao Zhang, Yong Wang, Xiangxiang Chu, and Zhiwu Lu. 2026. Harder is better: Boosting mathematical reasoning via difficulty-aware grpo and multi-aspect question reformulation. *arXiv preprint arXiv:2601.20614*.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, et al. 2024. A survey on llm-as-a-judge. *The Innovation*.

Dong Guo, Faming Wu, Feida Zhu, Fuxing Leng, Guang Shi, Haobin Chen, Haoqi Fan, Jian Wang, Jianyu Jiang, Jiawei Wang, et al. 2025. Seed1. 5-vl technical report. *arXiv preprint arXiv:2505.07062*.

Yicheng He, Chengsong Huang, Zongxia Li, Jiaxin Huang, and Yonghui Yang. 2025. Visplay: Self-evolving vision-language models from images. *Preprint*, arXiv:2511.15661.

Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. 2025. R-zero: Self-evolving reasoning llm from zero data. *arXiv preprint arXiv:2508.05004*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Yuxiang Ji, Ziyu Ma, Yong Wang, Guanhua Chen, Xiangxiang Chu, and Liaoni Wu. 2025. Tree search for llm agent reinforcement learning. *arXiv preprint arXiv:2509.21240*.

Hongrui Jia, Chaoya Jiang, Shikun Zhang, and Wei Ye. 2026. From blind spots to gains: Diagnostic-driven iterative training for large multimodal models. *Preprint*, arXiv:2602.22859.

Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. 2016. A diagram is worth a dozen images. In *European conference on computer vision*, pages 235–251. Springer.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. 2024. Mvbench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22195–22206.

Mingxin Li, Yanzhao Zhang, Dingkun Long, Keqin Chen, Sibo Song, Shuai Bai, Zhibo Yang, Pengjun Xie, An Yang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2026. Qwen3-vl-embedding and qwen3-vl-reranker: A unified framework for state-of-the-art multimodal retrieval and ranking. *Preprint*, arXiv:2601.04720.

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

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

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

Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xu-Cheng Yin, Cheng-Lin Liu, Lianwen Jin, and Xiang Bai. 2024c. Ocrbench: on the hidden mystery of ocr in large multimodal models. *Science China Information Sciences*, 67(12):220102.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. 2023. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594.

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

Lang Mei, Siyu Mo, Zhihan Yang, and Chong Chen. 2025. A survey of multimodal retrieval-augmented generation. *arXiv preprint arXiv:2504.08748*.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. 2025. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 20286–20332.

Siru Ouyang, Jun Yan, I Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T Le, Samira Daruki, Xiangru Tang, et al. 2025. Reasoningbank: Scaling agent self-evolving with reasoning memory. *arXiv preprint arXiv:2509.25140*.

Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. 2023. Automatic prompt optimization with "gradient descent" and beam search. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 7957–7968.

Runqi Qiao, Qiuna Tan, Guanting Dong, MinhuiWu MinhuiWu, Chong Sun, Xiaoshuai Song, Jiapeng Wang, Zhuoma Gongque, Shanglin Lei, Yifan Zhang, et al. 2025. We-math: Does your large multimodal model achieve human-like mathematical reasoning? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 20023–20070.

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

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *Preprint*, arXiv:1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652.

Xuanbo Su, Yingfang Zhang, Hao Luo, Xiaoteng Liu, and Leo Huang. 2025. Mistake notebook learning: Batch-clustered failures for training-free agent adaptation. *arXiv preprint arXiv:2512.11485*.

Nan Sun, Jing Tang, Lei Sun, Rui Chen, Yuxing Lu, Xiangxiang Chu, Hefei Ling, and Yujun Cai. 2026. Beyond query bias: Candidate-aware iterative refinement for zero-shot composed image retrieval. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 22318–22329.

Meghana Sunil, Manikandarajan Venmathimaran, and Muthu Subash Kavitha. 2026. ireasoner: Trajectory-aware intrinsic reasoning supervision for self-evolving large multimodal models. *Preprint*, arXiv:2601.05877.

