ViLoMem: Agentic Learner with Grow-and-Refine Multimodal Semantic Memory

Supplementary Material

1. Additional Results and Ablation Study

1.1. Integration with more models

To verify the flexibility of **ViLoMem**, we extend our evaluation beyond the main experiments to recent reasoning-enhanced models, including GLM-4.1v [3], InternVL3-38B [10], and Gemini 2.5 [1]. As shown in Table 1, **ViLoMem** demonstrates robust adaptability across different architecture designs and inference regimes, consistently improving performance over both baseline and step-by-step configurations. This pattern echoes our observations in the main paper that visual perception remains a dominant bottleneck for multimodal reasoning [4, 9] and that decoupling visual distraction from logical hallucination yields complementary gains across tasks. Notably, models equipped with "thinking" or long-chain reasoning capabilities exhibit superior compatibility with the step-by-step format required for memory retrieval: their extended inference process allows for tighter integration of retrieved visual and logical guidelines into the reasoning chain, enabling them to correct potential errors before they propagate. These results suggest that **ViLoMem** is particularly well-suited to models with strong deliberative reasoning, while still offering consistent benefits to smaller or less capable solvers.

1.2. Attention Mechanism Ablation

Table 2 presents the ablation study of the attention mechanism. In general, the integration of attention maps yields consistent performance gains across hallucination and general reasoning benchmarks (e.g., HallusionBench, MM-Star), corroborating the critical importance of visual memory in refining perceptual grounding. However, we observe a performance plateau or marginal decline on mathematics-centric datasets (MathVista and MathVision). We attribute this limitation to two primary factors: (1) **Visualization Precision**: Current attention visualization methods struggle to faithfully preserve fine-grained geometric structures and chart details, which are essential for mathematical reasoning. (2) **Contextual Interpretation**: While serving as an auxiliary image to enhance visual context, the attention map imposes higher demands on the model's intrinsic capability to interpret heatmap overlays. The benefit of this enriched context is contingent on the model's ability to align these explicit visual cues with the raw image features without information loss.

1.3. Additional Case Study

Figure 1 summarizes representative qualitative cases. For many vision-intensive questions (e.g., traffic-light color, visible portion of the sun, object localization, and optical-illusion setups), logical memory is either not retrieved or fails to offer useful guidance, while visual memory provides concrete viewing strategies such as checking the actual illuminated region, reading tiny objects and relative positions from the viewer's frame, or isolating targets from distracting backgrounds. In these cases, attention maps concentrate on the queried regions (e.g., the active light, visible solar arc, or relevant segments), so that the retrieved visual guidelines directly steer the solver toward task-relevant evidence.

For geometry and chart-reading tasks, visual and logical memories are complementary: logical memory provides reusable rules for measurement and graph interpretation, while visual memory focuses on concrete inspection behaviors such as aligning with gridlines, following step edges, or checking true line orientation under strong illusions. Together, these cases highlight a clear division of labor: visual memory governs "where to look" and mitigates systematic perceptual traps, whereas logical memory refines "how to reason" once the correct visual evidence has been attended.

1.4. Comparison with Existing Memory Methods

We benchmark **ViLoMem** against state-of-the-art memory mechanisms [6, 8]. While the original Dynamic-Cheetsheet [6] employs cumulative memory, its unbounded context growth is infeasible for our large-scale setting (approx. 1,000 cases per benchmark), so we adopt the retrieval-based configuration from the open-source Dynamic-Cheetsheet codebase, which follows the similar methodology as ACE [8]. For a fair multimodal comparison, we replicate the official prompt structure and use the same MLLM for both memory generation and inference. In this setup, the retrieval module relies purely on text similarity without image-aware matching.

