Clarification as Supervision: Reinforcement Learning for Vision-Language Interfaces

John Gkountouras[1] **Ivan Titov**[1,2]
[1]ILLC, University of Amsterdam
[2]ILCC, University of Edinburgh
i.gkountouras@uva.nl ititov@inf.ed.ac.uk

Abstract

Recent text-only models demonstrate remarkable reasoning capabilities. Extending these to visual domains requires vision-language models to translate images into text descriptions. However, current models, trained to produce captions for human readers, often omit the precise details that reasoning systems require. This creates an interface mismatch: reasoners often fail not due to reasoning limitations but because they lack access to critical visual information. We propose Adaptive-Clarification Reinforcement Learning (AC-RL), which, through interaction, teaches vision models what information reasoners need. Our key insight is that clarification requests during training reveal information gaps; by penalizing success that requires clarification, we create pressure for comprehensive initial captions that enable the reasoner to solve the problem in a single pass. AC-RL improves average accuracy by 4.4 points over pretrained baselines across seven visual reasoning benchmarks, and analysis shows it would cut clarification requests by up to 44% if those were allowed. By treating clarification as a form of implicit supervision, AC-RL demonstrates that vision-language interfaces can be effectively learned through interaction alone, without requiring explicit annotations.

1 Introduction

Recent advances in reinforcement learning have produced text-based reasoning models with remarkable reasoning capabilities [14, 33]. While these reasoning capabilities are impressive, extending them to visual domains requires careful consideration of how visual and linguistic information should interface.

Several recent works explore decoupled architectures for visual reasoning, where vision modules translate images into text descriptions that are then processed by text-only reasoners [3, 53, 17]. This modular paradigm can offer practical advantages: it enables reuse of existing text-only reasoning models without costly multimodal retraining, allows flexible composition of specialized components, and provides interpretable interfaces between perception and reasoning. This decoupling is particularly relevant when powerful reasoners are available only through APIs that cannot be fine-tuned, or when reasoners are prohibitively expensive to fine-tune. Furthermore, many domains have specialized vision-language models (e.g., for medical imaging [24], web interfaces [22], or engineering diagrams [7]) but these models often lack the broader reasoning ability of text-based models. This further motivates decoupling perception from reasoning, and learning the interface between them. Systems like COLA and ViCor demonstrate this approach, using LLMs as coordinators that operate on text descriptions of visual content [4]. The common thread in these approaches is that visual information flows through a linguistic bottleneck, requiring careful design of what information to communicate [35, 16].

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

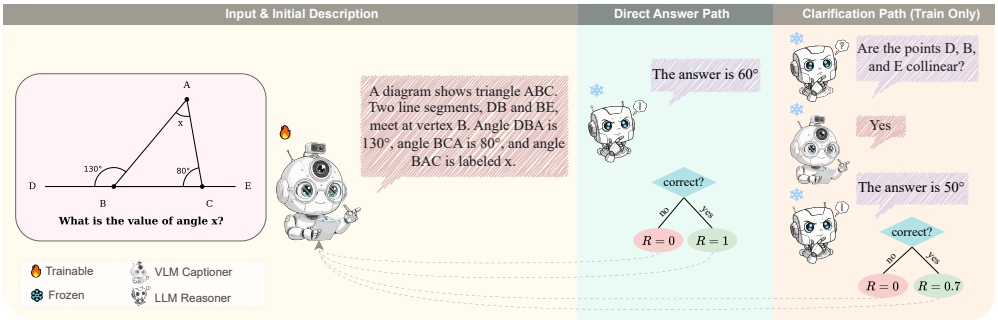


Figure 1: **Adaptive-Clarification Reinforcement Learning (AC-RL) training framework**. Given an image and a question, a trainable captioner generates an initial description. During training, the frozen reasoner evaluates whether this description contains sufficient detail to solve the problem. If yes (Direct Answer Path), it attempts to answer directly, receiving reward $R = 1$ for correct answers or $R = 0$ for incorrect ones. If the description lacks crucial information (Clarification Path), the reasoner requests specific details, which are provided by a frozen copy of the initial captioner. Correct answers after clarification receive partial reward $R = \alpha = 0.7$, while incorrect answers receive $R = 0$. Gradients (dotted arrows) flow only through the initial caption generation, not through clarification responses. At inference, only the direct answer path is used: the model has learned to generate sufficiently detailed initial captions, eliminating the need for clarification.

This decoupling however creates a critical alignment challenge: vision-language tools must learn what visual information a reasoner requires for successful problem-solving. Vision-language models are typically trained on diverse multimodal datasets to produce descriptions sufficient for general visual understanding and question answering. Reasoners require different visual information than human readers do, and what counts as informative depends on what the reasoner does with the caption. Traditional supervised approaches would require annotating "ideal captions" for each reasoner, an infeasible task given the diversity of visual reasoning problems and the implicit nature of reasoner preferences. Moreover, what constitutes an informative caption cannot be determined a priori; it emerges only through interaction with the reasoning model.

Reinforcement learning offers a natural framework for this interface learning problem, but applying it to vision-reasoner coordination is challenging. The primary difficulty lies in the sparsity of learning signals: when using binary task rewards, the vision model receives identical zero rewards whether its caption is completely inadequate or missing one crucial detail. Additionally, the large action space of language generation combined with sparse rewards leads to inefficient exploration, where most generated captions result in failure without providing informative gradients.

To overcome the limitations of this sparse reward signal, we introduce Adaptive-Clarification Reinforcement Learning (AC-RL), a framework that encourages vision-language models to produce captions aligned with the needs of a specific reasoner (Figure 1). The key idea is to make use of successful clarification during training: when the reasoner fails to solve a task directly but succeeds after a clarification exchange, the initial caption likely contained partially useful but incomplete information. We assign partial credit to such clarification-based success but prefer solutions achieved on the first attempt, thereby creating optimization pressure for more informative initial captions. During training, clarification responses are generated by a frozen reference model, and gradients flow only through the initial caption. This ensures that the clarifier is not optimized and prevents the captioner from "hiding behind" a strong clarifier instead of learning to produce sufficiently informative initial captions. Over time, this leads the vision module to front-load relevant details into the initial caption, enabling single-pass inference without clarification at test time.

More specifically, AC-RL transforms sparse binary rewards into a tiered structure: full reward for direct success, partial reward (e.g., $\alpha = 0.7$) for success requiring clarification, and zero for failure. This densification serves dual purposes. First, it converts many zero-reward episodes into partially rewarded ones, providing a learning signal when initial captions are nearly sufficient. Second, the penalty $(1 - \alpha)$ creates pressure to discover self-sufficient captioning strategies aligned with single-pass deployment. Through thousands of interactions, the vision model explores different description strategies, learning without explicit supervision what quantitative details, spatial relationships, or structural patterns this particular reasoner needs for solving a problem. The learned strategies also transfer to reasoners unseen during training (Section 4.2.3).

We evaluate AC-RL on visual reasoning benchmarks, chosen as a controlled testbed where clear ground truth enables clean measurement and strong text-only reasoners exist, isolating the vision-language interface as the primary bottleneck. Most of these benchmarks target different forms of mathematical reasoning (such as visual geometry, diagram interpretation, logic puzzles, or reasoning over plots and tables) yet some also extend beyond mathematics, including problems from chemistry, physics, and biology, e.g., MMMU [49].

The key contributions of our work are as follows.

- An exploration-based framework that enables vision-language models to discover through reinforcement learning what visual information a reasoner requires, adapting from human-caption pretraining without explicit supervision.
- A clarification-aware reward structure that uses interaction patterns as learning signals, allowing models to identify information gaps and iteratively improve their captioning strategies through trial and error.
- An empirical demonstration that our clarification-aware training scaffold effectively teaches captioners to anticipate reasoner needs, leading to improved accuracy on seven visual reasoning benchmarks and a measurable reduction in clarification dependency at inference.

2 Related Work

Reinforcement learning for reasoning in language models. Recent work shows RL can teach extended mathematical reasoning, with DeepSeek-R1 demonstrating that learned policies outperform prompt-based chain-of-thought [14]. Visual extensions employ diverse strategies: training stability (Skywork-R1V2 [42], Vision-R1 [19]), replay mechanisms (VL-Rethinker [40], OpenVLThinker [6]), and cross-modal formalization (R1-OneVision [47], Mulberry [48]). These methods focus on extending reasoning chains to handle visual inputs. We take an orthogonal approach by shaping the interface between perception and reasoning modules, using clarification-aware rewards to teach captioners what information reasoners need.

Decoupled perception–reasoning and interface design. Decoupling visual perception from linguistic reasoning offers modularity and the ability to reuse strong text-only reasoners, but it raises an interface-alignment challenge: captions optimized for human readability may omit the quantitative and structural cues a reasoner needs [53, 16, 35]. Coordination frameworks use an LLM to route or aggregate information from one or more VLMs (e.g., COLA's coordinator that queries complementary experts) [3], or to interleave "see-think-confirm" phases that explicitly ground and verify intermediate steps (VCTP) [4]. Neuro-symbolic systems like VisProg sidestep monolithic pipelines by composing programs over off-the-shelf vision tools [17]. Our approach adheres to the decoupled setup but replaces fixed protocols with an RL objective that learns, from interaction, which *caption features* best serve a specific reasoner.