Omkar Thawakar, Shravan Venkatraman, Ritesh Thawkar, Abdelrahman Shaker, Hisham Cholakkal, Rao Muhammad Anwer, Salman Khan, and Fahad Khan. 2025. Evolmm: Self-evolving large multimodal models with continuous rewards. *Preprint*, arXiv:2511.16672.

Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. 2024. Measuring multimodal mathematical reasoning with math-vision dataset. *Advances in Neural Information Processing Systems*, 37:95095–95169.

Qinsi Wang, Bo Liu, Tianyi Zhou, Jing Shi, Yueqian Lin, Yiran Chen, Hai Helen Li, Kun Wan, and Wentian Zhao. 2025a. Vision-zero: Scalable vlm self-improvement via strategic gamified self-play. *Preprint*, arXiv:2509.25541.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. 2025b. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. 2025c. Agent workflow memory. In *International Conference on Machine Learning*, pages 63897–63911. PMLR.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022a. Finetuned language models are zero-shot learners. *Preprint*, arXiv:2109.01652.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022b. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

xAI. Realworldqa: A benchmark for real-world spatial understanding. https://huggingface.co/datasets/xai-org/RealworldQA. Accessed: 2025-04-26.

Chaoran Xu, Yingmao Miao, Pengfei Zhang, Hao Dou, Lei Sun, and Xiangxiang Chu. 2026. Rise: Reliable improvement in self-evolving vision-language models. *arXiv preprint arXiv:2605.20914*.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. 2023. Large language models as optimizers. In *The Twelfth International Conference on Learning Representations*.

Zhibo Yang, Jun Tang, Zhaohai Li, Pengfei Wang, Jianqiang Wan, Humen Zhong, Xuejing Liu, Mingkun Yang, Peng Wang, Shuai Bai, et al. 2025. Cc-ocr: A comprehensive and challenging ocr benchmark for evaluating large multimodal models in literacy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21744–21754.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, and 16 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *Preprint*, arXiv:2503.14476.

Zhenlong Yuan, Xiangyan Qu, Chengxuan Qian, Rui Chen, Jing Tang, Lei Sun, Xiangxiang Chu, Dapeng Zhang, Yiwei Wang, Yujun Cai, and Shuo Li. 2026. Video-STAR: Reinforcing open-vocabulary action recognition with tools. In *The Fourteenth International Conference on Learning Representations*.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. 2024. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9556–9567.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. 2024. Agenttuning: Enabling generalized agent abilities for llms. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3053–3077.

Yunpeng Zhai, Shuchang Tao, Cheng Chen, Anni Zou, Ziqian Chen, Qingxu Fu, Shinji Mai, Li Yu, Jiaji Deng, Zouying Cao, et al. 2025. Agentevolver: Towards efficient self-evolving agent system. *arXiv preprint arXiv:2511.10395*.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, et al. 2025a. Agentic context engineering: Evolving contexts for self-improving language models. *arXiv preprint arXiv:2510.04618*.

Tianjun Zhang, Aman Madaan, Luyu Gao, Steven Zheng, Swaroop Mishra, Yiming Yang, Niket Tandon, and Uri Alon. 2024a. In-context principle learning from mistakes. In *Proceedings of the 41st International Conference on Machine Learning*, pages 59520–59558.

Wenhao Zhang, Yuexiang Xie, Yuchang Sun, Yanxi Chen, Guoyin Wang, Yaliang Li, Bolin Ding, and Jingren Zhou. 2026. On-policy rl meets off-policy experts: Harmonizing supervised fine-tuning and reinforcement learning via dynamic weighting. *Preprint*, arXiv:2508.11408.

Xin Zhang, Yanzhao Zhang, Wen Xie, Mingxin Li, Ziqi Dai, Dingkun Long, Pengjun Xie, Meishan Zhang, Wenjie Li, and Min Zhang. 2024b. Gme: improving universal multimodal retrieval by multimodal llms. *arXiv preprint arXiv:2412.16855*.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025b. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *Preprint*, arXiv:2506.05176.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. 2024. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19632–19642.