Experimental results in Table 2 show that this direct adaptation of logical memory methods is suboptimal in multimodal settings and can even underperform the baseline, especially for smaller models. In practice, such text-only retrieval often surfaces visually dissimilar examples with similar questions, resurfacing prior misperceptions as salient "hints" that misdirect attention away from the correct regions of the current problem. Qualitative inspection further reveals that Dynamic-Cheetsheet and ACE are tai-

Table 1. Additional evaluation results on GLM4.1v, InternVL3-38B, and Gemini2.5-flash across six multimodal reasoning benchmarks. Models with "(step)" and "(+ **ViLoMem**)" are prompted by step-by-step reasoning. Results demonstrate consistent improvements from **ViLoMem** across diverse model architectures.

Method	MMMU (dev)	MathVista (mini)	MathVision (mini)	HallusionBench	MMStar	RealWorldQA
GLM4.1v (baseline)	69.14	72.57	56.88	73.08	72.90	73.33
GLM4.1v (step)	70.29	73.47	58.22	72.77	73.40	72.54
GLM4.1v (+ **ViLoMem**)	**71.52**	**73.97**	**61.51**	**74.02**	**73.47**	72.68
InternVL3-38B (baseline)	62.92	70.80	35.53	67.40	69.33	71.99
InternVL3-38B (step)	64.18	71.90	35.56	71.50	67.80	72.42
InternVL3-38B (+ **ViLoMem**)	**65.97**	**73.80**	**36.84**	**72.34**	**69.73**	**73.20**
Gemini2.5-flash (baseline)	72.18	81.10	53.21	72.67	72.07	76.99
Gemini2.5-flash (step)	71.90	81.41	53.94	76.34	72.40	71.50
Gemini2.5-flash (+ **ViLoMem**)	**72.86**	**83.40**	**58.22**	**78.33**	**73.20**	76.42

Table 2. Comprehensive ablation study and comparison with existing memory methods across six multimodal reasoning benchmarks. We compare **ViLoMem** with attention mechanism variants and the Dynamic-Cheetsheet [6] baseline adapted for multimodal tasks.

Method	MMMU (dev)	MathVista (mini)	MathVision (mini)	HallusionBench	MMStar	RealWorldQA
GPT-4.1						
baseline	74.00	70.40	46.12*	58.50	69.80	73.72
step	74.16	74.27	47.47	74.44	70.43	72.03
+ dynamic-cheetsheet	70.95	73.87	48.68	75.30	68.68	70.13
+ **ViLoMem**	77.26	**76.88**	**53.95**	75.29	**72.43**	**74.38**
+ **ViLoMem** & attention	**78.21**	76.87	50.66	**75.73**	71.76	**74.38**
Qwen3-VL-235B-A22B-Instruct						
baseline	78.70	84.90	61.28*	63.20	78.40	79.30
step	75.97	83.66	62.17	74.58	76.16	78.66
+ dynamic-cheetsheet	72.13	83.25	60.06	70.62	75.49	77.11
+ **ViLoMem**	**79.40**	**84.98**	**62.83**	75.21	78.31	77.22
+ **ViLoMem** & attention	78.14	83.87	60.86	**75.95**	**78.46**	77.88
Qwen3-VL-8B-Instruct						
baseline	66.38*	77.20	48.13*	61.10	70.91	71.50
step	65.52	77.80	48.35	73.08	70.22	70.85
+ dynamic-cheetsheet	63.39	74.92	46.81	68.39	69.12	69.98
+ **ViLoMem**	**69.90**	**77.87**	**49.34**	73.19	**72.13**	**73.59**
+ **ViLoMem** & attention	67.52	77.07	48.72	**74.87**	72.67	73.46

lored to code- or logic-centric schemas: even when driven by an MLLM, they mainly produce fine-grained corrections of specific visual details (digits, colors, marks) rather than robust guidance on how to inspect diagrams. These detail-level cues lack stable visual grounding and easily conflict with the actual image, inducing additional hallucinations that smaller models are particularly vulnerable to. This contrast highlights the need for **ViLoMem**'s decoupled visual stream and question-aware retrieval, which explicitly organize and retrieve perception-oriented error patterns instead of repurposing logic-only memories.