Learning alignment through interaction. Several methods optimize captions for downstream reasoning rather than human readability. Most relevant to our work, RACRO [13] and its updated version, RAPID [12], train a captioner against a frozen text-only reasoner using answer-correctness rewards, demonstrating that interface learning is possible through RL alone. Our approach is complementary: we replace the binary correctness signal with a tiered reward derived from clarification interactions, providing graded supervision over caption sufficiency. LAMOC and VLRM use language-model feedback and a VLM-as-reward-model, respectively [8, 10]. OmniCaptioner generates long-context descriptions to support LLM reasoning [30], and Critic-V employs a learned VLM critic [50]. Beyond vision, multi-agent frameworks show that LLMs can coordinate through language-only protocols and that adapting inputs to a solver's biases can improve performance [44, 54].

3 Methodology

3.1 The Vision-Reasoner Interface Problem

We consider a modular architecture where a trainable vision-language model, the captioner $\mathcal{V}_\theta$, translates images into text descriptions that enable a frozen text-only model, the reasoner $\mathcal{R}$, to solve

visual reasoning tasks. Given an image I and question Q, the system produces an answer A. The central challenge lies in learning what visual information the specific reasoner requires, without explicit supervision defining "ideal captions".

We specifically target scenarios where the reasoner $\mathcal{R}$ is frozen, reflecting deployment constraints where the reasoning system cannot be modified (e.g., accessible only via an API). This makes the learning problem more challenging: the captioner must unilaterally adapt to a fixed target through interaction, with no co-adaptation to make the reasoner more accommodating.

Our approach uses successful clarification as implicit supervision. When the reasoner fails to solve a task directly but succeeds after a clarification exchange, this suggests that the initial caption was partially informative: it must have provided enough context for the reasoner to identify what might be missing, formulate a meaningful follow-up question, and ultimately solve the problem. AC-RL captures this signal through a tiered reward structure: full reward for direct success, partial reward for success after clarification, and zero otherwise. Over time, the captioner learns to anticipate the kind of information that would otherwise be requested and to include it proactively in the initial description. This leads to progressively more informative captions and enables efficient single-pass inference without clarification at test time.

3.2 Training and Inference Protocols

During training, we permit structured interaction between the captioner and reasoner. The captioner first generates an initial caption $c_0 \sim \pi_\theta(\cdot \mid I, Q)$ describing the visual content. The reasoner processes this caption and either produces an answer directly or requests clarification with a specific question q_1. When clarification is requested, a *frozen reference model* π_{ref} provides the response $c_1 \sim \pi_{\text{ref}}(\cdot \mid I, Q, q_1)$. The reasoner then produces its final answer A using all available information.

The clarification response comes from a frozen checkpoint that receives no gradients during training. This design ensures that the captioner cannot rely on improving clarification capabilities and must instead learn to front-load relevant information into the initial caption. Details of this protocol and the complete algorithm appear in Figure 1 and Appendix A.

At inference time, the system operates in a single pass: the captioner generates one description $c_0 \sim \pi_\theta(\cdot \mid I, Q)$, and the reasoner must produce the answer based solely on this initial caption. This single-pass constraint is crucial for practical applications where multi-turn interaction would be computationally expensive or require architectural changes to existing tool-calling frameworks. By learning to front-load information during training, our approach produces captioners that work with standard single-pass inference. The reasoner processes the initial caption without needing to be modified to request clarifications.

3.3 Clarification-Aware Reward Design

AC-RL uses the tiered reward structure that densifies the learning signal. In standard reinforcement learning for visual question answering, episodes receive binary rewards based solely on answer correctness. This sparse signal provides limited feedback when the captioner produces nearly sufficient but incomplete descriptions.

Our reward function addresses this sparsity by distinguishing three outcomes:

$$R(\tau) = \begin{cases} 1 & \text{if correct answer without clarification} \\ \alpha & \text{if correct answer with clarification} \\ 0 & \text{if incorrect answer} \end{cases} \tag{1}$$

where $\alpha \in (0, 1)$; τ denotes the complete episode trajectory.

This structure serves dual purposes. First, it converts many zero-reward episodes into partially rewarded ones, providing gradient signal when the initial caption contains most but not all necessary information. This densification is particularly valuable early in training when captions frequently lack specific details. Second, the penalty $(1 - \alpha)$ for requiring clarification creates optimization pressure toward self-sufficient initial captions that align with single-pass deployment.

Clarification responses come from a frozen reference model rather than the training policy. This ensures gradients flow only through the initial caption, encouraging front-loading instead of relying

on a clarifier that is used only to define partial rewards and is absent at test time. Also note that problems beyond the reasoner's capability contribute no gradient: when all caption variants for a problem receive zero reward, all advantages are zero and the problem is effectively ignored during optimization.

The clarification mechanism thus acts as a scaffold that provides intermediate credit assignment. Episodes where the reasoner would fail with the initial caption alone but succeeds after clarification receive partial reward, signaling that the caption was nearly adequate. This graded feedback enables more sample-efficient learning compared to binary rewards that treat all failures equivalently.

3.4 Policy Optimization

We optimize the captioner using a KL-regularized objective that balances task performance with proximity to the pretrained initialization:

$$J(\theta) = \mathbb{E}_{(I,Q)\sim\mathcal{D}}\left[\mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)]\right] - \beta \cdot D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}) \tag{2}$$

where π_{ref} denotes a fixed reference policy. We employ Beta-Normalization Policy Optimization [45][1], which is a variant of GRPO [33], an on-policy RL algorithm that optimizes over groups of responses per prompt. Although rewards are assigned individually to each rollout, the update is driven by relative performance across rollouts for the same task (I, Q): only captions that perform strictly better than alternatives contribute a gradient. When all rollouts fail identically (including after clarification) no update is applied, preventing tasks unsolvable by the reasoner from penalizing otherwise potentially strong captions.

Gradients flow only through the initial caption generation c_0. Neither the frozen reasoner $\mathcal{R}$ nor the clarification model π_{ref} receives gradient updates, ensuring the captioner adapts unilaterally to the fixed reasoner's preferences. We show in Appendix B that our tiered reward preserves unbiasedness of the policy gradient estimator despite post-action stochasticity from the reasoner.

4 Experiments

We evaluate whether Adaptive-Clarification Reinforcement Learning (AC-RL) successfully aligns vision-language models with the information needs of downstream reasoning systems. Our experimental design tests five key hypotheses: (1) AC-RL improves task performance compared to both pretrained models and standard reinforcement learning approaches (i.e., learning with the tiered rewards and clarifications is beneficial), (2) the clarification-aware training scaffold contributes meaningfully to performance gains beyond standard RL, (3) the improvements stem from learning to front-load reasoner-relevant information into initial captions, (4) AC-RL is robust to the penalty α, and (5) generalizes to held-out reasoners and to additional benchmarks.

System Architecture. We instantiate the trainable captioning policy with InternVL3-2B or Qwen2.5-VL-3B. Their modest size enables extensive RL experimentation and ablations, and when paired with a strong reasoner, they provide reliable baseline competence across the evaluated benchmarks. Their scale also makes GRPO optimization tractable without requiring extensive computational resources. The frozen reasoning system $\mathcal{R}$ is DeepSeek-R1-Qwen-32B, a powerful text-only model trained for mathematical reasoning with particularly strong instruction following capabilities. We also evaluate the vision models as standalone systems to quantify the benefits of architectural decoupling. To verify that AC-RL operates in the deployment scenario where the reasoner is exposed only through an API and cannot be modified or inspected, we additionally instantiate $\mathcal{R}$ with Google's Gemini 3.1 Flash-Lite, accessed exclusively through its API for both training and evaluation.

Training Configurations. We compare six configurations to isolate the effects of design choices: **Standalone VLM** (the VLM answers directly), **End-to-End RL** (BNPO directly on the standalone VLM with binary correctness rewards), **Pretrained + Reasoner** (no fine-tuning, modular only), **Binary-Reward RL** (caption-only RL with binary task rewards, matching the VPO objective in RACRO/RAPID [13, 12]), **Binary + Similarity** (binary rewards plus an image-text similarity term, Section 4.2.4), and **AC-RL** (our tiered rewards). All RL methods train on ViRL-39K [40].

[1] BNPO normalizes rewards within each group with a fitted Beta distribution (Appendix A).

Training Protocol. AC-RL training uses the clarification-aware scaffold detailed in Section 3. During training, the captioner generates $c_0 \sim \pi_\theta$, and if the reasoner requests clarification, a frozen reference policy provides the response. The tiered rewards ($R = 1$ for direct success, $R = 0.7$ with clarification, $R = 0$ for failure) create gradients only through the initial caption. We optimize using BNPO with KL regularization. AC-RL also maintains greater generation diversity than standard RL throughout training (Appendix D).