Yilun Zhao, Haowei Zhang, Lujing Xie, Tongyan Hu, Guo Gan, Yitao Long, Zhiyuan Hu, Weiyuan Chen, Chuhan Li, Zhijian Xu, et al. 2025. Mmvu: Measuring expert-level multi-discipline video understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 8475–8489.

Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. Synapse: Trajectory-as-exemplar prompting with memory for computer control. In *The Twelfth International Conference on Learning Representations*.

Huichi Zhou, Yihang Chen, Siyuan Guo, Xue Yan, Kin Hei Lee, Zihan Wang, Ka Yiu Lee, Guchun Zhang, Kun Shao, Linyi Yang, et al. 2025. Memento: Fine-tuning llm agents without fine-tuning llms. *arXiv preprint arXiv:2508.16153*.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. Large language models are human-level prompt engineers. In *The eleventh international conference on learning representations*.

A Prompts Used in the M^{2}Note Framework

A.1 Tuner Model Prompts

Guidance Extraction

You are a Cognitive Strategy Expert. Your goal is to induce **generalizable rules** from specific model errors, not just correct them.

Subject: {subject}
Error Data: {error_context}

Task: Analyze the underlying logical flaws in the Error Data and formulate a reusable problem-solving heuristic.

Constraints (Total $\leq$ 350 words):
1. **Abstracted Error Pattern** ($\leq$ 60 words)
- Describe the *type* of situation where this error occurs (strip specific numbers/names).
- Do NOT simply rewrite the original questions.
2. **Root Cause Analysis** ($\leq$ 50 words)
- Identify the cognitive gap (e.g., confusing correlation with causation, ignoring boundary conditions).
3. **Generalizable Strategy (CORE)** ($\leq$ 120 words)
- Provide a step-by-step heuristic or checklist applicable to ANY similar future problem.
- Use "If [condition], Then [action]" format where possible.
- Must work beyond the provided examples.
4. **Anti-Patterns & Boundary** ($\leq$ 80 words)
- List 1-2 specific scenarios where this strategy should NOT be applied.
- List 1-2 common misinterpretations of this guidance.

Output Style:
- High-level, structured, knowledge-focused, and reusable for similar future questions.
- Avoid referencing specific details from the provided Error Data unless illustrating a pattern.

Subject Classification

You are an expert in categorizing questions into precise, high-relevance subjects for Retrieval-Augmented Generation (RAG).

Your goal is to assign each question a subject label (about 10-50 words) that:
- Maximizes retrieval relevance by precisely describing the problem type and solution method.
- Groups only genuinely similar questions together (same domain AND same approach).
- Avoids over-broad categories that would match unrelated problems.
- Reuses the same subject name for closely related questions.

CRITICAL: The subject must be specific enough to prevent irrelevant retrieval. Include, when applicable:
1. Primary Domain (e.g., Combinatorics, Complex Analysis, Linear Algebra, Physics, Programming, Document Understanding)
2. Problem Type (e.g., counting with constraints, roots of unity products, debugging API parameters, OCR table extraction)
3. Solution Method (e.g., stars and bars, polynomial/root identities, Hensel's lemma, reproduce-minimize-fix)

Examples of GOOD subjects (specific):
✓Complex Analysis: Evaluating products over roots of unity using polynomial evaluation and complex identities
✓Document Understanding: Extracting tables from scanned PDFs using layout detection + OCR + row/column reconstruction

Examples of BAD subjects (too broad):
✗number theory (too broad - could match any modulo problem)
✗physics (too broad - could match any physics problem)

Key principle: If a problem spans multiple domains, label it by the primary one.

Output only the finalized subject label(s).

