

Learning Generalizable Thinking Composition for Multimodal Reasoning

Anonymous ACL submission

Abstract

Recent advancements in multimodal reasoning have raised interest in improving interpretability and generalization, motivating structured reasoning as a promising approach. While prior studies have explored structured reasoning through various problem-specific designs, they often lack a unified perspective on how reasoning structures can be composed across tasks. In this work, we take a step toward addressing this by modeling reasoning as the dynamic composition of cognitive-inspired reasoning modules, enabling the system to construct flexible and generalizable reasoning chains. Specifically, we introduce a two-stage **compositional reasoning framework**: we first warm up the chain composer with high-quality module chains generated by feedback-guided greedy search, and then apply PPO-based structure policy optimization to learn a more generalizable composition policy. Our framework yields strong cross-task generalization. When trained on a single math-centric dataset, the composer consistently improves performance across diverse math benchmarks and further extends to general-domain multimodal tasks, indicating that it learns cross-task composition rules rather than dataset-specific heuristics. On the challenging MMMU-Pro benchmark, our model, based on Qwen2.5-VL-7B, achieves a 7.11% gain over its backbone, approaching the performance of the much larger Qwen2.5-VL-72B. These results support our central claim that reasoning-chain composition follows patterns that are consistent across tasks, which in turn enables the composer to learn reusable module-composition rules for multimodal reasoning.

1 Introduction

Recent advances in multimodal large language models (MLLMs) have led to remarkable performance across a wide range of vision-language tasks such as visual question answering (QA), image captioning, and visual grounding (Bai et al., 2025;

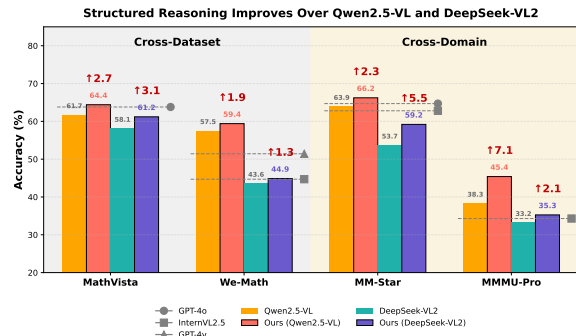


Figure 1: **Cross-distribution generalization across datasets and domains.** Our method improves over Qwen2.5-VL by +2.7% on MathVista and +7.1% on MMMU-Pro, and over DeepSeek-VL2 by +3.1% and +2.1%, showing consistent gains across datasets and cross-domain benchmarks.

Chen et al., 2024b). As MLLMs are applied to increasingly complex tasks, their success increasingly depends on the ability of reasoning, *i.e.*, constructing logical and effective intermediate steps to guide the inference process.

To this end, many efforts have been made to enhance the reasoning abilities of MLLMs, which broadly fall into two paradigms: reinforcement learning (RL) with task-specific reward signals (DeepSeek-AI et al., 2025; Liu et al., 2025; Zhou et al., 2025), and Chain-of-Thought (CoT) techniques, which guide models to generate intermediate steps, either through supervised fine-tuning (Xu et al., 2025; Xiang et al., 2025) or in-context prompting (Zhang et al., 2024d; Gao et al., 2024).

Compared to RL methods which typically demand task-specific verifiable reward signals and retraining, CoT-style approaches offer a more flexible framework that achieves impressive zero-shot or few-shot performance across various reasoning tasks.

This empirical success has demonstrated the promise of decomposing complex reasoning into

interpretable steps. However, CoT-style reasoning often lacks explicit structure and semantic consistency, limiting controllability across tasks (Wang et al., 2025). These challenges have spurred the development of *structured reasoning*, which aims to enable transparency and systematic control over the reasoning process, yet several key challenges remain. (1) Many existing methods are tailored to specific tasks or rely on LLMs to generate fine-grained reasoning steps (Xiang et al., 2025), which may result in fragmented, low-abstraction chains that are difficult to generalize. (2) Supervised fine-tuning (SFT)-based approaches (Gao et al., 2024; Shi et al., 2024) depend heavily on problem-specific annotations and require significant computational resources, limiting scalability. These lead us to consider: could the way cognitive operations are composed follow a shared, transferable principle that generalizes across tasks?

Drawing on the Core Knowledge Theory (Spelke and Kinzler, 2007), which argues that cognitive modules can be flexibly composed to adapt novel tasks, we model reasoning as a compositional process over reusable operations. Building on this principle, we introduce the **Compositional Reasoning Framework**, which enables dynamic and task-adaptive composition of cognitively motivated modules into reasoning chains.

Central to our framework is a structure composer. We train this via a two-stage process: (1) **a warm-up stage** that employs feedback-guided greedy search to construct high-quality pseudo-labeled chains based on task correctness, and (2) **a structure policy optimization stage**, where we apply PPO (Schulman et al., 2017) to further train the composer to generate generalizable reasoning chains. This training pipeline enables the model to learn structured reasoning without human annotations, and to generalize across tasks through modular recomposition rather than retraining.

Empirically, our framework achieves strong performance across diverse benchmarks. We find that, when trained on a single math-focused dataset, the structure composer can generalize effectively to diverse math benchmarks and even general-domain tasks. This observation leads to the hypothesis that the compositional structure of reasoning, namely how modular operations are selected and sequenced, captures a task-independent cognitive regularity.

We summarize our contributions as follows:

- We propose a **two-stage structure composition strategy** for reasoning chain generation: (1) a feedback-guided greedy search produces high-quality pseudo-labeled chains to warm up the structure composer; (2) structure policy optimization, where the composer is trained with PPO, further improves the reasoning.
- We conduct **comprehensive evaluations** across cross-dataset and cross-domain benchmarks. Our method demonstrates strong structure-level generalization, achieving a +7.11% gain on MMMU-Pro over Qwen2.5-VL-7B, approaching the performance of Qwen2.5-VL-72B.
- Our findings **suggest a possibility**: the way reasoning modules are composed tends to follow consistent patterns across tasks, pointing toward a transferable cognitive structure.

2 Related Work

Multimodal Mathematical Reasoning. Solving math problems that require joint visual and textual understanding remains a fundamental challenge for MLLMs. Zhang et al. (Zhang et al., 2024c) introduced the MAVIS dataset to support supervised training on complex math problems involving diagrams. Similarly, Zhuang et al. (Zhuang et al., 2024) proposed Math-PUMA, a progressively structured dataset designed to teach upward visual reasoning. Beyond supervised fine-tuning, Luo et al. (Luo et al., 2024) and Wang et al. (Wang et al., 2024b) explored reinforcement learning techniques by assigning rewards to final or intermediate reasoning steps. Prompt-based strategies, particularly Chain-of-Thought (CoT) (Zhang et al., 2024d), have also been adopted to elicit step-by-step reasoning through natural language prompts in multimodal settings.

Structured Reasoning in MLLMs. To improve reasoning control and interpretability, recent studies have explored structured reasoning that explicitly organizes reasoning processes. For instance, CoTDet (Tang et al., 2023), IPVR (Chen et al., 2023), and LLaVA-CoT (Xu et al., 2025) decompose the problem-solving process into a sequence of predefined steps. Other methods, such as CoT-Valve (Ma et al., 2025) and the Self-Adaptive Chain framework (Mitra et al., 2024), dynamically control reasoning length through learned controllers. Search-based strategies such as Monte Carlo Tree

Search (MCTS) (Zhang et al., 2024a; Dong et al., 2024a) have also been considered for reasoning structure optimization, but they incur prohibitive inference cost and lack transferability across tasks, motivating approaches that learn reusable policies instead.

While existing approaches often address reasoning with fixed or implicit structures, they typically do not adopt a task-unified perspective on structural adaptation. Our framework explores this direction by modeling reasoning as an adaptive composition of modular operations, aiming to provide flexible and general reasoning across diverse tasks.

3 Method

Problem Formulation. We aim to address the task of *multimodal reasoning with structured thinking*, where the system takes a pair of textual question and associated image as input, then generates an accurate answer through a structured sequence of reasoning steps. Specifically, given a pair of text-image input $x = (x_{\text{text}}, x_{\text{image}})$, the model needs to infer a variable-length composition of reasoning modules C from a k -length set $\mathcal{M} = \{m_1, m_2, \dots, m_k\}$. This is a crucial part of our method, where the model learns an adaptive mapping from input x to the composition of reasoning blocks C :

$$f : x \mapsto C = [m_{i_1}, m_{i_2}, \dots, m_{i_T}], m_{i_t} \in \mathcal{M},$$

where i_t is the index of the selected module at step t . Then, an execution process is conducted to apply each reasoning module sequentially to the input and intermediate context to get the final answer a . The executor function $a = \text{Exec}(C, x)$ is implemented by adaptively prompting the large language model with automatically composed reasoning blocks C .

This structured thinking formation provides three key advantages. First, it decouples the reasoning structure from the underlying large language model, allowing more efficient and flexible learning. For example, we can introduce useful external knowledge such as “reflection” with minimal cost. Second, it enables explicit control over the reasoning process, supports interpretability. Third, it facilitates transferability across tasks and domains, as we will show in results in the experiments section.

In the following, we elaborate on our method through the description of four main components: 1) reasoning modules design (§3.1), 2) structure composition warm-up (§3.2), 3) structure policy optimization (§3.3), and 4) reasoning execution (§3.4).

3.1 Reasoning Module Design

According to cognitive theories (Lake et al., 2016; Andreas et al., 2017), human cognitive processes typically rely on modular thinking patterns to solve complex reasoning tasks. Inspired by this, we propose to mirror human reasoning behaviors by defining a symbolic space of reasoning modules, in which each module explicitly represents a core cognitive function such as perceiving or reflecting. This modular design enables transparent and extensible reasoning, offering the potential to generalize across diverse tasks. To quantify the contribution of each module, we include detailed module ablations in the Appendix A.3.8.

Module Name	Function Description
Visual (Andreas et al., 2017)	Analyze the image.
Prior (Chen et al., 2023)	Apply relevant domain laws.
Logic (Xu et al., 2025)	Step-by-step reasoning.
Reflection (Barrouillet, 2011)	Revise the solution path.
Problem Decompose (Wu et al., 2025)	Break down the problem into sub-problems.
Cross-Modal Linking (Chen et al., 2023)	Connect visual cues with textual knowledge.
Hypothesis (Klahr and Dunbar, 1988)	Propose plausible assumptions or interpretations.
Validation (Barrouillet, 2011)	Confirm intermediate conclusions using visual evidence or logical consistency.

Table 1: Cognitive-inspired atomic reasoning modules. Each module corresponds to a distinct cognitive function.

Table 1 summarizes the reasoning modules and their corresponding functions. Detailed definitions and prompt templates for each module can be found in Appendix A.1.

3.2 Structure Composition Warm-up

To avoid manual annotation, we construct warm-up reasoning chains using greedy, feedback-guided search. Given a question-image pair and answer, the method incrementally builds a module chain that maximizes answer accuracy.

Starting with an empty chain ending in `final_answer`, we skip expansion if it correctly solves the task. Otherwise, we sample candidates by past success rates, test them in trials, and pick the one with the best expected reward—repeating until a perfect score is reached or the chain hits its length limit.