Evaluation Protocol. All models are evaluated using **single-pass evaluation**: the captioner produces a description that the reasoner uses to generate a final answer, with no clarification permitted. This protocol ensures that performance gains reflect improved caption quality rather than multi-turn interaction benefits. For behavioral analyses in Section 4.3, we additionally conduct instrumented runs with **clarification-enabled evaluation** where clarification is allowed, to measure clarification patterns. We evaluate on seven benchmarks spanning diverse visual math reasoning challenges, plus three additional benchmarks, and compare against leading proprietary and open-weights models (details in Appendix C). We report exact-match accuracy using EvalScope [37] and VLMEvalKit [9].

4.1 Overall Performance

Table 1: Main results on multi-modal reasoning benchmarks: MathVista (MVista), MathVision (MVision), MathVerse (MVerse), MMMU, WeMath (WeM), DynaMath (DynaM), and LogicVista (LVista). Paired bootstrap significance tests are reported in Appendix G.

	Model	MVista	MVision	MVerse	MMMU	WeM	DynaM	LVista	AVG
Proprietary Models	GPT-4o-20241120	60.0	31.2	40.6	70.7	45.8	34.5	52.8	47.9
	Gemini-2.0-Flash	70.4	43.6	47.7	72.6	47.4	42.1	52.3	53.7
	Claude-3.7-Sonnet	66.8	41.9	46.7	75.0	49.3	39.7	58.2	53.9
	o1	73.9	42.2	—	**78.2**	—	—	—	—
	Gemini 2.5 Pro	**80.9**	**69.1**	**76.9**	74.7	**78.0**	**56.3**	**73.8**	**72.8**
	Seed1.5-VL (Thinking)	79.5†	68.7	—	77.9	77.5	—	—	75.9*
Open-Weights Models	InternVL3-2B-MPO	57.0	21.7	25.3	48.6	22.4	14.6	36.9	32.4
	InternVL3-8B-MPO	71.6	29.3	39.8	62.7	37.1	25.5	44.1	44.3
	Ovis2-8B	71.8†	25.9	42.3	59.0	—	—	39.4	47.7
	InternVL3-14B-MPO	75.1	37.2	44.4	67.1	43.0	31.3	51.2	49.9
	QVQ-72B-Preview	70.3	34.9	48.2	70.3	39.0	30.7	58.2	50.2
	MMR1-Math-v0-7B	71.0†	30.2	49.2	—	—	—	50.8	50.3
	InternVL3-38B-MPO	75.1	34.2	48.2	70.1	**48.6**	**35.3**	**58.4**	52.8
	VL-Rethinker-72B	**80.3**	**43.9**	—	68.8	—	—	—	—
	InternVL3-78B-MPO	79.0	43.1	**51.0**	**72.2**	46.0	35.1	55.9	**54.6**
InternVL 2B	Standalone VLM	57.0	21.9	25.3	48.6	22.4	14.6	36.9	32.4
	Pretrained + 32B Reasoner	61.0	34.7	28.9	57.4	**32.8**	12.0	48.3	39.3
	AC-RL + 32B Reasoner (ours)	**65.3**	**36.7**	**36.8**	**58.4**	32.1	**20.0**	**49.0**	**42.6**
		(+4.3)	(+2.0)	(+7.9)	(+1.0)	(-0.7)	(+8.0)	(+0.7)	(+3.3)
Qwen2.5 VL-3B	Standalone VLM	**64.5**	21.9	28.8	50.1	24.2	13.4	39.6	34.6
	Pretrained + 32B Reasoner	59.7	32.8	29.2	55.2	**34.7**	14.2	47.2	39.0
	AC-RL + 32B Reasoner (ours)	63.8	**36.8**	**34.4**	**57.7**	33.2	**24.8**	**53.0**	**43.4**
		(+4.1)	(+4.0)	(+5.2)	(+2.5)	(-1.5)	(+10.6)	(+5.8)	(+4.4)
	Pretrained + Gemini Reasoner	61.8	37.9	31.9	62.7	31.6	10.4	49.4	40.8
	AC-RL + Gemini Reasoner	**66.1**	**40.5**	**39.5**	**66.8**	**34.4**	**20.2**	**54.4**	**46.0**
		(+4.3)	(+2.6)	(+7.6)	(+4.1)	(+2.8)	(+9.8)	(+5.0)	(+5.2)

†: Result on *testmini/mini* subset. *: Restricted to the available datasets; not directly comparable to full-suite averages.

Table 1 presents our results in the context of leading proprietary and open-weights models. We first note that small vision-language models achieve limited performance when solving problems directly: InternVL3-2B and Qwen2.5-VL-3B reach only 32.4% and 34.6% average accuracy respectively as standalone systems. Simply pairing these models with a strong reasoner (Pretrained + Reasoner) improves performance to 39.3% and 39.0%, demonstrating the value of modular architectures. However, applying AC-RL yields the largest gains.

With a Qwen2.5-VL-3B captioner, AC-RL improves the average accuracy from 39.0 to 43.4 (+4.4 points), with substantial gains on robustness and vision-centric benchmarks like *DynaMath* (+10.6) and *MathVerse* (+5.2). The InternVL3-2B captioner sees a similar +3.3 average point increase. The

Table 2: **Decomposition of performance gains on Qwen2.5-VL-3B** across multi-modal reasoning benchmarks: MathVista (MVista), MathVision (MVision), MathVerse (MVerse), MMMU, WeMath (WeM), DynaMath (DynaM), and LogicVista (LVista). All models are evaluated in a single-pass setting. The results show that AC-RL provides a significant performance boost beyond both architectural decoupling and Binary Rewards. Significance tests are presented in Appendix G.

Training Method	MVista	MVision	MVerse	MMMU	WeM	DynaM	LVista	AVG
VLM-only (No Reasoner)	**64.50**	21.90	28.80	50.10	24.20	13.40	39.60	34.64
VLM-only + End-to-end RL	64.50	21.20	29.70	51.00	28.50	13.40	40.70	35.60
Decoupled (No RL)	59.69	32.80	29.18	55.22	**34.71**	14.17	47.20	39.00
Binary-Reward RL	62.60	34.30	31.09	55.44	33.45	17.56	47.42	40.27
Binary + Similarity	62.00	36.48	23.73	57.44	32.10	19.76	51.90	40.49
AC-RL (Ours)	63.80	**36.84**	**34.39**	**57.70**	33.22	**24.75**	**53.02**	**43.39**

bottom rows in the Qwen2.5-VL-3B group apply AC-RL with Gemini 3.1 Flash-Lite [11] as the frozen reasoner instead of the open-source 32B model. Trained and evaluated entirely against the API, AC-RL improves over the pretrained captioner paired with the same reasoner on all 7 benchmarks, with a $+5.2$ point average gain. The per-benchmark pattern matches the open-source case, with the largest gains on the extraction-heavy benchmarks (DynaMath, MathVerse, LogicVista). This confirms that AC-RL operates with a black-box API reasoner, with no aspect of the reasoner exposed beyond its API outputs. These results, obtained under an identical single-pass protocol, demonstrate that AC-RL effectively aligns the captioning policy with the downstream reasoner's needs.

We observe minor regressions on WeMath (-0.7 to -1.5 points), which we attribute to a structural difference: WeMath problems are organized around required textbook knowledge concepts rather than visual extraction. To test whether caption quality is the bottleneck, we hold the 32B reasoner fixed and scale the captioner from 3B to 32B (no RL); MathVision and DynaMath improve by $+11.9$ and $+22.2$ points, respectively, while WeMath moves by only $+2.3$ (in-depth analysis in Appendix H). Finally, the interface bottleneck addressed by AC-RL also persists at larger captioner scales: pairing a pretrained Qwen2.5-VL-7B with the same 32B reasoner improves over the 7B standalone by $+3.6$ average points, with gains concentrated on the same extraction-heavy benchmarks (Appendix I). AC-RL optimizes what the reasoner sees, not what it knows.

The gains also carry over to medical images and tables. With the same 32B reasoner,[2] AC-RL improves accuracy from 50.2 to 58.3 on WorldMedQA-V [31], which pairs medical licensing exam questions with clinical images, and from 31.8 to 42.4 on TableVQA-Bench [21]. Both benchmarks depend on reading specific details from the image, which is the setting AC-RL targets.

4.2 Ablations

To better understand the source of these improvements, we analyze the incremental value of each component in our approach using the Qwen2.5-VL-3B model (Table 2). All configurations are evaluated using direct inference (no clarification allowed).

4.2.1 Decomposition of Performance Gains

The results illustrate a clear progression. End-to-end RL applied directly to the standalone VLM yields only $+1.0$ points on average, indicating that the small VLM's reasoning capacity, rather than access to RL, bottlenecks the standalone setup. Decoupling the captioner from the reasoner (Decoupled, No RL) yields substantially larger gains, particularly on structurally complex tasks like *MathVision* ($21.9 \rightarrow 32.8$). Adding binary-reward RL on top of decoupling provides further improvements across most benchmarks, and AC-RL delivers the most substantial additional gains: *DynaMath* improves by $+7.2$ points ($17.56 \rightarrow 24.75$), *LogicVista* by $+5.6$ points, and *MathVerse* by $+3.3$ points.