Guidence Merge

Synthesize guidance for subject: {subject}
Existing guidance from related subjects: {existing_guidance}
New guidance: {new_guidance}

Merge into a single coherent guidance (max 2048 chars) that:
- Combines insights from related subjects with new guidance
- Eliminates redundancy while preserving key information and examples of the mistakes
- **Preserves and emphasizes applicability conditions**
- clearly state when each method applies
- Focuses on actionable advice
- Maintains consistent style
- **Includes warnings about when NOT to apply the guidance** to avoid misapplication

Merged guidance:

Subject Merge

The following subjects are related: {subjects}
Provide a single, concise subject name ($\leq$10 words) that best represents all of these. Only output the subject name, nothing else.

A.2 Tuning Model Prompts

User Prompt

Question: {question}
Options: {options}
Please select the correct answer from the options above.

RAG-Enhanced System Prompt

The following mistake notes are not necessarily tied to the current question, but you may use them to deepen your analytical approach

IMPORTANT: Before applying any guidance below, carefully evaluate:
1. Does the current problem match the applicability conditions stated in the guidance?
2. Is the problem type and context similar to the examples in the guidance?
3. If the problem is totally different (e.g., combinatorics vs modulo arithmetic, complex numbers vs number theory), do NOT force-fit the guidance.
4. Only use guidance that is clearly relevant to the current problem.

Gudience Subject: {subject} - {guidance}
Before solving, review the guidance. State whether it is: applicable, partially applicable, or irrelevant. Use only applicable parts.

Since the guidance introduced via multimodal RAG (Lewis et al., 2020; Mei et al., 2025) may not necessarily apply to the current question, in addition to the top-K and threshold settings in the RAG configuration, we further emphasize in the system prompt that the model should judge whether the retrieved content is applicable to the current question. This is crucial for reducing hallucinations. As shown in Fig. 5, several cases are presented where the model's response explicitly includes this judgment: the first two cases are applicable, while the last one illustrates an inapplicable scenario. Table 5 compares blindly following the guidance (naive) with introducing a judgment mechanism (judgement); the latter effectively improves the model's problem-solving accuracy.

B More Visualizations

Below we present several entries from the mistake notebook, obtained by training *Qwen3-VL-8B-Instruct* as the **Tuning Model** on MMMU (Yue et al., 2024). As shown, the structured guidance is produced using the **Tuner Model** with the "Guidance Extraction" prompt, while the **Note** and the **Task** field are derived using the "Subject Classification" prompt.

Table 5: **Effect of applicability judgment for retrieved guidance**. Results on three benchmarks and two backbones. *Naive* uses retrieved guidance directly, while *Judgment* verifies relevance first, consistently improving performance. Best results are in **bold**.

Method	MathVista	$MMMU_{val}$	MMStar
Self-Evolving: Qwen3-VL-8B-Instruct (Open-source)			
Vanilla	73.2	67.1	62.1
w/ M^2Note (Naive)	76.2	67.8	**64.1**
w/ M^2Note (Judgment)	**77.3**	**68.4**	63.9
Self-Evolving: Qwen3-VL-Plus (Proprietary)			
Vanilla	78.0	77.9	68.3
w/ M^2Note (Naive)	79.5	76.9	69.0
w/ M^2Note (Judgment)	**80.1**	**79.7**	**69.4**

Note 1: Document Understanding

Task Extract structured brainstorming workflow steps from a quadrant diagram (visual category mapping + label extraction).
Abstracted Error Pattern Options that sound reasonable are mistaken as valid steps even when they contradict the diagram's collaborative workflow (e.g., silent reading, during disagreement).
Root Cause Plausibility is prioritized over fidelity to the visual framework; behavioral/structural constraints are ignored.
Generalizable Strategy (CORE)

- Reject options introducing elements absent from the visual model (silence, conflict resolution, external tools).
- Reject behaviors inconsistent with the interaction pattern (solo reading vs group discussion).
- If ambiguous, validate against core stages: categorization → discussion → selection.
- Prefer behavioral alignment over fluent wording.