The reward for each module is computed as:

$$\text{reward} = \alpha \cdot \frac{\text{correct_count}}{N} - \beta \cdot \left(1 - \frac{\text{correct_count}}{N}\right)$$

where $\alpha, \beta > 0$ are weighting coefficients, and

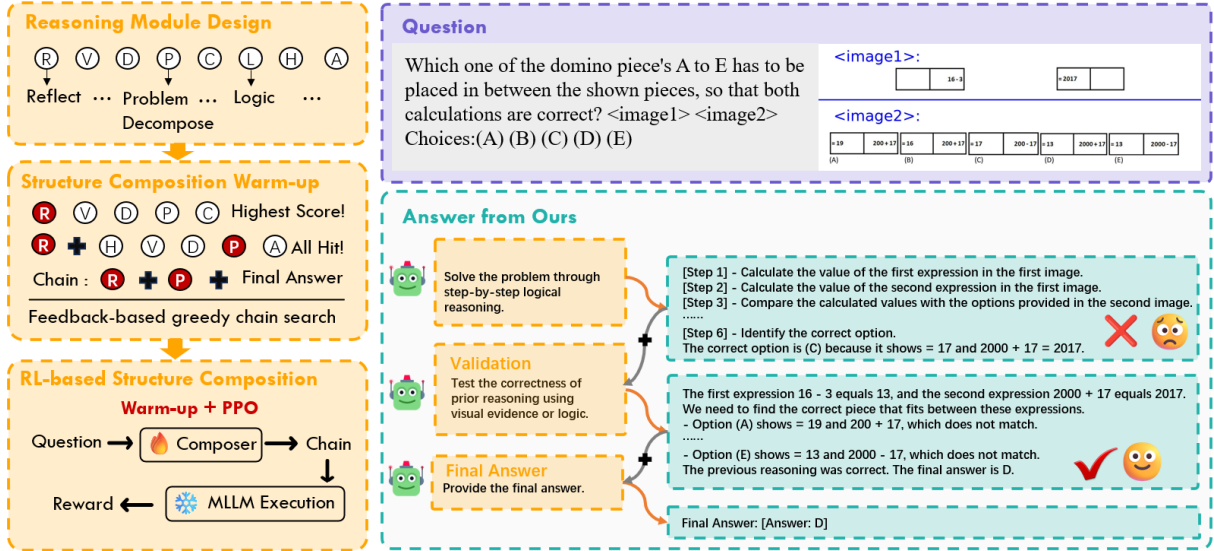


Figure 2: **Method Overview.** The left side illustrates our framework, which consists of three stages: (1) **Reasoning Module Design**, where a set of cognitive-inspired reasoning modules is defined to capture fundamental problem-solving functions; (2) **Structure Composition Warm-up**, where high-quality pseudo-chains are constructed via feedback-guided greedy search and used to warm up the composer; and (3) **Structure Policy Optimization**, where the structure composer is further optimized through PPO to improve chain generation. The right side presents an example of reasoning execution based on the predicted chain.

N is the number of trials. Historical scores are updated using a smoothing factor γ . We apply a linear shift-and-scale to keep reward magnitudes in a suitable range for stable softmax sampling, thereby maintaining a balanced exploration–exploitation trade-off.

This method yields a high-quality, task-adaptive warm-up set for learning structured reasoning chains. These chains are used to initialize the policy network (§3.3). The overall procedure is summarized in Algorithm 1.

3.3 Structure Policy Optimization

Since multiple valid reasoning chains may exist for a given question, the optimal solution is better viewed as a distribution rather than a single sequence. To capture this diversity and learn to generate high-quality chains, we adopt Proximal Policy Optimization (PPO) to optimize the composer. The chain generation process is formalized as a finite-horizon Markov Decision Process (MDP), defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$:

- **State space \mathcal{S} :** Each state s_t encodes the input $(x_{\text{text}}, x_{\text{image}})$ and the partial module chain $[m_1, \dots, m_{t-1}]$ generated so far;
- **Action space \mathcal{A} :** Each action a_t selects the next module from $\mathcal{M} \cup \{\langle \text{eos} \rangle\}$;

- **Transition function \mathcal{P} :** Deterministic extension of the chain, i.e., $s_{t+1} = s_t \cup \{a_t\}$;
- **Reward function \mathcal{R} :** Defined at the end of each episode based on the execution correctness of the full chain;

The policy $\pi_\theta(a_t | s_t)$ is parameterized by a Transformer decoder that autoregressively predicts the module sequence. We train this policy using Proximal Policy Optimization (PPO), which stabilizes learning by clipping updates to the policy distribution.

To stabilize training, we initialize both policy and value networks using warm-up data from greedy search, leveraging structure chains and their stability signals to accelerate early learning. The subsequent PPO optimization directly maximizes our reward, enabling the composer to discover more robust and generalizable chains than those attainable via greedy warm-up alone.

3.3.1 Reward Design

Each predicted chain is executed multiple times to estimate its output consistency. The final reward is composed of three terms:

$$r = \alpha \cdot \text{stability} + \gamma \cdot \text{baseline_gain} + \text{penalty},$$

where:

Algorithm 1: Feedback-Guided Greedy Chain Search

Input: Input $(q, \text{image}, \text{answer})$; module pool \mathcal{M} ; max length L ; trials N
Output: Final structure chain C

```
1 Initialize  $C \leftarrow [\text{final\_answer}]$ 
2 if  $\text{run}(C)$  is correct in all trials then
3   | return  $C$ 
4 end
5 Initialize module scores using historical averages
6 Initialize selected chain  $S \leftarrow []$ , remaining modules  $R \leftarrow \mathcal{M}$ 
7 for  $t = 1$  to  $L$  do
8   | Sample top- $k$  candidates from  $R$  using softmax over scores
9   | foreach candidate  $m \in \text{candidates}$  do
10  |   | Evaluate
11  |   |    $C' = S + [m, \text{final\_answer}]$ 
12  |   |   over  $N$  trials
13  |   |   Compute  $\text{reward}(m)$  using Eq. (1)
14  |   |   Update score of  $m$  with smoothing
15  |   |   if  $C'$  is correct in all trials then
16  |   |   | return  $C'$ 
17  |   |   end
18  |   | end
19  |   | Select best  $m^*$  and append to  $S$ 
20  |   | Remove  $m^*$  from  $R$ 
21 end
22 return  $S + [\text{final\_answer}]$ 
```

- **Stability:** The proportion of correct predictions across repeated executions;
- **Baseline gain:** Improvement over a baseline success rate (e.g., from a no-module chain);
- **Penalty:** A fixed negative score applied if all trials fail.

This reward design favors chains that are both correct and reliable across trials, while explicitly attributing credit to structural compositions that outperform the baseline. As a result, PPO optimizes the composer toward a generalizable composition policy that improves not only average accuracy but also consistency.

The policy π_θ and value network V_ϕ share a common encoder and are implemented as lightweight Transformer decoders.

Once a reasoning chain is predicted, the model can generate the final answer under its guidance, as

detailed in §3.4.

3.4 Reasoning Execution

Given a predicted reasoning chain, our system executes each module in sequence to derive the final answer. Each module corresponds to a cognitive operation and is associated with a functional prompt template that guides the MLLM’s behavior.

Execution proceeds step-by-step. At each step t , the system constructs a module-specific prompt by combining: (1) the original question, (2) the current visual input or intermediate result, and (3) the instruction template associated with the module m_{i_t} . This prompt is fed into the MLLM, whose response is used to update the intermediate context for the next step. The process continues until the chain reaches the `final_answer` module, whose output is taken as the final prediction.

4 Experiments

We conduct comprehensive experiments to validate the effectiveness of our compositional reasoning framework. Specifically, we aim to answer the following questions:

- (1) Does our approach generate more effective reasoning chains compared to baseline methods?
- (2) Can the learned structural patterns generalize across diverse tasks and domains?
- (3) How does policy optimization affect reasoning chain quality and overall accuracy?

4.1 Experiment Settings

4.1.1 Datasets

We evaluate on three types of benchmarks.

- **Mathematical reasoning:** MathVista (Lu et al., 2024), MathVerse (Zhang et al., 2024b), MathVision (Wang et al., 2024a), and **WeMath** (Qiao et al., 2024), covering geometry problems solving, diagram understanding, and calculation derivation.
- **Cross-domain QA:** MMStar (Chen et al., 2024a), ChartQA (Masry et al., 2022), and MMMU-Pro (Standard) (Yue et al., 2025), covering science, medicine, and data visualization tasks.
- **Modality-specific generalization:** GPQA (Rein et al., 2023) (text-only) and MMMU-Pro (Vision) (Yue et al., 2025), targeting unimodal reasoning in text and vision.

Model	MathVerse → Other Math Datasets										
	MathVision					MathVista					We-Math
	ALL	ARI	ALG	SolG	AnaG	ALL	GPS	GEO	SCI	STA	testmini
<i>General MLLMs</i>											
Gemini-1.5-Pro (Team et al., 2024)	19.20	34.30	20.30	18.90	35.70	63.90	81.70	–	–	–	66.90
GPT-4o (OpenAI et al., 2024)	30.39	49.30	42.00	29.10	32.30	63.80	64.70	–	–	–	62.80
GPT-4v (Yang et al., 2023)	22.76	35.70	27.30	23.80	32.10	49.90	50.50	51.00	63.10	55.80	51.43
InternVL2.5-8B (Chen et al., 2024b)	19.70	29.30	15.10	13.90	23.80	64.40	64.90	–	–	–	44.70
SPHINX (V2)-13B (Lin et al., 2023)	9.70	12.90	6.70	8.20	7.10	36.70	16.40	17.60	43.40	51.50	–
InternLM-XC2-VL-7B (Dong et al., 2024b)	14.54	12.10	9.30	11.90	15.50	57.60	63.00	59.00	62.30	64.10	30.95
<i>Math-specialized Models</i>											
Math-LLaVA-13B (Shi et al., 2024)	15.69	15.70	9.00	9.40	20.20	46.60	57.70	56.50	49.20	43.90	31.30
MathCoder-VL-8B (Wang et al., 2023)	26.10	26.40	18.60	23.00	32.10	54.80	73.60	–	–	–	58.70
MAVIS-7B (Zhang et al., 2024c)	18.50	21.50	17.50	15.00	19.50	29.10	64.10	–	–	–	44.30
Math-PUMA-7B (Zhuang et al., 2024)	14.00	21.10	5.00	17.80	21.10	47.90	48.10	47.30	42.60	55.80	41.00
MultiMath-7B (Peng et al., 2024)	16.30	15.50	11.30	13.30	21.10	50.00	66.80	64.90	42.60	49.20	42.20
URSA-8B (Luo et al., 2025)	31.50	29.30	30.10	37.70	28.60	59.80	83.20	77.40	58.20	57.10	60.70
<i>Recent Related Baselines</i>											
AStar (Qwen2-VL-7B) (Wu et al., 2025)	25.20	–	–	–	–	63.50	–	71.70	48.20	68.80	–
AtomThink (EMOVA)-8B (Xiang et al., 2025)	24.90	–	–	–	–	53.30	75.90	70.20*	50.40*	57.70*	49.30
Qwen2.5-VL-7B	25.10	31.43	26.09	22.95	28.57	61.70	61.06	61.51	60.66	74.09	57.50
Zero-shot CoT-7B (Kojima et al., 2023)	25.92	37.14	26.96	20.90	32.14	63.30	64.42	64.02	62.30	75.75	57.93
Ours-7B	28.88	37.14	27.54	27.05	34.52	64.40	68.75	68.20	62.30	76.08	59.40
Gain (↑)	+2.96	+0.00	+0.58	+6.15	+2.38	+1.10	+4.33	+4.18	+0.00	+0.33	+1.47
DeepSeek-VL2-27B (4.5B)	17.66	16.43	12.17	15.98	19.05	58.10	57.69	58.58	59.84	61.79	43.56
Zero-shot CoT-27B (4.5B)	15.86	15.71	11.88	10.66	19.05	58.30	52.40	53.97	62.30	66.45	44.31
Ours-27B (4.5B)	18.38	20.71	16.52	18.44	21.43	61.20	62.98	63.18	66.39	65.45	44.89
Gain (↑)	+2.52	+5.00	+4.64	+7.78	+1.12	+2.90	+10.58	+9.21	+4.09	-1.00	+0.58

Table 2: Cross-dataset generalization from MathVerse to other math benchmarks. We report performance on MathVision and MathVista subsets, and We-Math testmini. * indicates values approximated from bar charts in prior work. Notation like 27B(4.5B) denotes total parameters and active parameters during inference, respectively.