4.2.2 Sensitivity to Clarification Penalty

In all main experiments, we set $\alpha = 0.7$ (clarification penalty of 0.3). We conduct a sensitivity analysis in Appendix E, finding that intermediate penalties (0.3–0.5) consistently outperform extremes,

[2]These results use the AC-RL checkpoint from Appendix E. Appendix N reports further benchmarks.

Table 3: Clarification behavior analysis. "Rate" = how often the reasoner requests clarification. "Gap" = accuracy change when clarification is allowed (Abs = absolute points, Rel = fraction of errors corrected). Acc_{deny} = accuracy without clarification on the problems where clarification is requested. Δ_{deny} = Single-Pass minus Acc_{deny}.

Dataset	Model	Clarification Rate	Clarification Gap			Denied Clarification	
			Single-Pass	+Clarification	Gap (Abs / Rel)	Acc_{deny}	Δ_{deny}
MathVision	Binary-Reward	40.7%	34.31%	37.20%	+2.89 / 0.044	22.88%	11.43
	AC-RL	29.0%	36.84%	37.66%	**+0.82 / 0.013**	21.93%	**14.91**
MathVerse	Binary-Reward	62.9%	31.09%	35.15%	+4.06 / 0.059	28.43%	2.66
	AC-RL	35.0%	34.39%	34.26%	**-0.13 / -0.002**	26.45%	**7.94**

Table 4: Clarifier controls. AC-RL here is the checkpoint from Appendix E.

Dataset	AC-RL	Weaker Clarifier	Permuted Rewards	Uninformative Clarifier	Binary Reward	Pretrained (No RL)
LogicVista	**55.70%**	52.80%	50.11%	46.76%	47.87%	47.20%
MathVision	**40.23%**	38.09%	38.22%	37.17%	38.16%	32.80%

with the method showing robustness across this range. Penalties that are too weak provide insufficient optimization pressure, while overly strong penalties approach binary rewards and lose the partial credit benefits. A decaying schedule for α does not improve over the fixed value either (Appendix E).

4.2.3 Generalization Across Reasoners

A natural question is whether learned strategies generalize or overfit to the training reasoner. We pair the AC-RL captioner with two reasoners unseen during training: DeepSeek-R1-0528-Qwen3-8B (smaller open-weights) and OpenAI's GPT-5 mini [34] (proprietary, API only). Under the unseen 8B reasoner, AC-RL improves over the pretrained captioner by $+8.7$ points on LogicVista and $+6.5$ on MathVision; under GPT-5 mini, it improves by $+3.9$ on MathVista and $+2.9$ on LogicVista. Full numbers and the checkpoint used for these studies are in Appendix L. The strategies AC-RL learns therefore transfer across reasoners differing in scale, training procedure, and access modality.

4.2.4 Reward Densification and Clarifier Controls

To isolate the contribution of the clarification signal specifically, as opposed to reward densification more generally, we compare against an alternative that augments the binary correctness reward with image-text similarity from a vision-language embedding model, an approach explored in prior caption-optimization work [52, 10, 25]. Calibrating a similarity term from Qwen3-VL-Embedding-2B [26] on top of binary rewards, we find that similarity-densified RL (Binary + Similarity in Table 2) improves over Binary-RL on 4 of 7 benchmarks but is outperformed by AC-RL on all of them, with a gap of $+5.0$ on DynaMath, the most extraction-heavy benchmark. The pattern reflects the underlying mechanism: similarity rewards optimize for generic visual fidelity, whereas clarification-based rewards target task-specific information sufficiency. On MathVerse, similarity-densified RL falls even below Binary-RL, because whole-image similarity pulls the captioner away from the few values a sparse diagram requires (Appendix J).

We also test the clarification signal directly with three controls (Table 4). The first replaces the clarifier with a weaker model, Qwen2-VL-2B [43]. The second replaces every clarification response with a fixed uninformative string, so the reasoner can still request clarification but receives no new information, while the reward rule and tier values stay unchanged. The third shuffles the positive rewards (1 and α) among the successful rollouts of each group, so a caption solved directly can receive α and one that needed clarification can receive 1. Each group keeps the same reward values, and with them the same density of the signal, and only their link to clarification is removed. Accuracy on LogicVista drops as the clarification carries less information, and with an uninformative clarifier it falls to the level of binary-reward RL. Permuting the rewards costs 5.6 points on LogicVista. On MathVision, all three controls end up close to binary-reward RL. A denser reward alone therefore does not explain the gain (details in Appendix K).

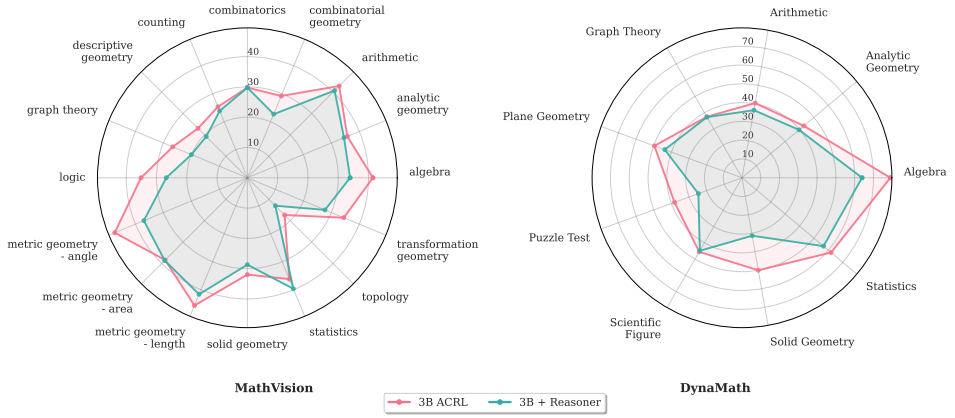


Figure 2: Subject-level performance comparing AC-RL to the pretrained baseline using Qwen2.5-VL-3B + Reasoner. Left: MathVision subjects. Right: DynaMath categories. AC-RL shows targeted improvements in quantitatively intensive categories.

4.3 Analysis of Interface Behavior

To understand the mechanism underlying AC-RL's performance gains, we analyze how the training procedure modifies the captioner's behavior. Our hypothesis is that the clarification-aware reward teaches the model to *front-load* reasoner-salient information into the initial caption, thereby reducing the need for clarification.

We first measure the **clarification attempt rate** using clarification-enabled evaluation (for measurement purposes only) and counting how frequently the reasoner requests clarification when processing captions from AC-RL-trained versus baseline models. Table 3 shows that AC-RL reduces the frequency of clarification requests. On MathVision, the clarification rate drops from 40.7% (Binary-Reward RL baseline) to 29.0% (AC-RL). On MathVerse, the reduction is even larger, from 62.9% to 35.0% (a 44% relative drop). This confirms that AC-RL-trained captioners learn to preemptively include information that would otherwise trigger follow-up questions.

Building on this evidence, we compute the **clarification gap**: the difference in accuracy between clarification-enabled evaluation and single-pass evaluation. A smaller gap indicates that the initial caption is more informationally self-sufficient. Table 3 presents these results. For the baseline model, allowing clarification provides substantial accuracy gains: +2.89 points on MathVision and +4.06 points on MathVerse. In contrast, the AC-RL model shows essentially no benefit from clarification: +0.82 points on MathVision and -0.13 points on MathVerse, indicating that its initial captions are sufficiently informative that follow-up exchanges add little. The relative improvement metric (fraction of previously incorrect answers corrected by clarification) confirms the pattern: AC-RL achieves 1.3% relative improvement on MathVision versus 4.4% for the baseline. This also holds with up to three rounds of clarification, which improve the pretrained captioner by up to 5.2 points but leave the accuracy of AC-RL essentially unchanged (Appendix Q).

To assess whether clarification requests are genuinely necessary, we measure accuracy under **denied clarification**: we identify instances where the model requested clarification during clarification-enabled evaluation, and compute Acc_{deny}, the accuracy on this subset when no clarification is allowed. The drop $\Delta_{\text{deny}} = \text{Acc}_{\text{single}} - \text{Acc}_{\text{deny}}$, where $\text{Acc}_{\text{single}}$ is the single-pass accuracy on all problems, quantifies how much the model relies on clarification when it requests it. AC-RL exhibits larger drops than Binary-Reward RL despite making far fewer requests: $\Delta_{\text{deny}} = 14.91$ vs. 11.43 on MathVision and 7.94 vs. 2.66 on MathVerse. This suggests that AC-RL learns to distinguish between recoverable and irrecoverable information gaps: it produces self-sufficient captions when possible, but when it does request clarification, these requests target instances where critical visual details cannot be inferred from context alone.

Finally, a captioner trained on answer correctness could also learn to state the answer in its caption and act as a partial solver. We test this on MathVerse-VO by labeling each caption sentence for answer-like content, that is, final answers, solution steps and directives to the reasoner (Table 5). The pretrained captioner already states a final answer in 47.6% of its captions, despite the instruction not

Table 5: Caption content on MathVerse-VO. Final Answer and Any Answer-Like = fraction of captions with such sentences. Derived Values = fraction of numeric claims that are computed and not visible in the image. Visible Correct = fraction of visible values transcribed correctly. Filtered Acc. = reasoner accuracy after removing answer-like sentences.