Boundary Not suitable when silent reflection is explicitly part of the process, or when the model allows multiple valid interpretations.

Note 2: Medical Imaging

Task Identify anatomical landmarks in fundus photographs (visual cue detection + retinal structure recognition).
Abstracted Error Pattern Structural sparsity is overgeneralized into complete absence across a region without checking the exact boundary/layer.
Root Cause Confusing minimal presence with absence; weak boundary checking for layered anatomy and region-specific exceptions.
Generalizable Strategy (CORE)

- Verify absence at the exact indicated location (not nearby areas).
- Check layers independently: photoreceptors → bipolar → ganglion → blood vessels.
- Choose "all of the above" only if every listed condition holds.
- Cross-reference standard anatomy for region-specific exceptions.

Boundary Not for dynamic processes (disease progression) or non-structural properties (function).

Note 3: Chemistry

Task Determine the maximum number of saccharin molecules from an atomic inventory (limiting-reactant style stoichiometry).
Abstracted Error Pattern Users misidentify the limiting element by ignoring stoichiometric ratios or miscounting atoms in the molecular formula.
Root Cause Atom availability is mistaken for molecule yield; constraints are not computed per element to find the true bottleneck.
Generalizable Strategy (CORE)

1. Derive the exact molecular formula (count all atoms).
2. For each element: available atoms $\div$ required per molecule $\Rightarrow$ max molecules.
3. The smallest quotient determines the maximum yield (the limiting element).
4. Sanity-check: if quotients look inconsistent, recheck formula/inventory.
5. Confirm the final answer satisfies the limiting element's requirement exactly.

Boundary Not for non-stoichiometric settings (catalysis, equilibrium-limited yield) or inventories with impurities/intermediates not modeled.

Note 4: Education

Task Identify a classroom activity for literary analysis via role-play and multi-perspective retelling (media simulation).
Abstracted Error Pattern The activity is misidentified by matching the image's media appearance (e.g., a news set) rather than its pedagogical function.
Root Cause Surface-level visual similarity overrides functional interpretation (real media format vs simulated classroom role-play).
Generalizable Strategy (CORE)

- Eliminate options that are literal media formats (e.g., News Program).
- Prefer terms describing pedagogical function: interviewing, role assumption, perspective simulation.
- Check definitions for student agency, dialogue, and interpretive framing.
- Treat broadcast-studio visuals as distractors; focus on the learning objective.
- Differentiate close terms: Hot Seat (structured questioning) vs Readers Theatre (scripted reading).

Boundary Not for purely observational tasks (watching media) or when the explicit goal is media production.

Note 5: Document Understanding

Task Count pathogens on leaf images (visual segmentation + pattern recognition).
Abstracted Error Pattern Multiple symptom patterns are incorrectly treated as multiple pathogens, ignoring overlap and single-agent variability.
Root Cause Symptoms (markers) are conflated with causal entities; overlap/co-infection and environmental confounds are not checked.
Generalizable Strategy (CORE)

- Check whether different patterns overlap spatially/temporally; overlap may indicate one agent.
- Compare with known symptom profiles to validate distinct causes.
- If distinctness is supported, count independent infection zones (not symptom types).
- Consider context (treatment, stressors) that can mimic/mask infections.
- If evidence is weak, prefer "unknown" over forcing a precise count.

Boundary Not when symptoms clearly indicate unrelated agents (e.g., fungal + viral), or when the question asks for types rather than counts.

Note 6: Combinatorics

Task Schedule exams under conflict constraints using graph coloring (minimum number of time slots).
Abstracted Error Pattern Minimum slots are underestimated by ignoring constraint propagation or assuming a locally good coloring is globally feasible.
Root Cause Chromatic number is conflated with clique size; indirect (non-local) conflicts are not fully enforced/validated.
Generalizable Strategy (CORE)

1. Build a conflict graph: exams = vertices; edges = shared students/resources.
2. Color the graph (greedy or exact), enforcing all pairwise conflicts.
3. Validate the schedule: no student may have two exams in the same slot/day.
4. If violations occur, re-solve using backtracking/constraint propagation.
5. Re-check feasibility against real-world rules (concurrency constraints).