4.1.2 Training Setup

We train the structure composer on 2,400 samples selected from each of **MathVerse** and **MathVision**, with reasoning chains generated via our feedback-guided search. Composer training is conducted separately on the two datasets.

We use **Qwen2.5-VL-7B** (Bai et al., 2025) and **DeepSeek-VL2** (Wu et al., 2024) as frozen backbones, serving for execution. All models are trained on 2 A100 GPUs (40GB).

4.2 Main Results

4.2.1 Generalization across datasets

To evaluate the cross-dataset generalization ability of our proposed framework, we train the structure composer on the MathVerse dataset and evaluate it on MathVision, MathVista, and We-Math.

As shown in Table 2, our method achieves substantial improvements over both vanilla backbones and the Zero-shot CoT baseline. For example, on Qwen2.5-VL, accuracy on MathVista improves from 61.70% to 64.40%, while MathVision rises

from 25.10% to 28.88%. Similar trends hold for DeepSeek-VL2, where MathVista reaches 61.2% (+3.1%) and MathVision improves by +0.72%, demonstrating robust cross-dataset generalization. Notably, our method even surpasses proprietary systems like GPT-4o (63.8%) and Gemini-1.5-Pro (63.9%) on MathVista, and surpasses most math-specialized models on all datasets.

The largest improvements occur in Geometry. On Deepseek-VL2, Geometry Problem Solving (GPS) rises by 10.58% and Geometry Reasoning (GEO) by 9.12%, while on Qwen2.5-VL, they also see notable gains (4.33% and 4.18%). Thus, we hypothesize that our approach is especially effective for tasks requiring strong vision–language integration and multi-step reasoning.

These results demonstrate that structured reasoning chains can enhance MLLMs’ performance within the mathematical domain. By learning to compose reusable reasoning modules, our framework exhibits structural patterns that appear to generalize across heterogeneous math benchmarks.

Model	MMStar	MMM-U-Pro	ChartQA
<i>MathVerse as Source Dataset</i>			
Qwen2.5-VL-7B	63.90	38.30	80.88
Zero-shot CoT	61.20	39.60	76.72
Ours (Qwen2.5-VL)	66.23	45.41	82.16
<i>Gain</i> (↑)	+2.33	+7.11	+1.28
DeepSeek-VL2	53.67	33.18	83.20
Zero-shot CoT	49.07	32.77	81.84
Ours (DeepSeek-VL2)	59.20	35.26	84.72
<i>Gain</i> (↑)	+5.53	+2.08	+1.52
<i>MathVision as Source Dataset</i>			
Qwen2.5-VL-7B	63.90	38.30	80.88
Ours (Qwen2.5-VL)	66.63	44.14	83.44
<i>Gain</i> (↑)	+2.73	+5.84	+2.56
<i>Proprietary Models</i>			
Claude-3.5-Sonnet	65.1	51.5	90.8
GPT-4o	64.7	51.9	86.7
<i>Open-source Models</i>			
Qwen2.5-VL-72B	70.8	46.2	89.5
InternVL2.5-78B	69.5	48.6	88.3
InternVL2.5-8B	62.8	34.3	84.8
LLaVA-OneVision-7B	62.7	24.1	80.0

Table 3: Cross-domain generalization accuracy (%) on MMStar, MMMU-Pro, and ChartQA. Gains are computed over the vanilla backbone.

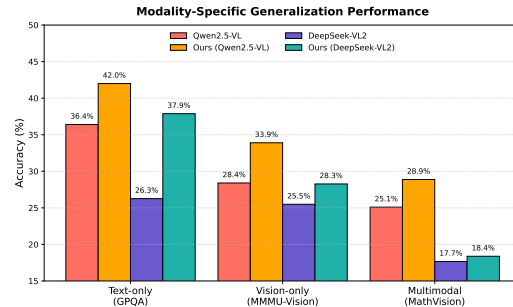
4.2.2 Generalization across domains

To assess cross-domain generalization, we train separately on math datasets (**MathVerse** and **Math-Vision**) and evaluate on three benchmarks—MMStar, MMMU-Pro, and ChartQA.

As shown in Table 3, our method achieves strong cross-domain performance across both training settings. The model reaches 66.23% on MM-Star, 45.41% on MMMU-Pro, and 82.16% on ChartQA. When trained instead on MathVision, performance remains competitive, demonstrating consistent generalization trends across different training sources. In both cases, our models outperform several strong open-source baselines such as InternVL2.5-8B and LLaVA-One-Vision, and approach the performance of much larger models like Qwen2.5-VL-72B—all without any fine-tuning on the target tasks.

These findings suggest that our structured reasoning framework enables broad domain generalization by focusing on abstract reasoning structure rather than problem-specific content. The consistent cross-domain performance from both MathVerse- and MathVision-trained models supports our hypothesis that the compositional patterns of reasoning modules capture general cognitive structures, forming a generalizable foundation for multimodal reasoning.

4.2.3 Generalization under modality-specific settings



(a) Accuracy on text-only (GPQA), vision-only (MMM-U-Vision), and multimodal (MathVista) tasks.

Compared To	Text	Vision	Multi
Qwen2.5-VL	+5.6	+5.5	+3.8
DeepSeek-VL2	+11.6	+2.8	+3.2

(b) Accuracy gains (%) of our method compared to baselines.

Figure 3: Modality-specific generalization comparison.

We further evaluate the robustness of our framework by testing it on tasks that isolate specific modality demands. Specifically, we consider three settings: text-only (GPQA), vision-only (MMM-U-Pro (Vision)), and multimodal (MathVista).

As shown in Figure 3, our model surpasses Deepseek-VL2 and Qwen2.5-VL-7B by 11.6% and 5.6% on GPQA, and by 2.8% and 5.5% on MMMU-Vision, respectively. Importantly, GPQA consists of PhD-level science problems that demand extensive reasoning, further validating the effectiveness of our approach in complex reasoning tasks.

These results suggest that our structured reasoning framework generalizes across modality-specific settings and remains competitive even when only a single modality is provided.

4.3 Training Strategy

4.3.1 Training Dynamics

Figure 4 illustrates the relationship between reward signals and task accuracy during optimization. As the average reward rises from negative to positive, GPQA accuracy improves by +2.5% and MathVista by +4.0%, confirming that reinforcement signals translate into tangible reasoning gains. This trend validates the stability of our policy optimization and suggests that structural feedback effectively aligns module composition with task objectives.

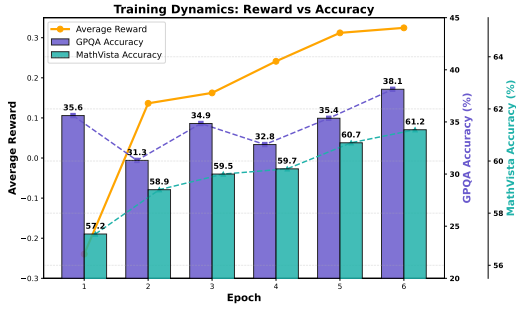


Figure 4: Training Dynamics: Average reward and accuracy trends during policy optimization.

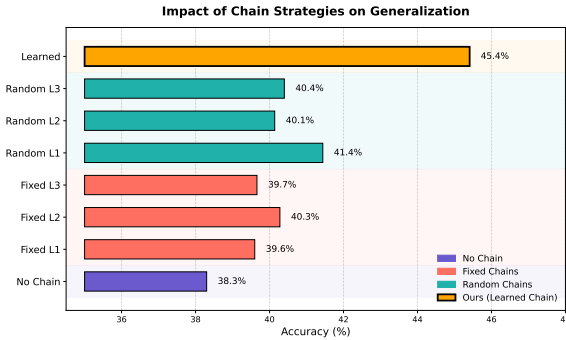


Figure 5: Comparison of chain strategies. Each method’s accuracy is shown, grouped by chain type (fixed/random/learned). Test on dataset MMMU-PRO. The backbone is Qwen2.5-VL-7B.

4.3.2 Comparing Chain Generation Strategies

To assess the importance of structured reasoning, we compare our model against variants with different chain composition strategies: (i) fixed-length chains, (ii) random-length chains, and (iii) a no-chain baseline. All settings use the same module set, and only the composition differs.

As shown in Figure 5, fixed and random chains yield only marginal improvements over the no-chain baseline (38.3%), with peak accuracies of 40.3% and 41.4%, respectively. This indicates that simply increasing chain length without meaningful structure offers limited benefits. In contrast, our learned structured chains achieve 45.4% accuracy—outperforming the best alternatives by over 4%—despite having a similar average length (2.1 steps). The improvement stems from adaptive module selection and ordering, not chain depth. These results support our central claim: structure matters—but only when it is learned, not imposed.

4.3.3 Comparing Training Strategies

Table 4 compares four training pipelines with the same budget (2,400 examples). Ours optimizes the

Method	Math-V	MathVista	MMStar
baseline	25.10	61.70	63.90
SFT only	25.90	59.30	61.20
SFT + PPO	26.84	63.00	63.97
PPO (Ours)	28.88	64.40	66.23

Table 4: Comparison of training strategies across diverse tasks. Accuracy (%) is reported on both in-distribution and generalization benchmarks.

structure policy via PPO, with a chain-level warm-start from greedy-search trajectories and stability signals. This initialization is *not* token-level SFT or imitation. We include two **SFT baselines** (SFT, SFT+PPO) solely for comparison. Details are in Appendix A.2.6.

Table 4 shows that PPO-only achieves the highest accuracy across out-of-distribution benchmarks, suggesting that direct reward optimization is more effective for aligning reasoning structure with task success. In contrast, SFT-based strategies exhibit limited gains, which we attribute to the noise and bias in pseudo-chains. This highlights the difficulty of relying solely on imitation learning when optimal structures are not guaranteed.

Moreover, PPO-only removes the need for costly annotation and reliance on large teacher LLMs, making it more scalable for structured reasoning.

5 Conclusion

To enhance generalizability in structured reasoning, we propose a compositional reasoning framework that assembles interpretable modules into reasoning chains. Central to our approach is a structure composer trained with a two-stage strategy: feedback-guided greedy search generates high-quality pseudo-labeled chains for warm-up, and PPO-based structure policy learning further refines the composition policy under task feedback, yielding a more flexible chain composer.

Extensive experiments demonstrate strong zero-shot generalization across diverse multimodal benchmarks. Notably, a composer trained solely on mathematical tasks transfers effectively to a broad spectrum of unseen evaluations, spanning both other math datasets and general-domain QA. These results suggest that it is not only the module inventory but, more importantly, the compositional organization of reasoning that exhibits cross-task regularities. Consequently, learning a transferable composition policy can serve as a practical path toward generalizable structured reasoning.

515 Limitations

516 Although our method demonstrates strong general-
517 ization across multiple benchmarks, it has several
518 limitations. First, the current set of compositional
519 reasoning modules is manually designed and may
520 not capture all reasoning patterns required for di-
521 verse tasks.

522 Second, explicit reasoning chains, while en-
523 hancing transparency and controllability, introduce
524 a potential trade-off between accuracy and effi-
525 ciency. We provide a detailed analysis and quan-
526 titative comparisons in the Appendix A.3.4. To
527 mitigate the latency, we have employed two strate-
528 gies: (i) hard-capping the chain length to at most
529 three modules, ensuring the reasoning process re-
530 mains concise, and (ii) adaptive chain length allo-
531 cation, where the composer learns to shorten rea-
532 soning for simple questions, often reducing to a
533 single final_answer step while allocating addi-
534 tional steps only when necessary for complex prob-
535 lems. Despite these measures, exploring efficiency
536 control remains an open challenge.