Captioner	Final Answer	Any Answer-Like	Derived Values	Visible Correct	Filtered Acc.
Pretrained	47.64%	81.48%	19.24%	60.41%	20.76%
AC-RL	**18.01%**	**48.02%**	**8.49%**	**68.59%**	**27.15%**

to, and AC-RL lowers this to 18.0%. AC-RL captions also contain fewer computed values that do not appear in the image, and they transcribe the visible values more accurately, while being 40% shorter on average. Directives to the reasoner, which could be used to discourage clarification requests, do not become more frequent. When we remove all answer-like sentences from the captions of both models, AC-RL outperforms the pretrained captioner by 6.4 points, and by 9.0 points on the problems where neither captioner produces such content. The gains of AC-RL are therefore not carried by answer-like content (Appendix P).

4.4 Subject-Level Performance Analysis

We decompose performance on MathVision and DynaMath by subject area and stratify DynaMath by education level (elementary, high school, undergraduate), comparing AC-RL to the pretrained baseline (Qwen2.5-VL-3B + Reasoner).

Figure 2 visualizes the per-subject performance comparison. The analysis reveals that AC-RL's gains are concentrated in subjects that arguably depend heavily on precise quantitative and structural information. On MathVision, we observe the largest improvements in metric geometry for angles (+10.4 points), transformation geometry (+8.3 points), and algebra (+7.5 points). DynaMath shows even more pronounced gains in solid geometry (+18.7 points), algebra (+15.1 points), and puzzle tests (+13.5 points). These subjects require extracting specific numerical values, spatial relationships, or structural patterns from images. In contrast, performance differences are minimal in subjects that rely more on general visual understanding or pattern recognition.

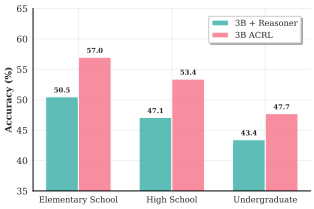


Figure 3: DynaMath average accuracy across education levels. AC-RL consistently outperforms the baseline regardless of problem difficulty.

Figure 3 shows that AC-RL maintains consistent improvements across all difficulty levels on DynaMath. The absolute gains remain relatively stable at 6.5, 6.3, and 4.3 points for elementary, high school, and undergraduate levels, respectively. While both models show expected degradation as problem complexity increases, AC-RL preserves its advantage by learning to extract critical visual details needed at each level. The smaller gain at the undergraduate level may reflect limits in what visual information alone can contribute to abstract problems.

5 Conclusion

Our results suggest that interface alignment between AI modules can be learned through RL with a simple tiered reward, without requiring explicit caption annotations or specialized architectures. We demonstrated this on visual reasoning as a controlled testbed where clear ground truth enables clean measurement. The core mechanism of penalizing clarification-dependent success to encourage information front-loading could naturally extend to other settings where specialized VLMs interface with frozen reasoners, such as medical imaging [24] or engineering diagrams [7]. Our results on WorldMedQA-V are a first step in this direction, obtained with general-purpose models. We leave validation with specialized models for future work. Other promising directions include bidirectional adaptation where both modules co-evolve, using clarification content (not just occurrence) as richer supervision, schedules for the partial reward α beyond the linear decay we tested, and multi-turn clarification with decaying rewards for iterative refinement.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025.

[3] Liangyu Chen, Bo Li, Sheng Shen, Jingkang Yang, Chunyuan Li, Kurt Keutzer, Trevor Darrell, and Ziwei Liu. Large language models are visual reasoning coordinators. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[4] Zhenfang Chen, Qinhong Zhou, Yikang Shen, Yining Hong, Zhiqing Sun, Dan Gutfreund, and Chuang Gan. Visual chain-of-thought prompting for knowledge-based visual reasoning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(2):1254–1262, Mar. 2024.

[5] 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.

[6] Yihe Deng, Hritik Bansal, Fan Yin, Nanyun Peng, Wei Wang, and Kai-Wei Chang. Openvlthinker: Complex vision-language reasoning via iterative sft-rl cycles, 2025.

[7] Anna C Doris, Md Ferdous Alam, Amin Heyrani Nobari, and Faez Ahmed. Cad-coder: An open-source vision-language model for computer-aided design code generation. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 89220, page V03AT03A031. American Society of Mechanical Engineers, 2025.

[8] Yifan Du, Junyi Li, Tianyi Tang, Wayne Xin Zhao, and Ji-Rong Wen. Zero-shot visual question answering with language model feedback. *arXiv preprint arXiv:2305.17006*, 2023.

[9] 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.

[10] Maksim Dzabraev, Alexander Kunitsyn, and Andrei Ivaniuta. Vlrm: Vision-language models act as reward models for image captioning. *arXiv preprint arXiv:2404.01911*, 2024.

[11] Google DeepMind. Gemini 3.1 flash-lite. `https://deepmind.google/technologies/gemini/`, 2026. Large language model.

[12] Yunhao Gou, Kai Chen, Zhili Liu, Lanqing HONG, Xin Jin, Zhenguo Li, James Kwok, and Yu Zhang. Reasoning-aligned perception decoupling for scalable multi-modal reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026.

[13] Yunhao Gou, Kai Chen, Zhili Liu, Lanqing Hong, Xin Jin, Zhenguo Li, James T. Kwok, and Yu Zhang. Perceptual decoupling for scalable multi-modal reasoning via reward-optimized captioning, 2025.

[14] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

[15] Dong Guo, Faming Wu, Feida Zhu, Fuxing Leng, Guang Shi, Haobin Chen, Haoqi Fan, Jian Wang, Jianyu Jiang, Jiawei Wang, Jingji Chen, Jingjia Huang, Kang Lei, Liping Yuan, Lishu Luo, Pengfei Liu, Qinghao Ye, Rui Qian, Shen Yan, Shixiong Zhao, Shuai Peng, Shuangye Li, Sihang Yuan, Sijin Wu, Tianheng Cheng, Weiwei Liu, Wenqian Wang, Xianhan Zeng, Xiao Liu,

Xiaobo Qin, Xiaohan Ding, Xiaojun Xiao, Xiaoying Zhang, Xuanwei Zhang, Xuehan Xiong, Yanghua Peng, Yangrui Chen, Yanwei Li, Yanxu Hu, Yi Lin, Yiyuan Hu, Yiyuan Zhang, Youbin Wu, Yu Li, Yudong Liu, Yue Ling, Yujia Qin, Zanbo Wang, Zhiwu He, Aoxue Zhang, Bairen Yi, Bencheng Liao, Can Huang, Can Zhang, Chaorui Deng, Chaoyi Deng, Cheng Lin, Cheng Yuan, Chenggang Li, Chenhui Gou, Chenwei Lou, Chengzhi Wei, Chundian Liu, Chunyuan Li, Deyao Zhu, Donghong Zhong, Feng Li, Feng Zhang, Gang Wu, Guodong Li, Guohong Xiao, Haibin Lin, Haihua Yang, Haoming Wang, Heng Ji, Hongxiang Hao, Hui Shen, Huixia Li, Jiahao Li, Jialong Wu, Jianhua Zhu, Jianpeng Jiao, Jiashi Feng, Jiaze Chen, Jianhui Duan, Jihao Liu, Jin Zeng, Jingqun Tang, Jingyu Sun, Joya Chen, Jun Long, Junda Feng, Junfeng Zhan, Junjie Fang, Junting Lu, Kai Hua, Kai Liu, Kai Shen, Kaiyuan Zhang, Ke Shen, Ke Wang, Keyu Pan, Kun Zhang, Kunchang Li, Lanxin Li, Lei Li, Lei Shi, Li Han, Liang Xiang, Liangqiang Chen, Lin Chen, Lin Li, Lin Yan, Liying Chi, Longxiang Liu, Mengfei Du, Mingxuan Wang, Ningxin Pan, Peibin Chen, Pengfei Chen, Pengfei Wu, Qingqing Yuan, Qingyao Shuai, Qiuyan Tao, Renjie Zheng, Renrui Zhang, Ru Zhang, Rui Wang, Rui Yang, Rui Zhao, Shaoqiang Xu, Shihao Liang, Shipeng Yan, Shu Zhong, Shuaishuai Cao, Shuangzhi Wu, Shufan Liu, Shuhan Chang, Songhua Cai, Tenglong Ao, Tianhao Yang, Tingting Zhang, Wanjun Zhong, Wei Jia, Wei Weng, Weihao Yu, Wenhao Huang, Wenjia Zhu, Wenli Yang, Wenzhi Wang, Xiang Long, XiangRui Yin, Xiao Li, Xiaolei Zhu, Xiaoying Jia, Xijin Zhang, Xin Liu, Xinchen Zhang, Xinyu Yang, Xiongcai Luo, Xiuli Chen, Xuantong Zhong, Xuefeng Xiao, Xujing Li, Yan Wu, Yawei Wen, Yifan Du, Yihao Zhang, Yining Ye, Yonghui Wu, Yu Liu, Yu Yue, Yufeng Zhou, Yufeng Yuan, Yuhang Xu, Yuhong Yang, Yun Zhang, Yunhao Fang, Yuntao Li, Yurui Ren, Yuwen Xiong, Zehua Hong, Zehua Wang, Zewei Sun, Zeyu Wang, Zhao Cai, Zhaoyue Zha, Zhecheng An, Zhehui Zhao, Zhengzhuo Xu, Zhipeng Chen, Zhiyong Wu, Zhuofan Zheng, Zihao Wang, Zilong Huang, Ziyu Zhu, and Zuquan Song. Seed1.5-vl technical report, 2025.