Retrieval Efficiency at Scale. We further analyze how the two-stage retrieval pipeline scales with growing memory. Table 3 compares accuracy and per-case latency (ms) on WeMath for: no memory (S0), visual-only retrieval (S1), text-only retrieval (S2), and two-stage retrieval ($S_{\oplus}$ = S1 + S2). As memory grows from 87k to 150k tokens, S2 latency increases from 22.8ms to 115ms due to full-corpus text matching, while $S_{\oplus}$ latency only rises from 20.4ms to 61.6ms, a 22.3× reduction compared to S2 alone. Due to S1 first narrows the candidate set via efficient image embedding search (1024-dim), allowing S2 to rerank only the top candidates. Crucially, $S_{\oplus}$ also achieves the highest accuracy (74.58%), as the two stages provide complementary filtering: S1 captures visual similarity while S2 refines by semantic relevance.

Table 3. Two-stage retrieval analysis on WeMath. S0: no memory (baseline); S1: visual retrieval; S2: text retrieval; $S_{\oplus}$: S1 + S2 (two-stage). Latency in ms per case.

Stage / Acc. % (Latency ms)	S0	S1	S2	$S_{\oplus}$
87k tokens	72.07	72.94 (19.1)	73.22 (22.8)	73.91 (20.4)
150k tokens	72.07	73.72 (60.7)	73.41 (115)	**74.58** (61.6)

Table 4. Efficiency comparison between ViLoMem and Dynamic-Cheetsheet (DC) on MathVista. ViLoMem achieves superior accuracy with substantially lower retrieval cost and storage.

Method	Acc. (%)	Retrieval (ms)	Storage (MB)	#Memory (tokens)
Baseline	70.40	0	0	0
DC	73.87	325	18.44	221K
ViLoMem	**76.88**	**120**	**6.12**	**73K**

We further compare the computational efficiency of **ViLoMem** against Dynamic-Cheetsheet (DC) on MathVista in Table 4. **ViLoMem** achieves higher accuracy (+3.01) while requiring significantly lower retrieval latency (-63.1%) and storage cost (-66.8%). This efficiency stems from our selective memory update strategy: **ViLoMem** only generates memory entries for incorrect answers, whereas DC updates memory for every case, leading to a bloated memory pool (221K vs. 73K tokens) with proportionally higher retrieval overhead.

1.5. Failure Case Analysis

To understand the limitations of **ViLoMem**, we analyze cases on MMMU (1,041 samples, GPT-4.1 as solver) where the baseline answers correctly but **ViLoMem** fails. Overall, **ViLoMem** achieves a net gain of +8.86% over baseline; however, we identify 66 regression cases where memory retrieval hurts performance. Among these, 33.3% (22 cases) are caused by *generic visual memory*: the retrieved visual cues are only weakly relevant to the specific image and question, lacking image-specific and question-specific adaptation, which distracts reasoning despite being knowledge-correct. The remaining 66.7% (44 cases) are caused by *empty retrieval*: no memory is retrieved at all (0 matched entries), meaning the model receives no memory-augmented guidance and instead relies on a step-by-step prompt that may differ from the baseline's default prompting. Notably, no failures occur when both visual and logical memory are successfully retrieved, suggesting that the dual-stream retrieval mechanism is reliable when sufficient memory coverage exists.

1.6. Visual Memory Characterization

As shown in Table **??**, **ViLoMem** achieves 77.26% on MMMU with GPT-4.1, with both visual and logical memory contributing to the overall gain. To understand *which types of tasks* benefit most from each memory stream, we further analyze the 835 subject-matched samples across MMMU's six academic disciplines. Table 5 reports the improvement from the full system over baseline (Δ(B$\rightarrow$VLM)), and the independent contribution of each memory stream: Δ(vis) = ViLoMem $-$ w/o visual, and Δ(log) = ViLoMem $-$ w/o logic.