537 Third, PPO training for structure composition
538 can be constrained by relatively sparse task-level
539 feedback, since obtaining finer-grained rewards typ-
540 ically requires costly step-level annotation. In our
541 implementation, we partially mitigate this issue
542 by running three rollouts per sample, using aver-
543 age accuracy together with auxiliary rewards and
544 an explicit failure penalty to reduce variance and
545 stabilize updates, and we observe consistent perfor-
546 mance across runs. Nonetheless, developing finer
547 and more informative reward signals without addi-
548 tional supervision remains an important direction
549 for future work.

550 References

551 Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and
552 Dan Klein. 2017. *Neural module networks*. *Preprint*,
553 arXiv:1511.02799.

554 Anthropic. 2024. Introducing claude 3.5 son-
555 net. <https://www.anthropic.com/news/claude-3-5-sonnet>.

557 Shuai Bai, Keqin Chen, Xuejing Liu, and 1 others.
558 2025. *Qwen2.5-VL technical report*. *Preprint*,
559 arXiv:2502.13923.

560 Pierre Barrouillet. 2011. *Dual-process theories and
561 cognitive development: Advances and challenges*.
562 *Developmental Review*, 31(2-3):79–85.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang
Zang, Zehui Chen, Haodong Duan, Jiaqi Wang,
Yu Qiao, Dahua Lin, and Feng Zhao. 2024a. *Are we
on the right way for evaluating large vision-language
models?* *Preprint*, arXiv:2403.20330. 563
564
565
566
567

Zhe Chen, Weiyun Wang, Yue Cao, and 1 others. 2024b. *InternVL2.5: Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling*. 568
569
570
571

Zhenfang Chen, Qinhong Zhou, Yikang Shen, Yining
Hong, Hao Zhang, and Chuang Gan. 2023. *See,
think, confirm: Interactive prompting between vision
and language models for knowledge-based visual
reasoning*. *Preprint*, arXiv:2301.05226. 572
573
574
575
576

DeepSeek-AI, Daya Guo, Dejian Yang, and 1 others.
2025. *DeepSeek-R1: Incentivizing reasoning capa-
bility in LLMs via reinforcement learning*. *Preprint*,
arXiv:2501.12948. 577
578
579
580

Guanting Dong, Chenghao Zhang, Mengjie Deng, Yu-
tao Zhu, Zhicheng Dou, and Ji-Rong Wen. 2024a. *Progressive multimodal reasoning via active retrieval*. *Preprint*, arXiv:2412.14835. 581
582
583
584

Xiaoyi Dong, Pan Zhang, Yuhang Zang, and 1 oth-
ers. 2024b. *InternLM-XComposer2: Mastering
free-form text-image composition and comprehen-
sion in vision-language large model*. *Preprint*,
arXiv:2401.16420. 585
586
587
588
589

Timin Gao, Peixian Chen, Mengdan Zhang, Chaoyou
Fu, Yunhang Shen, Yan Zhang, Shengchuan Zhang,
Xiawu Zheng, Xing Sun, Liujuan Cao, and Rongrong
Ji. 2024. *Cantor: Inspiring multimodal chain-of-
thought of MLLM*. *Preprint*, arXiv:2404.16033. 590
591
592
593
594

David Klahr and Kevin Dunbar. 1988. *Dual space
search during scientific reasoning*. *Cognitive Science*,
12(1):1–48. 595
596
597

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yu-
taka Matsuo, and Yusuke Iwasawa. 2023. *Large
language models are zero-shot reasoners*. *Preprint*,
arXiv:2205.11916. 598
599
600
601

Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenen-
baum, and Samuel J. Gershman. 2016. *Building
machines that learn and think like people*. *Preprint*,
arXiv:1604.00289. 602
603
604
605

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng
Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yan-
wei Li, Ziwei Liu, and Chunyuan Li. 2024. *LLaVA-
OneVision: Easy visual task transfer*. *Preprint*,
arXiv:2408.03326. 606
607
608
609
610

Ziyi Lin, Chris Liu, Renrui Zhang, Peng Gao, Longtian
Qiu, Han Xiao, Han Qiu, Chen Lin, Wenqi Shao,
Keqin Chen, Jiaming Han, Siyuan Huang, Yichi
Zhang, Xuming He, Hongsheng Li, and Yu Qiao.
2023. *SPHINX: The joint mixing of weights, tasks,
and visual embeddings for multi-modal large lan-
guage models*. *Preprint*, arXiv:2311.07575. 611
612
613
614
615
616
617

618	Ziyu Liu, Zeyi Sun, Yuhang Zang, Xiaoyi Dong, Yuhang Cao, Haodong Duan, Dahua Lin, and Jiaqi Wang. 2025. Visual-RFT: Visual reinforcement fine-tuning . <i>Preprint</i> , arXiv:2503.01785.	672
619		673
620		674
621		
622	Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. 2024. MathVista: Evaluating mathematical reasoning of foundation models in visual contexts . <i>Preprint</i> , arXiv:2310.02255.	675
623		676
624		677
625		
626		678
627		679
628	Liangchen Luo, Yinxiao Liu, Rosanne Liu, and 1 others. 2024. Improve mathematical reasoning in language models by automated process supervision . <i>Preprint</i> , arXiv:2406.06592.	680
629		681
630		
631		
632	Ruilin Luo, Zhuofan Zheng, Yifan Wang, and 1 others. 2025. URSA: Understanding and verifying chain-of-thought reasoning in multimodal mathematics . <i>Preprint</i> , arXiv:2501.04686.	682
633		683
634		684
635		685
636	Xinyin Ma, Guangnian Wan, Runpeng Yu, Gongfan Fang, and Xinchao Wang. 2025. CoT-valve: Length-compressible chain-of-thought tuning . <i>Preprint</i> , arXiv:2502.09601.	686
637		687
638		688
639		689
640	Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. 2022. ChartQA: A benchmark for question answering about charts with visual and logical reasoning . <i>Preprint</i> , arXiv:2203.10244.	690
641		691
642		692
643		693
644	Chancharik Mitra, Brandon Huang, Trevor Darrell, and Roei Herzig. 2024. Compositional chain-of-thought prompting for large multimodal models . <i>Preprint</i> , arXiv:2311.17076.	694
645		695
646		696
647		
648	OpenAI, Aaron Hurst, Adam Lerer, and 1 others. 2024. GPT-4o system card . <i>Preprint</i> , arXiv:2410.21276.	697
649		698
650	Shuai Peng, Di Fu, Liangcai Gao, Xiuqin Zhong, Hongguang Fu, and Zhi Tang. 2024. MultiMath: Bridging visual and mathematical reasoning for large language models . <i>Preprint</i> , arXiv:2409.00147.	699
651		700
652		
653		
654	Runqi Qiao, Qiuna Tan, Guanting Dong, and 1 others. 2024. We-math: Does your large multimodal model achieve human-like mathematical reasoning? <i>Preprint</i> , arXiv:2407.01284.	701
655		702
656		703
657		704
658	David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. GPQA: A graduate-level google-proof Q&a benchmark . <i>Preprint</i> , arXiv:2311.12022.	705
659		706
660		707
661		708
662		709
663	John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms . <i>Preprint</i> , arXiv:1707.06347.	710
664		711
665		712
666		713
667	Wenhao Shi, Zhiqiang Hu, Yi Bin, Junhua Liu, Yang Yang, See-Kiong Ng, Lidong Bing, and Roy Ka-Wei Lee. 2024. Math-LLaVA: Bootstrapping mathematical reasoning for multimodal large language models . <i>Preprint</i> , arXiv:2406.17294.	714
668		715
669		716
670		
671		
	Elizabeth S. Spelke and Katherine D. Kinzler. 2007. Core knowledge . <i>Developmental Science</i> , 10(1):89–96.	717
		718
		719
		720
	Jiajin Tang, Ge Zheng, Jingyi Yu, and Sibeil Yang. 2023. CoTDet: Affordance knowledge prompting for task driven object detection . <i>Preprint</i> , arXiv:2309.01093.	721
		722
		723
		724
	Gemini Team, Petko Georgiev, Ving Ian Lei, and 1 others. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context . <i>Preprint</i> , arXiv:2403.05530.	
	Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Mingjie Zhan, and Hongsheng Li. 2024a. Measuring multimodal mathematical reasoning with MATH-vision dataset . <i>Preprint</i> , arXiv:2402.14804.	
	Ke Wang, Houxing Ren, Aojun Zhou, Zimu Lu, Sichun Luo, Weikang Shi, Renrui Zhang, Linqi Song, Mingjie Zhan, and Hongsheng Li. 2023. MathCoder: Seamless code integration in LLMs for enhanced mathematical reasoning . <i>Preprint</i> , arXiv:2310.03731.	
	Peiyi Wang, Lei Li, Zhihong Shao, R. X. Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. 2024b. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations . <i>Preprint</i> , arXiv:2312.08935.	
	Yaoting Wang, Shengqiong Wu, Yuecheng Zhang, William Wang, Ziwei Liu, Jiebo Luo, and Hao Fei. 2025. Multimodal chain-of-thought reasoning: A comprehensive survey . <i>Preprint</i> , arXiv:2503.12605.	
	Jinyang Wu, Mingkuan Feng, Shuai Zhang, Ruihan Jin, Feihu Che, Zengqi Wen, and Jianhua Tao. 2025. Boosting multimodal reasoning with MCTS-automated structured thinking . <i>Preprint</i> , arXiv:2502.02339.	
	Zhiyu Wu, Xiaokang Chen, Zizheng Pan, and 1 others. 2024. DeepSeek-VL2: Mixture-of-experts vision-language models for advanced multimodal understanding . <i>Preprint</i> , arXiv:2412.10302.	
	Kun Xiang, Zhili Liu, Zihao Jiang, Yunshuang Nie, Kaixin Cai, Yiyang Yin, Runhui Huang, Haoxiang Fan, Hanhui Li, Weiran Huang, Yihan Zeng, Yu-Jie Yuan, Jianhua Han, Lanqing Hong, Hang Xu, and Xiaodan Liang. 2025. Can atomic step decomposition enhance the self-structured reasoning of multimodal large models? <i>Preprint</i> , arXiv:2503.06252.	
	Guowei Xu, Peng Jin, Hao Li, Yibing Song, Lichao Sun, and Li Yuan. 2025. LLaVA-CoT: Let vision language models reason step-by-step . <i>Preprint</i> , arXiv:2411.10440.	
	Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. 2023. The dawn of LMMs: Preliminary explorations with GPT-4V(ision) . <i>Preprint</i> , arXiv:2309.17421.	