[16] Zixian Guo, Ming Liu, Qilong Wang, Zhilong Ji, Jinfeng Bai, Lei Zhang, and Wangmeng Zuo. Integrating visual interpretation and linguistic reasoning for math problem solving, 2025.

[17] Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14953–14962, 2023.

[18] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[19] Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*, 2025.

[20] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

[21] Yoonsik Kim, Moonbin Yim, and Ka Yeon Song. Tablevqa-bench: A visual question answering benchmark on multiple table domains, 2024.

[22] Kenton Lee, Mandar Joshi, Iulia Raluca Turc, Hexiang Hu, Fangyu Liu, Julian Martin Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. Pix2struct: Screenshot parsing as pretraining for visual language understanding. In *International Conference on Machine Learning*, pages 18893–18912. PMLR, 2023.

[23] Sicong Leng, Jing Wang, Jiaxi Li, Hao Zhang, Zhiqiang Hu, Boqiang Zhang, Yuming Jiang, Hang Zhang, Xin Li, Lidong Bing, et al. Mmr1: Enhancing multimodal reasoning with variance-aware sampling and open resources. *arXiv preprint arXiv:2509.21268*, 2025.

[24] Chunyuan Li, Cliff Wong, Sheng Zhang, Naoto Usuyama, Haotian Liu, Jianwei Yang, Tristan Naumann, Hoifung Poon, and Jianfeng Gao. Llava-med: Training a large language-and-vision assistant for biomedicine in one day. *Advances in Neural Information Processing Systems*, 36:28541–28564, 2023.

[25] Li Li, Yingzhe Peng, Xu Yang, Ruoxi Chen, Xu Haiyang, Yan Ming, and Huang Fei. L-clipscore: a lightweight embedding-based captioning metric for evaluating and training. *IEEE Transactions on Multimedia*, 2026.

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

[27] 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 *International Conference on Learning Representations (ICLR)*, 2024.

[28] Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *The 36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[29] Shiyin Lu, Yang Li, Qing-Guo Chen, Zhao Xu, Weihua Luo, Kaifu Zhang, and Han-Jia Ye. Ovis: Structural embedding alignment for multimodal large language model. *arXiv:2405.20797*, 2024.

[30] Yiting Lu, Jiakang Yuan, Zhen Li, Shitian Zhao, Qi Qin, Xinyue Li, Le Zhuo, Licheng Wen, Dongyang Liu, Yuewen Cao, Xiangchao Yan, Xin Li, Tianshuo Peng, Shufei Zhang, Botian Shi, Tao Chen, Zhibo Chen, Lei Bai, Peng Gao, and Bo Zhang. Omnicaptioner: One captioner to rule them all, 2025.

[31] João Matos, Shan Chen, Siena Kathleen V. Placino, Yingya Li, Juan Carlos Climent Pardo, Daphna Idan, Takeshi Tohyama, David Restrepo, Luis Filipe Nakayama, José María Millet Pascual-Leone, Guergana K Savova, Hugo Aerts, Leo Anthony Celi, An-Kwok Ian Wong, Danielle Bitterman, and Jack Gallifant. WorldMedQA-V: a multilingual, multimodal medical examination dataset for multimodal language models evaluation. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 7218–7231, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics.

[32] Runqi Qiao, Qiuna Tan, Guanting Dong, Minhui Wu, Chong Sun, Xiaoshuai Song, Zhuoma GongQue, Shanglin Lei, Zhe Wei, Miaoxuan Zhang, et al. We-math: Does your large multimodal model achieve human-like mathematical reasoning? *arXiv preprint arXiv:2407.01284*, 2024.

[33] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[34] Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

[35] Ayush Singh, Mansi Gupta, Shivank Garg, Abhinav Kumar, and Vansh Agrawal. Beyond captioning: Task-specific prompting for improved vlm performance in mathematical reasoning, 2024.

[36] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[37] ModelScope Team. EvalScope: Evaluation framework for large models, 2024.

[38] Qwen Team. Qvq: To see the world with wisdom, December 2024.

[39] Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`, 2020.

[40] Haozhe Wang, Chao Qu, Zuming Huang, Wei Chu, Fangzhen Lin, and Wenhu Chen. Vl-rethinker: Incentivizing self-reflection of vision-language models with reinforcement learning. *arXiv preprint arXiv:2504.08837*, 2025.

[41] Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with math-vision dataset. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

[42] Peiyu Wang, Yichen Wei, Yi Peng, Xiaokun Wang, Weijie Qiu, Wei Shen, Tianyidan Xie, Jiangbo Pei, Jianhao Zhang, Yunzhuo Hao, et al. Skywork r1v2: Multimodal hybrid reinforcement learning for reasoning. *arXiv preprint arXiv:2504.16656*, 2025.

[43] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution, 2024.

[44] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. Autogen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024.

[45] Changyi Xiao, Mengdi Zhang, and Yixin Cao. Bnpo: Beta normalization policy optimization, 2025.

[46] Yijia Xiao, Edward Sun, Tianyu Liu, and Wei Wang. Logicvista: Multimodal llm logical reasoning benchmark in visual contexts. *arXiv preprint arXiv:2407.04973*, 2024.

[47] Yi Yang, Xiaoxuan He, Hongkun Pan, Xiyan Jiang, Yan Deng, Xingtao Yang, Haoyu Lu, Dacheng Yin, Fengyun Rao, Minfeng Zhu, et al. R1-onevision: Advancing generalized multimodal reasoning through cross-modal formalization. *arXiv preprint arXiv:2503.10615*, 2025.

[48] Huanjin Yao, Jiaxing Huang, Wenhao Wu, Jingyi Zhang, Yibo Wang, Shunyu Liu, Yingjie Wang, Yuxin Song, Haocheng Feng, Li Shen, and Dacheng Tao. Mulberry: Empowering mllm with o1-like reasoning and reflection via collective monte carlo tree search. *CoRR*, abs/2412.18319, 2024.

[49] Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of CVPR*, 2024.

[50] Di Zhang, Jingdi Lei, Junxian Li, Xunzhi Wang, Yujie Liu, Zonglin Yang, Jiatong Li, Weida Wang, Suorong Yang, Jianbo Wu, et al. Critic-v: Vlm critics help catch vlm errors in multimodal reasoning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 9050–9061, 2025.

[51] Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, et al. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, pages 169–186. Springer, 2024.

[52] Shuai Zhao, Xiaohan Wang, Linchao Zhu, and Yi Yang. Test-time adaptation with CLIP reward for zero-shot generalization in vision-language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[53] Kaiwen Zhou, Kwonjoon Lee, Teruhisa Misu, and Xin Wang. ViCor: Bridging visual understanding and commonsense reasoning with large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10783–10795, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[54] Yue Zhou, Yada Zhu, Diego Antognini, Yoon Kim, and Yang Zhang. Paraphrase and solve: Exploring and exploiting the impact of surface form on mathematical reasoning in large language models. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2793–2804, Mexico City, Mexico, June 2024. Association for Computational Linguistics.

[55] 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.

[56] Chengke Zou, Xingang Guo, Rui Yang, Junyu Zhang, Bin Hu, and Huan Zhang. Dynamath: A dynamic visual benchmark for evaluating mathematical reasoning robustness of vision language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

A AC-RL Algorithm

We provide a formal specification of the Adaptive-Clarification Reinforcement Learning (AC-RL) algorithm. The algorithm operates in an episodic setting where each episode consists of a visual reasoning problem (I, Q) sampled from the dataset $\mathcal{D}$.

A.1 Formal Problem Setup

Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$ denote the Markov Decision Process where:

- $\mathcal{S} = \mathcal{I} \times \mathcal{Q} \times \mathcal{H}$ is the state space,
- $\mathscr{A} = \mathcal{C}$ is the action space (caption generation),
- $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition kernel,
- $R : \mathcal{T} \to [0, 1]$ is the reward function defined on trajectories,
- $\gamma = 1$ (undiscounted episodic setting).

A single episode proceeds as follows. At $t = 0$, the vision policy emits the initial caption $c_0 \sim \pi_\theta(\cdot \mid s_0)$ with $s_0 = (I, Q, \emptyset)$. The reasoner stochastically decides whether to request clarification and, if so, which question to ask; we denote this by a θ-independent kernel $q_1 \sim p(\cdot \mid s_0, c_0)$. When $q_1 \neq \emptyset$, the clarification caption is produced by a *frozen* checkpoint π_{ref}:

$$c_1 \sim \pi_{\text{ref}}(\cdot \mid s_1), \qquad s_1 = (I, Q, \{c_0, q_1\}),$$

and the reasoner produces a final answer according to a θ-independent kernel $A \sim p(\cdot \mid Q, c_0, (q_1, c_1))$. When $q_1 = \emptyset$, the reasoner answers from (Q, c_0) directly. The next state appends the sampled variables to the dialogue history. Thus P composes the reasoner's stochastic behavior and the frozen clarification-caption policy π_{ref}; conditioned on the agent's action c_0, these post-action mechanisms are *θ-independent* by construction. The episode terminates after the answer A is produced, and the reward is assigned as in the main text.