The results reveal clear discipline-dependent memory utilization patterns:

Table 5. Per-discipline memory contribution analysis on MMMU (GPT-4.1, 835 subject-matched samples). Δ(vis) and Δ(log) measure the independent contribution of visual and logical memory, respectively. Bold indicates the dominant memory stream.

Discipline	**N**	**Δ(B$\rightarrow$VLM)**	**Δ(vis)**	**Δ(log)**
Tech & Engineering	205	+12.7	**+9.8**	+5.4
Health & Medicine	99	+10.1	+1.0	**+8.1**
Humanities & Social Sci.	137	+9.5	+0.7	+0.0
Science	136	+8.1	+3.7	**+4.4**
Art & Design	93	+3.2	+1.1	**+4.3**
Business	165	+0.0	-3.6	-4.2

Visual memory dominates in Tech & Engineering (Δ(vis)=+9.8 vs. Δ(log)=+5.4). At the subject level, Energy & Power (+20.0), Math (+19.4), Agriculture (+14.3), and Mechanical Engineering (+12.5) benefit most from visual memory. These subjects feature specialized diagrams (circuit schematics, engineering drawings, mathematical plots, microscopic images) that require recognizing domain-specific visual patterns beyond the model's general vision capability.

Logical memory dominates in Health & Medicine (Δ(log)=+8.1 vs. Δ(vis)=+1.0). Diagnostics & Laboratory Medicine shows the largest overall gain (+24.2 from baseline), but the improvement is primarily driven by logical memory (+9.1), indicating that medical reasoning chains—rather than visual perception—are the main bottleneck.

Both streams are ineffective for Business (Δ(vis)=-3.6, Δ(log)=-4.2). The model's baseline visual encoder already handles standard business charts and tables well, and the retrieved memory introduces noise that harms performance. This highlights that memory augmentation is most valuable when the task exceeds the model's inherent capabilities.

1.7. Results on Visual Perception Benchmarks

On the math perception benchmark MathGlance [5], **ViLoMem** generates 166 visual and 16 logic memory entities, achieving +1.87% average accuracy improvement over the no-memory baseline on plane/solid geometry and graph tasks. Note that purely perceptual benchmarks such as OCRBench and DocVQA are not directly applicable to our framework, as they use scoring-based evaluation (character matching) rather than binary correctness judgments, which prevents the error-driven memory generation process.

2. Additional Experimental Details

This section provides additional implementation details that complement the experimental setup.

Model Deployment. For open-source models, we deploy most checkpoints using `vLLM` for efficient batched inference. Due to its scale, *Qwen3-VL-235B-A22B-Instruct* is accessed via its official API instead of local deploy-

ment, and all proprietary models (e.g., GPT-4.1, Gemini 2.5 flash) are evaluated through their corresponding APIs. For API-based evaluations, certain images or prompts may be flagged as unsafe by the provider's safety filters and thus rejected, which introduces a small amount of noise into the reported scores.

Decoding Hyperparameters. Unless otherwise specified, we use a temperature of 0.7 and a maximum generation length of $8{,}192$ tokens for all models. Within our memory pipeline, the maximum generation length is set to $1{,}024$ tokens for problem analysis and $2{,}048$ tokens for memory generation to balance expressiveness and efficiency. Baseline evaluations directly feed benchmark questions to the models without additional prompts, whereas the *Step* configuration prepends a simple step-by-step system prompt; the full template is shown in Figure 2.

Attention Map Generation. Attention maps are generated following the training-free small-detail perception framework of Zhang et al. [7], instantiated with *Qwen2.5-VL-3B* as the backbone model. This setup produces token-level saliency over input images, which we overlay as heatmaps to visualize and interpret visual memory retrieval.

Evaluation Protocol. We adopt VLMEvalKit [2] as the primary evaluation framework. When automatic matching fails or produces ambiguous results (e.g., due to formatting variations), we further apply *Math-Verify* and an LLM-as-a-judge protocol to reduce sensitivity to output formatting. The judge model is *Qwen3-8B-Instruct*, which assesses whether a model's response is semantically correct with respect to the reference answer.