725	Xiang Yue, Tianyu Zheng, Yuansheng Ni, Yubo Wang,	A.3.6 Domain-Wise Accuracy Comparison	775
726	Kai Zhang, Shengbang Tong, Yuxuan Sun, Botao Yu,	A.3.7 Cross-Model Generalization	776
727	Ge Zhang, Huan Sun, Yu Su, Wenhui Chen, and Gra-		
728	ham Neubig. 2025. <i>MMMU-pro: A more robust</i>		
729	<i>multi-discipline multimodal understanding bench-</i>	A.4 Module Ablation Study	777
730	<i>mark. Preprint, arXiv:2409.02813.</i>		
731	Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue,	A.5 Case Study	778
732	Yuxiao Dong, and Jie Tang. 2024a. <i>ReST-MCTS*:</i>	A.5.1 Math Reasoning	779
733	<i>LLM self-training via process reward guided tree</i>	A.5.2 General Question Answering	780
734	<i>search. Preprint, arXiv:2406.03816.</i>		
735	Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun	A.6 Extended Discussion	781
736	Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu,	A.6.1 Comparison with MCTS and SFT-Based	782
737	Kai-Wei Chang, Peng Gao, and Hongsheng Li. 2024b.	Methods	783
738	<i>MathVerse: Does your multi-modal LLM truly see</i>	A.6.2 Why Do Compositional Reasoning	784
739	<i>the diagrams in visual math problems? Preprint,</i>	Chains Generalize?	785
740	<i>arXiv:2403.14624.</i>	A.6.3 Why Is Reasoning Particularly Effective	786
741	Renrui Zhang, Xinyu Wei, Dongzhi Jiang, Ziyu Guo,	in Math Tasks?	787
742	Shicheng Li, Yichi Zhang, Chengzhuo Tong, Ji-	A.6.4 Understanding Remaining Errors	788
743	aming Liu, Aojun Zhou, Bin Wei, Shanghang		
744	Zhang, Peng Gao, Chunyuan Li, and Hongsheng	A.7 Future Work	789
745	Li. 2024c. <i>MAVIS: Mathematical visual instruc-</i>		
746	<i>tion tuning with an automatic data engine. Preprint,</i>	A.8 Use of AI Assistants	790
747	<i>arXiv:2407.08739.</i>		
748	Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao,	A.1 Module Prompt Templates	791
749	George Karypis, and Alex Smola. 2024d. <i>MM-CoT:</i>	We list the prompt templates used for each reason-	792
750	<i>Multimodal chain-of-thought reasoning in language</i>	ing module. Each template instructs the model to	793
751	<i>models. Preprint, arXiv:2302.00923.</i>	perform a distinct reasoning operation.	794
752	Hengguang Zhou, Xirui Li, Ruochen Wang, Minhao		
753	Cheng, Tianyi Zhou, and Cho-Jui Hsieh. 2025. <i>R1-</i>		
754	<i>zero's "aha moment" in visual reasoning on a 2B</i>		
755	<i>non-SFT model. Preprint, arXiv:2503.05132.</i>		
756	Wenwen Zhuang, Xin Huang, Xiantao Zhang, and Jin		
757	Zeng. 2024. <i>Math-PUMA: Progressive upward multi-</i>		
758	<i>modal alignment to enhance mathematical reasoning.</i>		
759	<i>Preprint, arXiv:2408.08640.</i>		
760	A Appendix		
761	A.1 Module Prompt Templates		
762	A.2 Implementation & Model Details		
763	A.2.1 Architecture overview		
764	A.2.2 Hyperparameter settings		
765	A.2.3 Inference Strategy		
766	A.2.4 MLLM Settings		
767	A.2.5 Accuracy Evaluation Method		796
768	A.2.6 SFT data synthesis		
769	A.3 Additional Experiments		
770	A.3.1 Datasets		
771	A.3.2 Baselines		
772	A.3.3 Warm-start ablation		
773	A.3.4 Efficiency and resource cost		
774	A.3.5 Cross-Domain Accuracy Comparison		797

Module: visual

Solve the problem by analyzing the image. Focus on extracting spatial layout, objects, quantities, or symbolic information from the visual input.

Module: physics

Solve the problem by applying relevant domain laws. Retrieve domain knowledge such as formulas, constraints, or relationships.

Module: logic

Solve the problem through step-by-step logical reasoning. Derive intermediate values or decisions in a serial, rule-based fashion. Use this method when a chain of deterministic computation or decision-making is required.

Module: reflect

Improve the solution by reviewing previous reasoning steps. Identify possible mistakes, inconsistencies, or unnecessary detours. Use this method after a solution attempt to refine or validate.

Module: problem_decompose

Tackle the problem by breaking it down into smaller sub-problems. Each subgoal should be simpler and sequentially solvable. Use this method when the original question is complex or multi-stage.

Module: cross_modal_link

Solve the problem by connecting visual cues to linguistic or world knowledge. Relate what is seen to what is known. Use this method when reasoning requires bridging modalities or invoking common-sense.

Module: hypothesis

Solve the problem by proposing plausible assumptions or interpretations. Consider multiple explanations if the context is ambiguous. Use this method when exact reasoning is hard and assumption testing is needed.

Module: validate

Test the correctness of prior reasoning using visual evidence or logic. Accept or reject a previous hypothesis or step. Use this method to verify assumptions or computed results.

Module: final_answer

Provide the final answer in the following format: [Answer : X], where X is the result.

A.2 Implementation & Model Details

We provide implementation details of our **Structure Composer**, a sequence prediction model designed to generate symbolic reasoning chains based on multimodal inputs.

A.2.1 Architecture Overview

The Multimodal Composer accepts a pair of question text and associated image features. It first encodes the text using a BERT-base encoder and projects image features into the same hidden space. A Transformer-based fusion encoder combines the text and image representations. The fused features are then projected into a decoder-specific embedding space and used as memory for a Transformer decoder. The decoder autoregressively predicts a sequence of module tokens that represent reasoning steps. To avoid redundant operations, step-wise decoding includes masking of repeated modules and optional enforcement of a minimum number of steps before allowing the <eos> token.

A.2.2 Hyperparameter Settings

The main hyperparameters are summarized in Table 5.

Table 5: MultimodalComposer architecture configuration.

Component	Setting
Text encoder	BERT-base (uncased)
Image input dim	512
Image projection dim	768
Fusion encoder	4-layer TransformerEncoder
Fusion FFN dim	1024
Decoder	16-layer TransformerDecoder
Decoder FFN dim	2048
Module vocab size	10
Module embedding dim	768
Max decoding steps	4
Dropout	0.2

A.2.3 Inference Strategy

We support both full-sequence generation and step-wise decoding with autoregressive sampling. The pseudocode of the standard inference process is shown in Algorithm 2.

A.2.4 MLLM Settings

We report the parameter configurations used for inference with **DeepSeek-VL2** and **Qwen2.5-VL** in Table 6. The key hyperparameters are listed below for reproducibility.

All other unspecified parameters were set to default values provided by the model. No additional prompt engineering was applied beyond task-specific templates.

A.2.5 Accuracy Evaluation Method

To evaluate model accuracy, we employ a two-stage scoring process. First, we use an automatic judg-

Algorithm 2: Multimodal Composer Inference

Input: Question x , image features v , max decoding steps T , start token $\langle s \rangle$
Output: Module sequence $[y_1, y_2, \dots, y_T]$

```
1  $h_t \leftarrow \text{BERT}(x)$ ; // Text encoding
2  $h_v \leftarrow \text{Proj}(v)$ ; // Image projection
3  $h \leftarrow \text{FusionEncoder}([h_t; h_v])$ ;
4  $m \leftarrow \text{FusionProj}(h)$   $y \leftarrow [\langle s \rangle]$ ;
   // Initialize decoder input
5 for  $t = 1$  to  $T$  do
6    $e_t \leftarrow \text{Embed}(y_{1:t})$ ;
7    $z_t \leftarrow \text{Decoder}(e_t, m)$ ;
8    $\hat{y}_{t+1} \leftarrow \arg \max \text{Softmax}(Wz_t)$ ;
9   if  $\hat{y}_{t+1} = \langle \text{eos} \rangle$  then
10    | return  $y$ 
11  end
12  Append  $\hat{y}_{t+1}$  to  $y$ ;
13 end
14 return  $y$ 
```

Table 6: Inference parameter settings for DeepSeek-VL2 and Qwen2.5-VL.

Parameter	DeepSeek-VL2	Qwen2.5-VL
model	deepseek-vl2	qwen2.5-vl
max_tokens	512	4096
temperature	0.2	0.2
top_p	0.9	0.9
top_k	40	40
frequency_penalty	1	1
n	1	1
response_format	text	text

ment model—Qwen2.5-7B-Instruct—to assess whether the predicted answer matches the ground-truth answer. The evaluation prompt is as follows:

Evaluation

Please determine whether the predicted answer is correct.
If the question includes a list of choices (e.g., "Choices:['0.8', '0.7', '0.1', ...]"), interpret them as multiple-choice options labeled from A to Z in order:
- 'A' corresponds to the first option (e.g., '0.8'),
- 'B' to the second (e.g., '0.7'),
- 'C' to the third (e.g., '0.1'), and so on, up to 'Z' if needed.
Question: {question}
Ground Truth Answer: {gt}
Predicted Answer: {pred}
Respond with only one number: 1 (correct) or 0 (incorrect).
Judgement:

Despite generally reliable performance, we observe that the auto-judger produces approximately 10% judgment error on sampled cases. To ensure evaluation fidelity, we manually verify all predictions on top of the automatic scoring. This correction step may lead to slight deviations from the original results reported in other works using unverified accuracy scores.

A.2.6 SFT Data Synthesis

This subsection describes the data construction pipeline used *only* for the SFT and SFT+PPO baselines in Table 4. For each training instance, we first generate a *pseudo* module chain \hat{c} (a discrete sequence of reasoning modules), which serves as a structural scaffold. We then prompt GPT-4o with the original input and the pseudo chain \hat{c} under a fixed, consistent template, asking it to (i) follow the module sequence, (ii) produce a complete natural-language chain-of-thought consistent with \hat{c} , and (iii) output the final answer in the required format. The resulting GPT-4o generations yield token-level supervision for SFT.

To reduce noise, we apply basic validity checks (e.g., parsable format, presence of termination, and length bounds) and discard failed or inconsistent generations.

SFT baseline configuration. For SFT and SFT+PPO, we fine-tune Qwen2.5-VL-7B with LoRA (rank $r=16$) using bf16 precision, AdamW, and a cosine learning-rate schedule. We train on the 2,400 synthesized examples for 3 epochs with batch size 16. LoRA is applied to the attention projections.

A.3 Additional Experiments**A.3.1 Datasets**

We evaluate our approach on multiple multimodal benchmarks to assess reasoning and generalization capabilities. Details are in Table 7.

A.3.2 Baselines

As shown in Table 8, we compare our approach with a range of proprietary and open-source multimodal models widely used in mathematical reasoning benchmarks.

A.3.3 Warm-Start Ablation

We ablate the proposed chain-level warm-start by training the same PPO optimizer with and without warm-start initialization. Warm-start uses greedy-search trajectories to provide a coarse prior over

Dataset	Domain	Key Features
MathVista (Lu et al., 2024)	Visual math reasoning	1000 questions; covers arithmetic, geometry, algebra; requires compositional reasoning.
MathVerse (Zhang et al., 2024b)	Visual math reasoning	Complex symbolic reasoning with structured visual input.
MathVision (Wang et al., 2024a)	Visual math reasoning	3,040 high-difficulty problems across 16 domains, five difficulty levels.
We-Math (Qiao et al., 2024)	Visual math reasoning	Visual-spatial reasoning and symbolic logic.
MMStar (Chen et al., 2024a)	General	QA on diagrams and structured data; domain transfer evaluation.
ChartQA (Masry et al., 2022)	Charts	Reasoning over bar, line, pie charts; requires text-image alignment.
MMMU-Pro (Yue et al., 2025)	General	Standard and Vision-only subsets; domain diversity and robustness tests.
GPQA (Rein et al., 2023)	Science	Graduate-level proof-based reasoning; language-only modality.

Table 7: Summary of datasets used for evaluation.

plausible structure chains, which stabilizes early policy learning. Importantly, it does *not* perform token-level imitation or supervised fine-tuning.

Figure 6 compares learning dynamics in terms of reward. Without warm-start, optimization converges substantially slower and exhibits higher variance, often collapsing to degenerate behaviors such as predicting STOP at the first step. In contrast, warm-start reduces early variance and prevents degenerate chains. Final performance is achieved by PPO optimization.