Clarification captioning is frozen. In all experiments, the clarification caption c_1 is generated by a *frozen* checkpoint π_{ref} (typically the reference policy). Its distribution does not change during training. Consequently, no gradients flow through π_{ref} or through the reasoner $\mathcal{R}$; only the log-probabilities of the initial caption tokens c_0 contribute to the policy update.

A.2 Advantage Computation

We use Beta-Normalization Policy Optimization (BNPO) [45] for advantage estimation. BNPO addresses a limitation of standard GRPO: while GRPO uses fixed normalization, BNPO adaptively normalizes rewards using a Beta distribution whose parameters evolve with the policy. This provides lower-variance gradient estimates and more stable training.

For a group of rewards $\{R^{(i)}\}_{i=1}^{M}$ from a single prompt, BNPO fits Beta distribution parameters $(\alpha_\beta, \beta_\beta)$ via method-of-moments from the group statistics, then computes advantages as:

$$A^{(i)}_{\text{BNPO}} = \frac{R^{(i)} - \mu_\beta}{\sigma_\beta + \epsilon}, \quad \text{where } \mu_\beta = \frac{\alpha_\beta}{\alpha_\beta + \beta_\beta} \tag{3}$$

Although BNPO was designed for binary rewards, we found it effective with our ternary reward structure, moderately outperforming standard GRPO in our experiments.

A.3 Policy Update

The policy is updated using the clipped surrogate objective with a fixed KL reference:

$$\mathcal{L}_{\text{clip}}(\theta) = -\mathbb{E}_{s_t, a_t}\Big[\min\Big(r_t(\theta)A_t, \\ \text{clip}\big(r_t(\theta), 1-\epsilon, 1+\epsilon\big)\, A_t\Big)\Big] + \beta\, D_{\text{KL}}(\pi_\theta \,\|\, \pi_{\text{KL}})\,.$$

where $r_t(\theta) = \pi_\theta(a_t \mid s_t)/\pi_{\theta_{\text{old}}}(a_t \mid s_t)$, and A_t is the advantage computed via BNPO [45]. The gradient is computed solely on the initial captioning segments c_0; the clarification responses c_1 are emitted by the frozen π_{ref} and are thus θ-independent.

A.4 Training Algorithm

Algorithm 1 Adaptive-Clarification Reinforcement Learning (AC-RL)

Require: Dataset $\mathcal{D}$, vision model $\mathcal{V}_\theta$, reasoner $\mathcal{R}$, penalty α, group size M, gradient steps K

Initialize policy $\pi_\theta \leftarrow \mathcal{V}_\theta$ with parameters θ
Initialize *frozen* clarification captioner π_{ref} (checkpoint used only for c_1)
Initialize fixed KL reference $\pi_{\text{ref}} \leftarrow \pi_\theta$
for iteration $t = 1$ to T **do**
  Sample batch $\mathcal{B} = \{(I_j, Q_j)\}_{j=1}^{B} \sim \mathcal{D}$
  for each $(I_j, Q_j) \in \mathcal{B}$ **do**
    for $i = 1$ to M **do**
      Generate initial caption: $c_0^{(i,j)} \sim \pi_\theta(\cdot \mid I_j, Q_j)$
      Sample reasoner's clarification decision: $q_1^{(i,j)} \sim \mathcal{R}_{\text{clarify}}(\cdot \mid Q_j, c_0^{(i,j)})$ (no gradients)
      if $q_1^{(i,j)} \neq \emptyset$ **then**
        Clarification from *frozen* checkpoint: $c_1^{(i,j)} \sim \pi_{\text{ref}}(\cdot \mid I_j, Q_j, (c_0^{(i,j)}, q_1^{(i,j)}))$ (no gradients)
        Get answer: $A^{(i,j)} \sim \mathcal{R}(\cdot \mid Q_j, c_0^{(i,j)}, (q_1^{(i,j)}, c_1^{(i,j)}))$ (no gradients)
        Set clarification flag: $C^{(i,j)} = 1$
      else
        Get answer: $A^{(i,j)} \sim \mathcal{R}(\cdot \mid Q_j, c_0^{(i,j)})$ (no gradients)
        Set clarification flag: $C^{(i,j)} = 0$
      end if
      Compute reward: $R^{(i,j)} = \begin{cases} 1.0 & \text{if correct}(A^{(i,j)}) \wedge C^{(i,j)} = 0 \\ \alpha & \text{if correct}(A^{(i,j)}) \wedge C^{(i,j)} = 1 \\ 0 & \text{otherwise} \end{cases}$
    end for
    Fit Beta parameters $(\alpha_\beta^{(j)}, \beta_\beta^{(j)})$ to $\{R^{(i,j)}\}_{i=1}^{M}$
    Compute BNPO advantages $\{A_{\text{BNPO}}^{(i,j)}\}_{i=1}^{M}$
  end for
  for $k = 1$ to K **do**
    Update policy with clipping and KL penalty: $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_{\text{clip}}(\theta; \pi_{\text{ref-KL}})$
  end for
end for

B Unbiasedness of the Three-Tier Reward

In this section, we provide a formal proof that our three-tier reward structure maintains the unbiasedness property of the REINFORCE policy gradient estimator, even when the reasoner exhibits stochasticity.

Theorem B.1 (Unbiasedness of the Three-Tier Reward with Stochastic Reasoner)**.** *Let $\xi \sim p(\cdot \mid \tau)$ denote all post-action randomness after the policy chooses its actions (e.g., the reasoner's sampling noise and, when clarification is used, the frozen clarification-caption sampling). Define the extended trajectory $\tilde{\tau} = (\tau, \xi)$ with joint density:*

$$p_\theta(\tilde{\tau}) = p_\theta(\tau) \cdot p(\xi \mid \tau)$$

where $p(\xi \mid \tau)$ is independent of θ.

Let the tiered reward function be defined as:

$$R_{\textit{tier}}(\tilde{\tau}) = \begin{cases} 1 & \textit{if correct}(A(\tilde{\tau})) \wedge C(\tilde{\tau}) = 0 \\ \alpha & \textit{if correct}(A(\tilde{\tau})) \wedge C(\tilde{\tau}) > 0 \\ 0 & \textit{otherwise} \end{cases}$$

where $\alpha \in (0, 1)$, and define the training objective:

$$J(\theta) = \mathbb{E}_{\tilde{\tau} \sim p_\theta}[R_{\textit{tier}}(\tilde{\tau})].$$

For any baseline $b_t(s_t)$ that does not depend on the action a_t, the REINFORCE estimator:

$$\hat{g}(\tilde{\tau}) = \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t \mid s_t) \cdot (R_{\textit{tier}}(\tilde{\tau}) - b_t(s_t))$$

satisfies $\mathbb{E}_{\tilde{\tau} \sim p_\theta}[\hat{g}(\tilde{\tau})] = \nabla_\theta J(\theta)$, i.e., the policy gradient remains unbiased despite the $0/\alpha/1$ reward shaping and post-action stochasticity.

Proof. **Step 1: Setup.** The extended trajectory $\tilde{\tau} = (\tau, \xi)$ includes both the policy-generated trajectory $\tau = (s_0, a_0, s_1, a_1, ..., s_T)$ and the post-action randomness ξ. The joint probability decomposes as:

$$p_\theta(\tilde{\tau}) = p(s_0) \prod_{t=0}^{T-1} \pi_\theta(a_t \mid s_t)\, P(s_{t+1} \mid s_t, a_t) \cdot p(\xi \mid \tau),$$

where $p(s_0)$ is the initial state distribution, P is the environment transition kernel, and $p(\xi \mid \tau)$ is the distribution over post-action randomness given the trajectory; by assumption, $p(\xi \mid \tau)$ is θ-independent.