3. Prompt Templates

We provide the full prompt templates used in our framework, including the step-by-step reasoning prompt used in the *Step* configuration (Figure 2), the Problem Analysis Prompt (Figure 3), the Logical Memory Generation Prompt (Figure 4), and the Visual Memory Generation Prompt (Figure 5), together with the LLM-as-a-judge verification prompt (Figure 6).

References

[1] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. 1

[2] Haodong Duan, Junming Yang, Yuxuan Qiao, Xinyu Fang, Lin Chen, Yuan Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Jiaqi Wang, et al. Vlmevalkit: An open-source toolkit for evaluating large multi-modality models. In *Proceedings of the 32nd ACM international conference on multimedia*, pages 11198–11201, 2024. 4

[3] Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, et al. Glm-4.5 v and glm-4.1 v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning. *arXiv preprint arXiv:2507.01006*, 2025. 1

[4] Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. In *The Twelfth International Conference on Learning Representations*. 1

[5] Yanpeng Sun, Shan Zhang, Wei Tang, Aotian Chen, Piotr Koniusz, Kai Zou, Yuan Xue, and Anton van den Hengel. Mathglance: Multimodal large language models do not know where to look in mathematical diagrams. *arXiv e-prints*, pages arXiv–2503, 2025. 3

[6] Mirac Suzgun, Mert Yuksekgonul, Federico Bianchi, Dan Jurafsky, and James Zou. Dynamic cheatsheet: Test-time learning with adaptive memory. *arXiv preprint arXiv:2504.07952*, 2025. 1, 2

[7] Jiarui Zhang, Mahyar Khayatkhoei, Prateek Chhikara, and Filip Ilievski. Mllms know where to look: Training-free perception of small visual details with multimodal llms. In *The Thirteenth International Conference on Learning Representations*. 4

[8] Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, et al. Agentic context engineering: Evolving contexts for self-improving language models. *arXiv preprint arXiv:2510.04618*, 2025. 1

[9] Shan Zhang, Aotian Chen, Yanpeng Sun, Jindong Gu, Yi-Yu Zheng, Piotr Koniusz, Kai Zou, Anton Van Den Hengel, and Yuan Xue. Primitive vision: Improving diagram understanding in mllms. In *International Conference on Machine Learning*, pages 74732–74755. PMLR, 2025. 1

[10] Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, et al. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025. 1