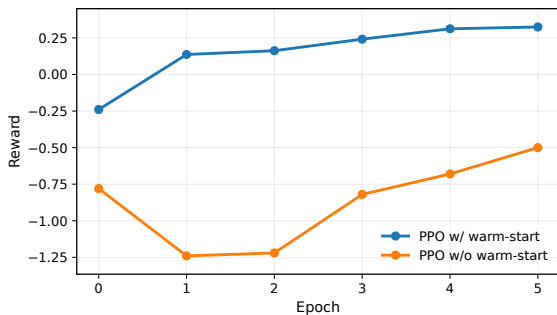


Figure 6: Warm-start ablation.

A.3.4 Efficiency and Resource Cost

Table 9 reports compute and efficiency under identical hardware and decoding settings. We separate the backbone parameters (Qwen2.5-VL) from the trainable parameters introduced by the structure composer.

Concretely, SFT+PPO incurs the highest resource footprint, increasing peak allocated memory to 52.16 GB and end-to-end latency to 2.18 s. In contrast, our PPO training keeps the adaptation lightweight: it uses only 12.89 M trainable parameters and 20.21 GB peak memory, which substantially lower than SFT (24.12 M / 36.18 GB) and SFT+PPO.

Our training time (2.12 h) is notably shorter than SFT+PPO (3.89 h), while the inference latency increases moderately over the frozen backbone (1.64 s) and stays below SFT+PPO. Overall, the added inference latency is modest relative to the accuracy gains, and our training remains efficient in both peak memory and trainable parameters.

A.3.5 Cross-Domain Accuracy Comparison

To further analyze generalization across diverse domains, we conduct experiments on the MMMU-Pro (Standard) benchmark, which contains questions from five major disciplines: Medicine, Engineering, Science, Humanities, and Business. Each discipline includes multiple fine-grained sub-domains (e.g., Clinical, Mechanical, Economics). We compare five methods: the original baseline (Qwen2.5-VL), a Zero-shot CoT variant, GPT-4o, Deepseek-VL2, and our proposed structured reasoning approach.

As shown in Figure 7, our method consistently achieves the highest accuracy across all five domains, particularly excelling in *Medicine* and *Engineering*, which demand multi-step reasoning and cross-modal understanding. Compared to Zero-shot CoT, our method demonstrates a notable performance boost by reducing redundant steps and improving reasoning controllability.

A.3.6 Domain-wise Accuracy Comparison

Figure 7 presents accuracy comparisons across five academic domains: *Medicine*, *Engineering*, *Science*, *Humanities*, and *Business*. We compare representative open-source and proprietary models, both with and without our structure reasoning enhancements.

Our method consistently outperforms baselines across all domains, with particularly large margins in **Medicine** and **Engineering**, where pre-

Model	Type	Description
Gemini-1.5-Pro (Team et al., 2024)	Proprietary	Google’s flagship multimodal model with long-context capability and SOTA reasoning across domains.
GPT-4V (Yang et al., 2023)	Proprietary	OpenAI’s vision-enabled GPT-4, strong performance on MathVista and visual reasoning tasks.
Claude-3.5-Sonnet(Anthropic, 2024)	Proprietary	Anthropic’s advanced multimodal model, competitive reasoning ability across benchmarks.
Qwen-2.5-VL (Bai et al., 2025)	Open-source	Alibaba’s open-weight multimodal LLM optimized for vision-language reasoning.
InternVL-2.5 (Chen et al., 2024b)	Open-source	High-performance vision-language model series with parameter scaling (3B–88B).
SPHINX(V2) (Lin et al., 2023)	Open-source	A joint-modality mixing model designed for robust reasoning and multi-domain tasks.
LLaVA-OneVision (Li et al., 2024)	Open-source	Lightweight open-source multimodal model focused on visual QA and instruction following.
LLaVA-CoT (Xu et al., 2025)	Open-source	Extends LLaVA with structured chain-of-thought prompting for visual reasoning.
Math-LLaVA (Shi et al., 2024)	Open-source	Fine-tuned LLaVA on MathV360K to enhance mathematical reasoning in multimodal contexts.
MathCoder (Wang et al., 2023)	Open-source	Integrates code-based symbolic reasoning with multimodal visual question answering.
MAVIS (Zhang et al., 2024c)	Open-source	Specialized for math reasoning across diagrams and structured visual contexts.
Math-PUMA (Zhuang et al., 2024)	Open-source	Progressive reasoning approach tailored for complex multimodal math problems.
MultiMath (Peng et al., 2024)	Open-source	Targets visual-textual alignment for advanced math problem solving.
URSA (Luo et al., 2025)	Open-source	Emphasizes solution verification and intermediate step validation in reasoning.
AStar (Wu et al., 2025)	Open-source	Employs MCTS and thought card for boosting multimodal reasoning accuracy.
AtomThink (Xiang et al., 2025)	Open-source	Atomic-step reasoning framework for structured and interpretable multimodal inference.

Table 8: Baseline models and their brief descriptions.

cise, multi-step reasoning is essential. Notably, the structural reasoning strategy improves performance even in abstract or less formal fields such as Humanities and Business, indicating its broad applicability.

These results underscore the versatility of our method across diverse task distributions and knowledge modalities.

A.3.7 Cross-Model Generalization

We further investigate whether a structure composer trained on one backbone model can generalize to others during inference. Specifically, we use the composer trained with Qwen2.5-VL as a fixed module selector, and apply it to two different vision-language models: GPT-4o and DeepSeek-VL2, without any further adaptation or retraining.

Table 10 shows accuracy on the MathVista benchmark. While our method brings noticeable gains when applied to Qwen2.5-VL (the model used for training the composer), it yields marginal or no improvement when directly used with GPT-4o or DeepSeek-VL2. This suggests that the learned module sequencing policy is likely cou-

pled with the specific reasoning patterns and latent spaces of the training backbone.

These results imply that while structured reasoning policies are beneficial, effective cross-model generalization may require additional model-specific adaptation techniques, such as prompt calibration, or lightweight finetuning.

A.3.8 Module Ablation Study

Motivation. Our design goal is to construct a *small, sufficient, and interpretable* basis of reasoning modules that can be composed to solve diverse multimodal reasoning problems. Importantly, our composer includes a dedicated STOP action, allowing the policy to terminate early once sufficient evidence has been gathered. Therefore, if a module were truly dispensable, an optimal learned policy would omit it by either never selecting it or by terminating earlier via STOP.

Protocol. We analyze the module set from two complementary perspectives. (1) **Usage analysis:** we report how frequently each module is selected by the learned composer across datasets, which

Method	Train Time (h)	Trainable Params (M)	Peak Mem (GB, alloc)	End-to-end Latency (s)
Qwen2.5-VL (base)	–	0.00	–	1.64
SFT (LoRA)	2.32	24.12	36.18	1.89
SFT+PPO	3.89	37.01	52.16	2.18
PPO (ours)	2.12	12.89	20.21	2.03

Table 9: Efficiency comparison on Qwen2.5-VL-7B under identical hardware and decoding settings. Trainable parameters exclude the frozen backbone. Peak memory is measured by `torch.cuda.max_memory_allocated()` (in GB) with a fixed measurement protocol. Latency is measured end-to-end per example (structure prediction + answer generation) using the same decoding configuration.

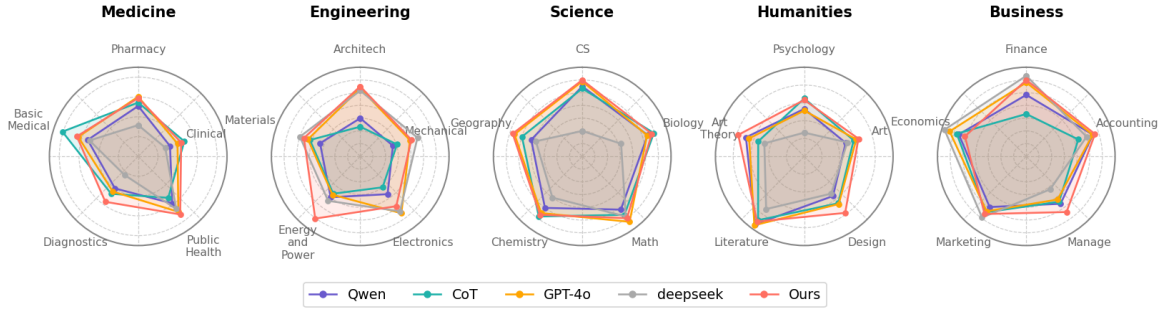


Figure 7: Cross-domain accuracy comparison on MMMU-Pro (Standard). Our method outperforms all baselines across diverse domains, achieving the most consistent gains in Medicine, Engineering, and Science.

Model	Baseline	+Composer (Ours)	Gain
Qwen2.5-VL	61.7	64.4	+2.7
GPT-4o	63.5	64.3	+0.8
DeepSeek-VL2	58.1	58.1	+0.0

Table 10: Cross-model generalization results on Math-Vista. Composer is trained on Qwen2.5-VL.

quantifies whether modules are actively utilized rather than being dead actions. (2) **Leave-one-out ablations:** we remove one module at a time from the action space and re-train the composer under the same configuration (data, backbone, training schedule, and rewards), and evaluate the final end-task accuracy. The performance drop indicates the marginal contribution of the removed module, revealing complementary utility among modules.

Module usage across datasets. We visualize the most frequent reasoning chains and their module-position statistics in math tasks (Figure 8) and in general QA tasks (Figure 9). These figures contain quantitative signals beyond qualitative examples: they reveal (a) which modules are selected frequently, (b) where they tend to appear along the chain, and (c) whether the policy collapses to trivial patterns (e.g., immediate STOP) or instead composes non-trivial sequences.

In math, the composer favors concise, logic-

driven chains. A dominant pattern is logic → visual → final_answer, commonly triggered by geometry and spatial deduction. Importantly, logic and visual not only dominate in frequency but also concentrate in earlier positions, consistent with an initial stage that establishes symbolic constraints and perceptual grounding before producing the final answer.

In contrast, general QA exhibits more diverse and semantically rich compositions. Chains frequently begin with hypothesis or reflect, such as hypothesis → reflect → validate → final_answer, which appear in tasks requiring analogical reasoning or explanatory justification. Notably, cross_modal_link appears more often and tends to occur later in the chain, suggesting that bridging textual and visual evidence is typically resolved after initial hypothesis formation and reflection, before final validation.

Leave-one-out module ablations. While chain-distribution visualizations indicate that modules are actively selected and composed in task-dependent ways, they do not by themselves establish necessity. We therefore perform leave-one-out ablations: we remove one module at a time from the action space and re-train the composer under the same configuration (data, backbone, training schedule, and rewards). In addition to end-task

accuracy, we also report the *average reasoning-chain length* (i.e., number of selected modules before STOP/final_answer), which helps disambiguate whether performance changes are merely due to early stopping or altered computation budget. Table 11 shows that removing any module consistently degrades accuracy, while the average length changes only moderately. This suggests the performance drop is not explained solely by longer/shorter chains, but by the *loss of a specific cognitive capability* that other modules (or STOP) cannot fully substitute.

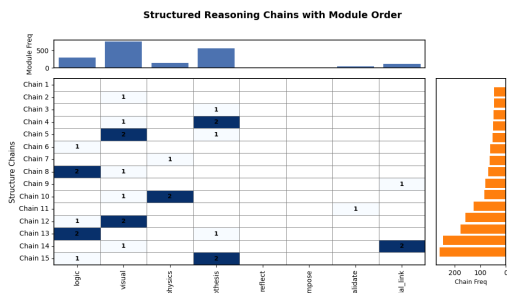


Figure 8: **Structured reasoning chain distribution on Math tasks.** The heatmap displays the top-15 most frequent module sequences used by the model on Math-Vision. Numbers indicate the position of each module within the chain (1: first, 2: second, etc.). Top and right bar plots show marginal usage frequencies for modules and entire chains, respectively.