Boundary Not for dynamic/time-dependent constraints beyond a static conflict graph, or when intra-day sequencing/rooms must be modeled.

C Experimental Settings

C.1 Implementation Details

Our supervised training settings on MMMU (Yue et al., 2024) and MathVista (Lu et al., 2023) are as follows: ❶ On MMMU, we train on the official *dev* split (150 STEM questions) to obtain a tuned memory, which is then used as an external notebook during inference on the MMMU *validation* split, where we report accuracy. ❷ On MathVista, we randomly sample 320 examples from the *test* split to train the notebook, and evaluate on the *text-mini* split, following common practice in prior work.

For the other benchmarks (MMStar (Chen et al., 2024), AI2D (Kembhavi et al., 2016), RealworldQA (xAI), and ChartQA (Masry et al., 2022)), we adopt a test-time scaling (TTS) (Muennighoff et al., 2025) setup, where the memory is updated online during evaluation to progressively enhance the model's capability.

C.2 Benchmark Details

For comprehensive evaluation across diverse visual modalities, reasoning skills, and knowledge domains, we conduct experiments on six widely-used multimodal benchmarks: MMMU (Yue et al., 2024), MathVista (Lu et al., 2023), MMStar (Chen et al., 2024), RealworldQA (xAI), AI2D (Kembhavi et al., 2016), and ChartQA (Masry et al., 2022). Together, these benchmarks cover broad subject understanding, visual mathematical reasoning, real-world recognition and commonsense, diagram interpretation, and chart comprehension.

MMMU (Yue et al., 2024) is a large-scale benchmark designed to measure multimodal understanding across a wide range of academic disciplines and professional fields. Questions typically require jointly interpreting images (e.g., figures, tables, diagrams, screenshots) and text, and then performing domain-specific reasoning. These questions span 30 subjects and 183 subfields, comprising 30 highly heterogeneous image types, such as charts, diagrams, maps, tables, music sheets, and chemical structures. MMMU is challenging due to its breadth of subjects and its emphasis on knowledge-intensive and reasoning-intensive problems rather than purely perceptual recognition.

MathVista (Lu et al., 2023) focuses on visual mathematical reasoning, where models must solve math problems grounded in visual context. Inputs often contain charts, diagrams, geometry figures, or real-world scenes with quantitative cues. The benchmark evaluates capabilities such as extracting numerical information from images, mapping visual elements to formal quantities, performing multi-step reasoning, and producing the final mathematical answer. It is well-suited for analyzing error patterns related to perception-to-symbol grounding and step-by-step quantitative reasoning.

MMStar (Chen et al., 2024) is a general multimodal evaluation benchmark aimed at assessing robust vision-language understanding across a variety of everyday and document-style visual inputs. Questions span recognition, attribute reasoning, spatial reasoning, and higher-level comprehension. Compared with subject-focused benchmarks, MMStar is commonly used to test overall multimodal competence and generalization, making it useful for verifying whether our method improves broad visual-text reasoning rather than overfitting to a narrow domain.

RealworldQA (xAI) targets question answering in real-world visual scenarios, emphasizing practical knowledge and reasoning grounded in natural images. Questions often rely on recognizing objects, scenes, affordances, and context, and may require commonsense or everyday knowledge beyond simple identification. This benchmark is helpful for evaluating whether the model's "mistake-driven" updates improve robustness in realistic settings where visual ambiguity and implicit assumptions are common.

AI2D (Kembhavi et al., 2016) is a benchmark for science diagram understanding. It contains elementary and middle-school style diagrams (e.g., life cycles, physics illustrations, anatomy/biology schematics) paired with questions that require interpreting labeled components, arrows, processes, and relationships. AI2D is particularly challenging because correct answers often depend on understanding diagram structure and semantics, not just recognizing visual entities.