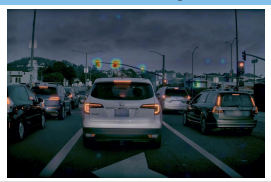
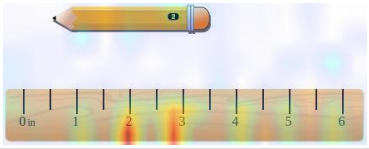
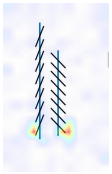
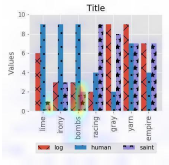
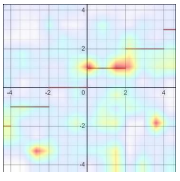
Question	Logic Memory	Visual Memory	Attention Map
CASE 1 **Is the traffic light green?**	/	When counting illuminated traffic lights, **verify the color of each light individually and confirm it is part of a traffic signal assembly**, not a decorative or non-traffic light fixture.	
CASE 2 **What percent of the sun is showing?**	When calculating the range of a data set, always double-check that the minimum and maximum values are correctly identified by reviewing all data points, especially when values repeat or are close in magnitude....	When **identifying 'tiny thing' or relative positions**, always **verify object size and spatial layout from the viewer's perspective**; matte finish and right-side positioning must be visually confirmed, not assumed....	
CASE 3 **Move the ruler to measure the length of the pencil to the nearest inch. The pencil is about (_) inches long.**	When solving geometry problems involving squares inscribed in circles or angles formed by intersecting chords, verify key relationships such as the diagonal of the square equaling the circle's diameter and the angle measure being half the sum of the intercepted arcs.....	Always **verify the exact starting point of the object being measured on the ruler**; do not assume alignment with 0 cm unless visually confirmed, as the object may begin at a non-zero mark....	
CASE 4 **Are blue lines in the image parallel? Yes or No**	When applying the Corresponding Angles Postulate, always verify that the angles lie on the same side of the transversal and in matching positions at each intersection.....	When assessing parallelism of lines surrounded by diagonal patterns, use a ruler or grid overlay to verify true orientation and spacing, as the **background can induce a perceptual illusion of verticality or parallelism**. When comparing line lengths in an image, **measure or visually align them directly rather than relying on assumptions about known optical illusions, as the actual lengths may differ from the illusion's typical setup....**	
CASE 5 **Where is the sheep?**	/	When identifying objects relative to others in a scene, **always verify both the object's identity and its spatial position (left/right/front/back) by comparing their actual locations in the image**, not assumptions based on size or context.	
CASE 6 **What is the position of the sink relative to the refrigerator?**	When a question asks for the direction of object A relative to object B, ensure the reference frame is correctly oriented: "**A in relation to B**" means you start from B and **determine where A lies, not the reverse**. Always double-check the subject and reference point in directional questions to avoid reversing the relationship.	When identifying objects relative to others in a scene, **always verify both the object's identity and its spatial position (left/right/front/back) by comparing their actual locations in the image**, not assumptions based on size or context.	
CASE 7 **What is the value of the smallest individual bar in the whole chart?**	When interpreting values on a logarithmic scale, remember that tick marks represent powers of 10, and values between 10^2 and 10^3 range from 100 to 1000; always verify whether a bar exceeds a linear threshold like 100 by estimating its actual value, not just its position relative to 10^2...	On a logarithmic scale, a bar ending exactly at a labeled tick (e.g., 10^2) represents that exact value, not a value greater than it; **always check if the bar exceeds the tick mark to determine if it's strictly larger**....	
CASE 8 **When does the function value first reach 2?**	When matching a correctly identified element (e.g., a region, value, or object) to a multiple-choice option, always verify that the letter of the choice corresponds to the correct labeled item in the diagram or question, not just the reasoning outcome. Double-check the mapping between your conclusion and the answer options to avoid selection errors....	When interpreting a step function graph, **always verify the exact y-value of each horizontal segment by aligning it with the y-axis gridlines**, rather than assuming values based on adjacent steps or integer patterns. When identifying the y-intercept on a graph, **always trace the curve to where it crosses the y-axis (x=0) and read the exact y-value at that point**, not the vertex or any nearby grid line....	
CASE 9 **Does this figure mainly depict a hen and eggs, no potatoes?**	When a question asks whether two colors are different in the context of an optical illusion, always consider whether it is asking about objective color values (e.g., RGB or physical properties) rather than perceived appearance; remember that illusions affect perception, not necessarily reality.	When comparing the color of objects on a gradient background, **verify that perceived differences are not caused by the background's luminance shift by isolating or masking the background to assess the objects' true color**...	

Figure 1. Showcase of representative cases demonstrating **ViLoMem**'s memory generation and retrieval process across different types of multimodal reasoning tasks.

Prompt: Step-by-Step Reasoning

Objective: Solve the given problem using a step-by-step process.

Expected Output Structure:

```
Step 1:
Step 2:
...
Step n: Final Answer: \boxed{answer}

Question:
```

Figure 2. The step-by-step reasoning system prompt used in the *Step* configuration.

Prompt: Problem Analysis

Objective:
Analyze the following problem to identify its subject area and the key concepts, principles, formulas, or laws required for its solution. This analysis will be used to retrieve relevant guiding principles from a knowledge base.
Instructions:
- Do not solve the problem.
- First, identify the primary subject (e.g., Physics, Chemistry, Biology, Mathematics).
- Then, list the core concepts or principles involved (e.g., Newton's Second Law, Conservation of Energy, Stoichiometry, Pythagorean theorem).
- Keep the analysis concise and focused.