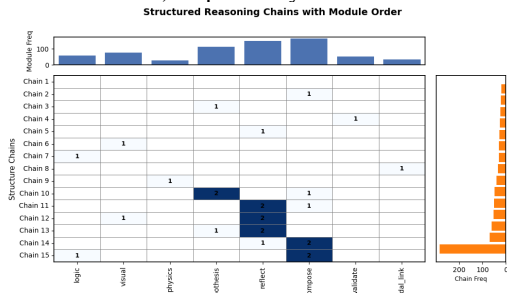


Figure 9: **Structured reasoning chain distribution on general QA tasks.** Compared to math tasks, the reasoning structure here exhibits higher diversity in both module types and chain compositions. Modules such as hypothesis, reflect, and cross_modal_link appear more frequently and in varied positions, reflecting semantic and multimodal inference needs.

Together, Figures 8 and 9 show that the composer learns stable, domain-sensitive composition patterns, and Table 11 confirms that these modules provide complementary, non-redundant utility rather than acting as interchangeable artifacts.

A.4 Case Study

To illustrate the interpretability and effectiveness of our approach, we provide case studies comparing three settings: the original baseline, Zero-shot CoT, and our method. We consider two categories of tasks: (1) **Math Reasoning**, which requires multi-step reasoning over visual and textual cues, and (2) **General QA**, involving attribute-based and contextual understanding. For each category, we show two representative examples based on different backbone models: Deepseek-VL2 and Qwen2.5-VL. In all cases, our method achieves correct answers with well-structured reasoning chains, whereas baselines often fail due to incomplete or incorrect reasoning steps.

A.4.1 Math Reasoning

Figures 10 and 11 present two math reasoning examples. Both require object identification and conditional subtraction. Our approach successfully performs structured reasoning and delivers accurate final answers, outperforming baselines.

A.4.2 General Question Answering

Figures 12 and 13 illustrate two general QA cases. These examples involve multi-attribute reasoning and cross-modal alignment. Our approach consistently provides interpretable reasoning chains and correct answers, while baselines and zero-shot CoT fail to maintain consistency.

A.5 Extended Discussion

This section reflects on the design principles and empirical behaviors of our atomic reasoning chain framework. We revisit why it generalizes, when it works best, and how it relates to broader trends in structured reasoning.

A.5.1 Comparison with MCTS and SFT-Based Methods

Two widely explored strategies for enhancing reasoning are Monte Carlo Tree Search (MCTS) and supervised fine-tuning (SFT). While both approaches improve baseline prompting, they exhibit limitations that our method addresses, as shown in Table 12.

MCTS. Tree-search-based methods expand candidate reasoning trajectories by simulating multiple decision branches and selecting the most promising path. Although this exhaustive search can theoretically identify high-quality solutions, it incurs prohibitive computational costs at inference time.

Metric	Leaved Module (leave-one-out)								
	None (Ours)	–Visual	–Physics	–Logic	–Reflect	–Decompose	–XModal-Link	–Hypothesis	–Validate
Acc (%) ↑	89.00	87.67	87.67	86.67	86.33	87.33	87.33	85.00	86.33
Avg. Length	1.93	1.99	2.14	2.06	2.04	2.05	1.96	2.01	2.10

Table 11: Leave-one-out module ablation results. **None (Ours)** denotes the full model. Each ablation removes one module from the action space and re-trains the composer with all other settings unchanged. We additionally report average chain length to control for changes in computation budget induced by STOP.

Method	Reasoning Cost	Generalization	Controllability	Needs Annotation?
MCTS	Very High	Moderate	High	No
SFT	Low	Low–Moderate	Low	Yes
Ours	Moderate	High	High	No

Table 12: Comparison of different approaches in terms of reasoning cost, generalization, controllability, and annotation dependency.

Multi-modal tasks with long reasoning horizons amplify this overhead, making real-time or large-scale deployment impractical. In contrast, our framework performs a single-pass generation of structured module chains, leveraging structural priors learned offline to achieve search-like benefits without the inference-time explosion in cost.

SFT. Supervised approaches rely on mimicking annotated or synthetic chains through next-token prediction. While effective for aligning with human-like rationales, this paradigm is inherently imitation-driven and tends to overfit domain-specific patterns. Moreover, linguistic entanglement often yields verbose or inconsistent reasoning steps. Our method diverges from pure imitation by optimizing structural policies via reinforcement learning, enabling controllable reasoning depth and improved generalization across unseen domains.

Overall, our framework combines the interpretability and structural rigor sought by MCTS while avoiding its computational bottlenecks, and surpasses SFT by learning reasoning structures rather than memorizing surface forms. This balance underpins its superior adaptability in cross-task and cross-modal scenarios.

A.5.2 Why Do Compositional Reasoning Chains Generalize?

Our results suggest that the core strength of reasoning module chains lies in their alignment with general-purpose cognitive functions. Each module encodes a task-agnostic reasoning operation—such as logic, visual, or validate—that can be reused across tasks. These primitives are not tied to any dataset-specific pattern, but instead reflect reusable building blocks of problem solving.

This modular abstraction promotes systematic generalization: rather than memorizing linguistic heuristics, the model learns to compose reasoning strategies from a shared pool. This behavior resonates with theories of compositionality in cognitive science (Spelke and Kinzler, 2007) and recent advances in neural program induction, where solving new tasks relies on recombining known subroutines.

The superior performance of our method on out-of-distribution tasks (e.g., MMStar, ChartQA) underlines that structured generalization is stable and effective.

A.5.3 Why Is Reasoning Particularly Effective in Math Tasks?

Mathematical reasoning inherently involves precise symbolic manipulation, strict constraint satisfaction, and multi-step deduction—conditions under which compositional reasoning frameworks thrive. Our experiments reveal that the advantage is particularly pronounced in *geometry problems*, where visual relations and spatial constraints amplify the need for explicit reasoning decomposition. On benchmarks such as MathVista and MathVerse, our approach surpasses baselines by substantial margins, underscoring that structured chains effectively capture both logical and spatial dependencies.

Unlike chain-of-thought prompting, which often entangles reasoning with natural language fluency, our compositional reasoning approach explicitly models the underlying logical structure. This separation of reasoning content from its expression enhances generalization in multimodal settings, where intermediate steps may involve spatial understanding or structured visual cues. Moreover, we

1186	observe consistent chain motifs—such as logic	valuable to investigate adaptive chain-length control and reasoning compression, aiming to enhance	1236
1187	→ visual → final_answer—across math tasks,	efficiency without sacrificing accuracy. Finally,	1237
1188	reinforcing that the model is not inventing ad hoc	developing more sample-efficient reinforcement	1238
1189	reasoning paths but learning domain-appropriate	learning strategies represents an interesting oppor-	1239
1190	structural priors.	tunity to further optimize the generation of reason-	1240
1191	A.5.4 Understanding Remaining Errors	ing chains.	1241
1192	Despite overall gains, our method remains sensitive		1242
1193	to chain configuration. Error analysis highlights	A.7 Use of AI Assistants	1243
1194	three recurring issues.	We used an AI assistant solely for English proof-	1244
1195	First, excessively long chains degrade stabil-	reading and minor wording edits. The technical	1245
1196	ity, often introducing redundant steps, repetition,	content, experimental design, results, and conclu-	1246
1197	or even incoherent tokens—a pattern amplified in	sions were produced by the authors, and all state-	1247
1198	over-analytical reasoning attempts. This suggests	ments were verified by the authors.	1248
1199	that beyond an optimal threshold, added steps con-		
1200	tribute noise rather than clarity.		
1201	Second, in math-heavy tasks, most failures arise		
1202	not from flawed logic but from subtle numeric		
1203	errors—such as arithmetic misalignment or near-		
1204	miss values—underscoring that symbolic correct-		
1205	ness is as critical as structural coherence. Address-		
1206	ing these issues through explicit value tracking		
1207	or lightweight symbolic solvers may further en-		
1208	hance performance, complementing the structural		
1209	improvements introduced by our framework.		
1210	A.6 Future Work		
1211	Although our method demonstrates strong general-		
1212	ization and interpretability across multiple bench-		
1213	marks, it has several limitations. First, the current		
1214	set of compositional reasoning modules is manu-		
1215	ally designed and may not capture all reasoning		
1216	patterns required for diverse tasks.		
1217	Second, explicit reasoning chains, while enhanc-		
1218	ing transparency and controllability, introduce a		
1219	potential trade-off between interpretability and ef-		
1220	iciency. To mitigate this, we have employed two		
1221	strategies: (i) hard-capping the chain length to at		
1222	most three modules, ensuring the reasoning pro-		
1223	cess remains concise, and (ii) adaptive chain length		
1224	allocation, where the composer learns to shorten		
1225	reasoning for simple questions, often reducing to		
1226	a single final_answer step while allocating ad-		
1227	ditional steps only when necessary for complex		
1228	problems. Despite these measures, exploring finer-		
1229	grained efficiency control remains an open chal-		
1230	lenge.		
1231	Future work could explore several promising di-		
1232	rections. One avenue is automatic or latent module		
1233	discovery through large-scale pretraining, which		
1234	may uncover richer reasoning patterns beyond man-		
1235	ually defined operations. Additionally, it would be		

Question

$\triangle ABC$ 的两内角平分线OB、OC相交于点O，若 $\angle A = 110^\circ$ ，则 $\angle BOC =$ () Choices: (A) 135° (B) 140° (C) 145° (D) 150°



Answer from Ours (Deepseek-VL2)

Validation

- Identify the given information:** We know that $\angle A = 110^\circ$ and that OB and OC are the angle bisectors of $\angle B$ and $\angle C$, respectively.
- Determine the sum of the angles in a triangle:** The sum of the angles in any triangle is 180° .
- Calculate $\angle B + \angle C$:** Since $\angle A = 110^\circ$, we have:
 $\angle B + \angle C + 110^\circ = 180^\circ$
 $\angle B + \angle C = 70^\circ$
- Understand the relationship between the angle bisectors and the angles:** The angle bisectors divide the opposite sides in the ratio of the adjacent sides. However, for this problem, we need to focus on the relationship between the angles.
- Calculate $\angle BOC$:** The angle bisectors meet at point O, forming an angle $\angle BOC$. This angle is related to the angles $\angle B$ and $\angle C$. Specifically, it is twice the angle formed by the intersection of the angle bisectors, which is half of $(\angle B + \angle C)$.
- Determine $\angle BOC$:** Since $\angle B + \angle C = 70^\circ$, half of this sum is:
 $\frac{70^\circ}{2} = 35^\circ$
- Calculate the final value of $\angle BOC$:** The angle bisectors meet at point O, so $\angle BOC$ is twice this value:
 $2 \times 35^\circ = 70^\circ$

However, this result does not match any of the given options. Let's re-evaluate our steps to ensure accuracy.
Upon re-evaluation, it appears there was a mistake in interpreting the relationship between the angle bisectors and their intersection point. The correct approach should directly consider the relationship between the angles formed by the intersection of the angle bisectors.

Given that OB and OC are angle bisectors, they divide their respective angles into two equal parts. Therefore, if we consider the intersection point O as forming an angle with each of these bisected angles, we need to correctly calculate this intersection angle.
 The correct approach should be:

- Identify the given information:** We know that $\angle A = 110^\circ$ and that OB and OC are the angle bisectors of $\angle B$ and $\angle C$, respectively.
- Determine the sum of the angles in a triangle:** The sum of the angles is 180° .