ChartQA (Masry et al., 2022) evaluates chart and plot understanding, including bar charts, line charts, pie charts, and other visualizations commonly seen in reports. Questions require reading values from axes/legends, comparing trends, performing simple arithmetic, or synthesizing information across multiple chart elements. ChartQA is valuable for analyzing errors related to visual-text alignment (e.g., legend-to-series mapping), numerical extraction, and compositional reasoning over structured visual data.

Question: *Using superposition and Laplace transform techniques, calculate the voltage $v_2(t)$ in the circuit.<image>*

Answer: *(D)* $v_2(t) = [2 - 2.4e^{-t/2} - (1/\sqrt{5})sin(t - 63.44°)]u(t)$

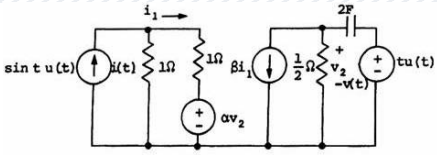


Fig. 1 α = 1, β = 2

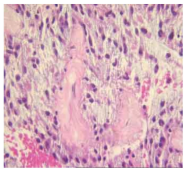
Updated System Prompt (injected subject & guidance):

• Topic: Solving linear circuits (superposition + Laplace) with dependent sources.
• Core rule: During superposition, do not deactivate dependent sources; treat them as active constraints. If a dependent source's output equals a node voltage, explicitly substitute it into KCL/KVL and solve consistently.

Updated Response (key excerpt to visualize):

• Applicability decision: The model ultimately concludes the guidance **is applicable** because the circuit includes dependent sources (notably a VCVS proportional to v_2, which must remain active under superposition).
• Outcome: Selects Option (D), matching the standard answer.

Question: *The lesion depicted here. <image>*



Ground-truth answer: *(E): a, b and c are correct*

Updated System Prompt (injected subject & guidance):

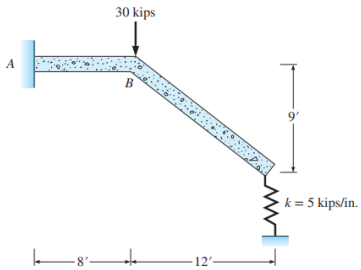
• Topic: Histopathology image interpretation with multiple-choice options.
• Core rule (composite-option check): For answers like "A+B+C," verify each component independently; select a composite only i f all listed statements are supported. Avoid "feature aggregation" bias (assuming partial match ⇒ whole option correct).

Updated Response (key excerpt to visualize):

• Applicability decision: The model explicitly labels **the guidance Applicable** because the task is histology-based and includes composite options requiring component-wise validation.
• Outcome: Selects Option (E), matching the standard answer.

Question: *Using a finite summation, compute the initial deflection at midspan for the beam in Figure P8.42. Given $E = 3000$ kips/in^2 .Use 3-ft segments. Assume $I = 0.5I_G$. Answer using a single word. <image>*

Ground-truth answer: *1.06 (units per problem context, typically inches)*



P8.42

Updated System Prompt (injected subject & guidance):

• Topic: Truss displacement using virtual work or method of consistent deformations.

• Core rule: apply a unit virtual load at the target point

→ compute real/virtual member forces: $\frac{\sum(\text{NnL})}{(\text{AE})}$;
Or solve redundant via compatibility (consistent deformations);
Emphasis: don't guess; ensure equilibrium closure; keep units consistent.

Updated Response (key excerpt to visualize):

• Applicability decision: **Irrelevant.** The guidance is for trusses (axial member forces, AE, virtual work), but the question is a beam deflection problem requiring finite summation / numerical integration of curvature using EI over 3-ft segments—different structural model, different governing equations.

Figure 5: **Qualitative examples of applicability judgment for retrieved guidance**. The model explicitly decides whether the injected subject-guidance pair is applicable: the first two cases are *Applicable* and lead to correct answers, while the last case is *Irrelevant* and is rejected.