Problem:
{question}
Output Format:
Subject: <The primary subject>
Key Concepts: <A brief list of key concepts>

Figure 3. The prompt template for analyzing the problem to identify its subject and key concepts.

Prompt: Logical Memory Generation

Objective:
Analyze the provided incorrect reasoning process for a scientific or mathematical problem. Your goal is to classify the error and, if it is a logical error, generate a high-quality, actionable guideline (a "memory") to prevent similar mistakes in the future.

Context:
- Problem: {question}
- Incorrect Reasoning Steps: {reasoning_steps}
- Correct Answer (for reference): {gold_answer}

Instructions:

1. **Analyze the Mistake:** Carefully review the `Incorrect Reasoning Steps` against the `Problem` and `Correct Answer` to pinpoint the primary mistake.
2. **Categorize the Error:** Classify the error into one of two types:
   - `Logical`: Any error in the reasoning process itself. This includes calculation mistakes, misapplication of a formula or theorem, logical fallacies, or conceptual misunderstandings. These errors can be identified from the text of the reasoning alone.
   - `Non-Logical`: An error that stems purely from misinterpreting the visual information in an image. This kind of error can **only** be confirmed by looking at the image (e.g., misidentifying a shape, reading a value from a graph incorrectly).
3. **Generate the Guideline (Memory):**
   - **Only if the `error_type` is `Logical`**, you must generate a guideline.
   - If the `error_type` is `Non-Logical`, the guideline must be left empty.

   Guideline Quality Requirements:
   - **Be Specific and Concrete:** The guideline must target the specific principle, theorem, formula, or reasoning pattern that was misused. Name the concept directly.
   - **Be Actionable:** Frame it as a clear instruction, a warning, or a rule of thumb (e.g., "Always check...", "Remember to differentiate between...", "When X occurs, apply Y...").
   - **Be Generalizable:** The advice should be abstracted from the specific numbers and context of this single problem so it can apply to a whole class of similar problems.
   - **Keep it Concise:** The guideline should be one to two sentences long.

   Guideline Examples (Good, Specific Examples):
   - **(Physics):** "When applying the conservation of energy to rolling objects, always include both translational and rotational kinetic energy in the equation."
   - **(Math):** "In geometry problems involving tangents to a circle, remember that the radius to the point of tangency is perpendicular to the tangent line."
   - **(Chemistry):** "For stoichiometry calculations, always ensure the chemical equation is correctly balanced before determining the molar ratios."

Output Format (use this exact structure):
error_type: <"Logical" or "Non-Logical">
analysis_summary: <A brief, one-sentence summary of what went wrong.>
guideline: <Your 1-2 sentence guideline if the error is "Logical", otherwise leave this field empty.>

Figure 4. The prompt template for generating logical memories.

Prompt: Visual Memory Generation

Objective:
You are an expert in visual reasoning and error analysis. Your task is to first describe the provided image objectively, then analyze an incorrect reasoning process to determine if the error stems from misinterpreting that image. If a visual error is found, you must generate a concise, actionable guideline (a "visual memory") to prevent this mistake in the future.

Context:
- Problem: {question}
- Incorrect Reasoning Steps: {reasoning_steps}
- Correct Answer (for reference): {gold_answer}

Attached Image: <image>

Thinking Process and Final Output:
Your response must follow a strict two-stage process. The first stage is your internal "thought process" which you will write out. The second stage is the final JSON output.

Stage 1: Internal Thought Process (Write this out first)
1. **Describe the Image:** Begin by providing an objective, detailed description of the attached image. List all key elements, labels, values, geometric shapes, and their relationships. This description will serve as the "ground truth" for your analysis.
2. **Analyze for Discrepancies:** Compare your image description and the image itself against the text in `Incorrect Reasoning Steps`. Identify any contradictions, misinterpretations, or omissions.