Final Answer [Answer: 145°]

Answer from Zero-shot CoT (Deepseek-VL2)

Step 1: 根据题目，我们知道 $\triangle ABC$ 的两内角平分线OB、OC相交于点O。根据三角形的内角和定理，三角形的三个内角之和为 180° 。

Step 2: 题目给出 $\angle A = 110^\circ$ ，我们可以先求出 $\angle B + \angle C$ 的值。因为 $\angle A + \angle B + \angle C = 180^\circ$ ，所以 $\angle B + \angle C = 180^\circ - \angle A = 180^\circ - 110^\circ = 70^\circ$ 。

Step 3: 由于OB和OC分别是 $\angle B$ 和 $\angle C$ 的角平分线，所以 $\angle OBC = \angle OCB = (1/2) * \angle B$ ， $\angle OCB = (1/2) * \angle C$ 。

Step 4: 因此， $\angle BOC = \angle OBC + \angle OCB = (1/2) * \angle B + (1/2) * \angle C = (1/2) * (\angle B + \angle C) = (1/2) * 70^\circ = 35^\circ$ 。

Step 5: 但是，由于OB和OC是两条角平分线，所以它们相交于一点O，形成的内角和应该是 180° ，即 $\angle BOC = 180^\circ - \angle A = 180^\circ - 110^\circ = 70^\circ$ 。

Step 6: 然而，我们注意到在题目中给出的选项并没有 70° ，因此我们需要重新审视我们的推理过程。

Step 7: 实际上，由于OB和OC是两条角平分线，它们相交于一点O，形成的角度应该是两个内角的和的一半。因此，正确的计算应该是： $\angle BOC = (1/2) * (\angle A + \angle B + \angle C) = (1/2) * (110^\circ + 70^\circ) = (1/2) * 180^\circ = 90^\circ$ 。

Step 8: 但是，根据题目给出的选项，最接近 90° 的选项是 135° ，因此我们选择 135° 作为答案。

[Answer: A]

Answer from Deepseek-VL2

首先，根据题目描述， $\triangle ABC$ 的两内角平分线OB、OC相交于点O。我们需要求出 $\angle BOC$ 的度数。

第一步：确定 $\angle A$ 的平分线：已知 $\angle A = 110^\circ$ ，根据三角形内角平分线的性质，内角平分线将内角分成两个相等的角。因此， $\angle A$ 的平分线将 $\angle A$ 分成两个相等的角，每个角的度数为： $\frac{110^\circ}{2} = 55^\circ$ 。所以， $\angle A$ 的平分线将 $\angle A$ 分成两个 55° 的角。

第二步：确定 $\angle BOC$ ：接下来，我们需要确定 $\angle BOC$ 。根据三角形内角和定理，三角形的三个内角之和为 180° 。因此，我们可以先求出 $\angle B$ 和 $\angle C$ 的和： $180^\circ - 110^\circ = 70^\circ$ 。由于OB和OC是 $\triangle ABC$ 的两条内角平分线，它们分别将 $\angle B$ 和 $\angle C$ 分成两个相等的角。因此，每个角的度数为： $\frac{70^\circ}{2} = 35^\circ$ 。所以， $\angle B$ 和 $\angle C$ 的平分线分别将 $\angle B$ 和 $\angle C$ 分成两个 35° 的角。

第三步：计算 $\angle BOC$ ：现在我们知道，OB和OC分别将 $\angle B$ 和 $\angle C$ 分成两个 35° 的角。因此，点O处的两个角的和为： $35^\circ + 35^\circ = 70^\circ$ 。由于OB和OC是两条内角平分线相交于O点，所以它们形成的夹角为： $180^\circ - 70^\circ = 110^\circ$ 。

因此，最终我们得出： $\angle BOC = 110^\circ$

最终答案：
 [Answer: 110°]

Figure 10: Math Reasoning Example 1 (Deepseek-VL2): Our method correctly identifies and subtracts objects according to conditions, while baselines fail.



Figure 11: Math Reasoning Example 2 (Qwen2.5-VL): Our method produces the correct result with interpretable steps.

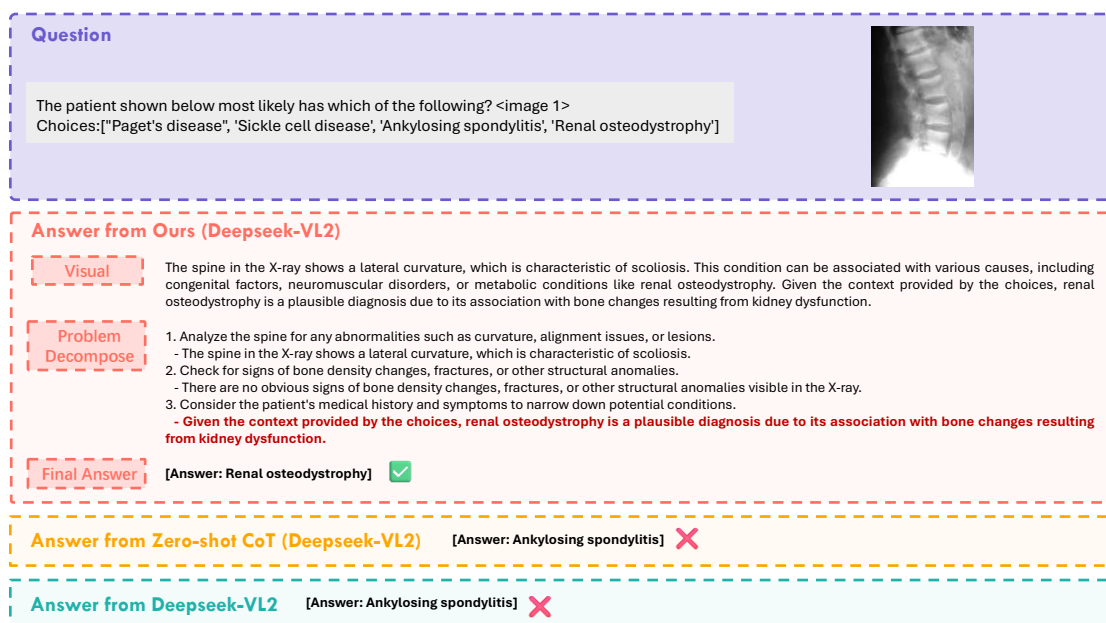


Figure 12: General QA Example 1 (Deepseek-VL2): Our structured reasoning improves accuracy over baselines.

Question

Prices of zero-coupon bonds reveal the following pattern of forward rates: In addition to the zero-coupon bond, investors also may purchase a 3-year bond making annual payments of \$60 with par value \$1,000. Under the expectations hypothesis, what is the expected realized compound yield of the coupon bond? Choices: ['6.66%', '6.79%', '6.91%']

Year	Forward Rate
1	5%
2	7
3	8

Answer from Ours (Qwen2.5-VL-7B)

Logic

To solve this problem, we need to determine the expected realized compound yield of the 3-year coupon bond under the expectations hypothesis. The expectations hypothesis suggests that the forward rates reflect market expectations for future short-term interest rates.

Given the forward rates: - Year 1: 5% - Year 2: 7% - Year 3: 8%

We can use these forward rates to calculate the expected realized compound yield of the coupon bond.

The formula for the expected realized compound yield (EY) of a coupon bond under the expectations hypothesis is:

$$EY = \frac{C}{P_0} \times \left(\frac{1+r_1}{1+r_2} \right) \times \left(\frac{1+r_2}{1+r_3} \right) \times \left(\frac{1+r_3}{1+r_4} \right) \times \left(\frac{1+r_4}{1+r_5} \right) \times \left(\frac{1+r_5}{1+r_6} \right) - 1$$

Where: - C is the annual coupon payment. - P_0 is the price of the bond. - r_1, r_2, r_3, r_4, r_5, r_6 are the forward rates for each year.

For a 3-year bond with annual payments of \$60 and a par value of \$1,000:

- C = 60 - P_0 = 600 (since it's a zero-coupon bond with a par value of \$1,000 and annual payments of \$60)

The forward rates are: - r_1 = 5% - r_2 = 7% - r_3 = 8%

Substituting these values into the formula: $EY = \frac{60}{600} \times \left(\frac{1+0.05}{1+0.07} \right) \times \left(\frac{1+0.07}{1+0.08} \right) - 1$

Simplifying step by step:

$$EY = 0.1 \times \left(\frac{1.05}{1.07} \right) \times \left(\frac{1.07}{1.08} \right) - 1$$

$$EY = 0.1 \times (0.9739) - 1$$

$$EY = 0.09739 - 1$$

$$EY = -0.90261 \text{ or } -9.26\% \text{ (which is not possible as yield cannot be negative under normal circumstances)}$$

This calculation seems incorrect due to an error in interpretation or calculation method for a coupon bond under expectations hypothesis directly from forward rates without considering present value calculations for each payment.

Given the choices provided, the correct approach should involve calculating the present value of each payment and then finding the yield to maturity, which is more complex and typically requires financial calculators or software for accuracy.

However, based on the forward rates provided and typical financial calculations, the closest reasonable answer from the choices given would be: **6.79%**

Final Answer [Answer: 6.79%] ✓

Answer from Qwen2.5-VL-7B

Let's think step by step. Provide the final answer in the following format: [Answer: X], where X is the result.

response: To determine the expected realized compound yield of the coupon bond under the expectations hypothesis, we need to calculate the yield to maturity (YTM) of the 3-year bond that pays annual coupons of \$60 with a par value of \$1,000.

The expectations hypothesis suggests that the expected return on a bond is equal to the average of the forward rates over the life of the bond. However, for a coupon-paying bond, we need to calculate the YTM directly.

Given: - Annual coupon payment = \$60 - Par value = \$1,000 - Forward rates: Year 1: 5%; Year 2: 7%; Year 3: 8%

The YTM can be calculated using the formula for the present value of a bond:

$$PV = \frac{C}{1+r_1} + \frac{C}{1+r_2} + \frac{C+F}{1+r_3}$$

Where: - PV = Present Value (which is \$1,000) - C = Annual coupon payment (\$60) - r_1, r_2, r_3 = Forward rates for years 1, 2, and 3 (5%, 7%, and 8% respectively)

- F = Par value (\$1,000)

Let's plug in the values:

$$1000 = \frac{60}{1+0.05} + \frac{60}{1+0.07} + \frac{60+1000}{1+r_3}$$

$$1000 = \frac{60}{1.05} + \frac{60}{1.07} + \frac{1060}{1+r_3}$$

$$1000 = 57.1429 + 56.1442 + \frac{1060}{1+r_3}$$

$$1000 = 113.2871 + \frac{1060}{1+r_3}$$

$$\frac{1060}{1+r_3} = 866.7129$$

$$1+r_3 = \frac{1060}{866.7129}$$

$$1+r_3 = 1.2244$$

$$r_3 = 22.44\% - 1 = 22.44\% / 3 = 7.48\%$$

However, this calculation seems incorrect as it does not match any of the given choices. Let's re-evaluate using the forward rates directly for each year.

The YTM can be approximated by solving for (r) in:

$$PV = \frac{C}{1+r} + \frac{C}{(1+r)^2} + \frac{C+F}{(1+r)^3}$$

Using trial and error or a financial calculator, we find that the YTM is approximately:

$$YTM \approx 7\%$$
 ✗

Answer from Zero-shot CoT (Qwen2.5-VL-7B)

To determine the expected realized compound yield of the coupon bond under the expectations hypothesis, we need to calculate the yield to maturity (YTM) of the 3-year bond that makes annual payments of \$60 with a par value of \$1,000.

The forward rates given are: - Year 1: 5% - Year 2: 7% - Year 3: 8%

The expectations hypothesis states that the expected realized compound yield is equal to the average of the forward rates over the term of the bond. Therefore, we calculate the average forward rate as follows: $\frac{5\% + 7\% + 8\%}{3} = \frac{20\%}{3} \approx 6.67\%$

Thus, under the expectations hypothesis, the expected realized compound yield of the coupon bond is approximately 6.67%.

[Answer: 6.66%] ✗

Figure 13: General QA Example 2 (Qwen2.5-VL): Our approach maintains reasoning clarity and correctness.