Stage 2: Final JSON Output (Provide ONLY this JSON block as the final answer)
After completing your thought process, generate a JSON object based on your analysis. The JSON should adhere to the following structure and guidelines.

Guidelines for `guideline` Generation:
- The guideline MUST be about how to correctly interpret a specific visual pattern or element.
- It must be a rule that can be applied to other, similar-looking problems.
- It should be concise (one to two sentences).

Guideline Examples (Good, Specific Visual Memories):
- **(Physics/Diagrams):** "In a free-body diagram, always verify that all forces, including friction and normal force, are accounted for before applying Newton's laws."
- **(Geometry):** "When an angle appears to be a right angle in a diagram, do not assume it is 90 degrees unless it is explicitly marked with a square symbol."
- **(Chemistry/Molecules):** "For complex organic molecules, double-check the placement of double bonds and functional groups as they dictate the molecule's reactivity."
- **(Biology/Graphs):** "When reading a bar chart, pay close attention to the Y-axis scale and units to avoid misinterpreting the magnitude of the results."

Avoid these types of guidelines (Bad, Non-Visual or Too Vague):
- "The model made a calculation error." (This is a logical error, not visual)
- "You need to look at the image more carefully." (Not actionable)
- "The reasoning about the physics was wrong." (Too general)

Final Output Format (use this exact JSON structure):

```
{
    "is_visual_error": true/false,
    "analysis_summary": "A brief, one-sentence summary of the visual
                         misinterpretation.",
    "guideline": "Your 1-2 sentence visual guideline. Provide this
                 only if is_visual_error is true, otherwise it
                 should be null."
}
```

Figure 5. The prompt template for generating visual memories.

Prompt: LLM-as-a-Judge Verification

Objective:
You are an expert answer verification judge. Your task is to determine whether a model prediction matches the gold answer.

Core Principle (Critical Rule):
- All decisions are based *only* on the gold answer; ignore the quality of the reasoning.
- If the extracted final answer from the prediction exactly matches the gold answer, set `verified=true`; otherwise, set `verified=false`.
- Do not consider whether the prediction's reasoning is correct or sensible.
- Do not give partial credit for "close" answers (e.g., $2 \neq 9$, C $\neq$ A).

Verification Steps:
1. **Identify the gold answer format.** Determine whether the gold answer is:
   - a single letter (`A/B/C/D/E`) for multiple-choice questions (compare letters only);
   - a number for numerical questions (compare numeric values, ignoring formatting such as 7 vs. 7.0);
   - a text span for open-ended questions (compare semantic meaning, allowing minor wording differences).
2. **Extract the final answer from the prediction.**
   - For multiple-choice questions, locate the final chosen letter (often after "Final Answer:" or "Answer:") and compare it with the gold letter.
   - For numerical questions, locate the final numeric value, ignoring units and extra text, then compare it to the gold number.
   - For text answers, extract the final answer phrase and compare its semantic meaning with the gold text (e.g., "Yes, the baby is crawling to the right." matches "Yes").
3. **Apply the strict matching rule.**
   - Only compare the final extracted answers.
   - Do not use external knowledge to judge whether an answer is reasonable.
   - If the extracted answer and the gold answer match under the appropriate format, output `verified=true`; otherwise, output `verified=false`.

Input Fields:
- Question: {question}
- Gold Answer: {gold_answer}
- Choices (optional, for multiple choice): {choices_text}
- Prediction: {prediction}

Output Format (JSON, exact structure):

```
{
  "reasoning": "Step 1: Extract answer from prediction: [extracted_value].
Step 2: Compare with gold: [gold_value].
Step 3: Match result: [yes/no].",
  "verified": true or false
}
```

Figure 6. The LLM-as-a-judge prompt template used to verify whether a model prediction matches the gold answer, independent of reasoning quality.