Step 2: Policy Gradient Theorem. For $R_{\text{tier}}(\tilde{\tau})$,

$$\begin{aligned} \nabla_\theta J(\theta) &= \nabla_\theta \int p_\theta(\tau)\, p(\xi \mid \tau)\, R_{\text{tier}}(\tilde{\tau})\, d\xi\, d\tau \\ &= \int p_\theta(\tau)\, p(\xi \mid \tau)\, R_{\text{tier}}(\tilde{\tau})\, \nabla_\theta \log p_\theta(\tau)\, d\xi\, d\tau \\ &= \int p_\theta(\tilde{\tau})\, R_{\text{tier}}(\tilde{\tau})\, \nabla_\theta \log p_\theta(\tau)\, d\tilde{\tau}, \end{aligned}$$

using that $\nabla_\theta \log p_\theta(\tilde{\tau}) = \nabla_\theta \log p_\theta(\tau) + \nabla_\theta \log p(\xi \mid \tau)$ and $\nabla_\theta \log p(\xi \mid \tau) = 0$ by θ-independence. Since $p(s_0)$ and P are θ-independent,

$$\nabla_\theta \log p_\theta(\tau) = \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t \mid s_t),$$

hence

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tilde{\tau} \sim p_\theta} \left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t \mid s_t) \cdot R_{\text{tier}}(\tilde{\tau}) \right].$$

Step 3: Baseline Subtraction. For any $b_t(s_t)$ not depending on a_t,

$$\mathbb{E}_{\tilde{\tau}\sim p_\theta}\left[\sum_{t=0}^{T-1}\nabla_\theta \log \pi_\theta(a_t \mid s_t)\cdot b_t(s_t)\right]$$
$$=\sum_{t=0}^{T-1}\mathbb{E}_{s_t}\left[b_t(s_t)\cdot \mathbb{E}_{a_t\sim\pi_\theta(\cdot|s_t)}\left[\nabla_\theta \log \pi_\theta(a_t \mid s_t)\right]\right]=0,$$

so

$$\mathbb{E}_{\tilde{\tau}\sim p_\theta}\left[\sum_{t=0}^{T-1}\nabla_\theta \log \pi_\theta(a_t \mid s_t)\cdot (R_{\text{tier}}(\tilde{\tau})-b_t(s_t))\right]$$
$$=\nabla_\theta J(\theta).$$

□

Remark B.2 (If the reasoner is θ-dependent). If, instead, $p(\xi \mid \tau)$ depends on θ (e.g., shared trunk), then

$$\nabla_\theta \log p_\theta(\tilde{\tau}) = \sum_t \nabla_\theta \log \pi_\theta(a_t \mid s_t) \;+\; \nabla_\theta \log p_\theta(\xi \mid \tau),$$

and an unbiased estimator must add the extra score term $\nabla_\theta \log p_\theta(\xi \mid \tau)$ multiplied by the same return. Alternatively, one may stop gradients through the reasoner or generate ξ using frozen modules to enforce θ-independence.

Proposition B.3 (Unbiased gradient with θ-dependent reasoner)**.** *If $p_\theta(\xi \mid \tau)$ depends on θ, then an unbiased estimator of $\nabla_\theta J(\theta)$ is*

$$\hat{g}(\tilde{\tau}) = \left[\sum_{t=0}^{T-1}\nabla_\theta \log \pi_\theta(a_t \mid s_t) + \nabla_\theta \log p_\theta(\xi \mid \tau)\right] R_{\textit{tier}}(\tilde{\tau}),$$

i.e., the score-function estimator must include the additional term $\nabla_\theta \log p_\theta(\xi \mid \tau)$ from the post-action randomness. Each term may be combined with a baseline that is independent of the respective sampled variable.

Corollary B.4. *Under the θ-independence of $p(\xi \mid \tau)$, the three-tier reward preserves the unbiasedness of the REINFORCE estimator for $\nabla_\theta J(\theta)$. Algorithms such as PPO, GRPO, and BNPO, which optimize clipped or normalized surrogate objectives, remain applicable with this reward; however, their gradient estimates are generally biased (by design) and converge to stationary points of their respective surrogate objectives rather than guaranteeing an unbiased gradient of $J(\theta)$.*

C Datasets and Baselines

We evaluate on seven benchmarks that span diverse visual contexts beyond symbolic mathematics: natural images, statistical charts, scientific diagrams, tables, real-world scenes, puzzles, and pattern recognition tasks. This diversity ensures our evaluation captures a broad range of visual reasoning challenges.

- **MathVista** (testmini) [27]: A consolidated benchmark of 1000 examples, covering figure QA, geometry, math word problems, and textbook QA across diverse reasoning types.
- **MathVision** [41]: 3040 problems sourced from real math competitions, spanning 16 mathematical disciplines across 5 difficulty levels.
- **MathVerse** [51]: 2612 visual math problems transformed into 6 versions with varying visual vs. textual information to test genuine diagram comprehension.
- **MMMU** (dev & validation) [49]: 1050 college-level questions across 6 disciplines and 30 image types including charts, diagrams, tables, and scientific figures.
- **WeMath** (testmini, strict) [32]: 1740 visual math problems organized around 67 hierarchical knowledge concepts, decomposing composite problems into sub-problems to assess foundational knowledge.
- **DynaMath** (worst-case) [56]: 501 seed questions represented as Python programs generating variants with different numerical values or geometric transformations to test reasoning robustness.
- **LogicVista** [46]: 448 questions evaluating logical reasoning across 5 types (deductive, inductive, spatial, numerical, mechanical) sourced from human IQ and reasoning test banks.

For the additional benchmarks (Appendix N), we use:

- **WorldMedQA-V** (English) [31]: 568 medical licensing exam questions, each paired with a clinical image.
- **TableVQA-Bench** [21]: questions over table images from four domains; we use a uniform random sample of 500 of its 1,500 questions.
- **ScienceQA** (test) [28]: 2017 multiple-choice science questions with an image.

Model Baselines. We compare against leading proprietary models: GPT-4o [1], Claude-3.7-Sonnet, Gemini-2.0-Flash [36], o1 [20], Gemini 2.5 Pro [5], and Seed1.5-VL [15]; open-weights general-purpose models: Qwen2.5-VL [2] and Ovis2 [29]; and reasoning-optimized models: InternVL3-MPO variants [55], VL-Rethinker [40], QVQ-72B-Preview [38], and MMR1-Math [23].

D Generation Diversity During Training

An interesting emergent property of AC-RL training is that it maintains greater generation diversity compared to standard binary-reward RL. Figure 4 tracks the fraction of training batches where all M generated captions receive identical rewards (zero standard deviation), which serves as an indicator of diversity collapse.

Both methods show an upward trend as entropy naturally decreases during policy optimization. However, AC-RL consistently maintains a lower fraction of uniform-reward batches (approximately 0.31 vs 0.42 at convergence). This difference likely stems from the tiered reward structure: while standard RL only distinguishes between success and failure, AC-RL's intermediate reward ($\alpha = 0.7$) creates a richer gradient landscape that encourages the model to explore different captioning strategies.

E Sensitivity to Clarification Penalty

The clarification penalty $(1 - \alpha)$ controls the reward reduction for success requiring clarification. Table 6 shows results across penalty values $\{0.1, 0.3, 0.5, 0.7\}$ on LogicVista and MathVision.

Intermediate penalties (0.3, 0.5) consistently outperform extremes, with near-identical MathVision performance demonstrating robustness. Penalties that are too weak (0.1) provide insufficient optimization pressure for front-loading information, while penalties that are too strong (0.7, rewarding

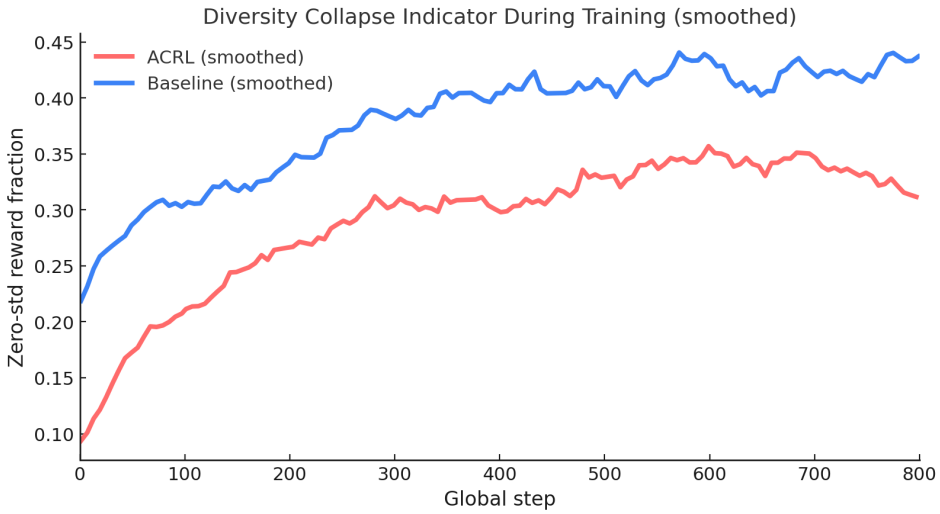


Figure 4: Fraction of uniform-reward batches during training. AC-RL (red) maintains lower values than standard RL (blue), indicating more diverse caption generation throughout training. Both methods show increasing trends as policies converge, but AC-RL's tiered reward structure preserves more exploration.

Table 6: Sensitivity to clarification penalty $(1 - \alpha)$.

Penalty	**LVista**	**MVision**
0.1	53.2	35.8
0.3	**55.7**	**40.2**
0.5	52.4	40.1
0.7	50.1	35.5

0.3 on clarified success) approach binary rewards, losing the partial credit benefits that enable denser learning signals.

We also train AC-RL with α decaying linearly from 0.9 to 0.5 over training, which crosses the fixed value 0.7 halfway. This run uses the configuration of the clarifier controls (Appendix K). It reaches 50.8 on LogicVista and 37.9 on MathVision, below the fixed $\alpha = 0.7$ on both benchmarks. Other schedules may do better, but this suggests that a schedule is an additional design choice and that the fixed value is a good default.

F Training Dynamics

Figure 5 tracks the (smoothed) frequency of clarification requests during training. The consistent decrease confirms our hypothesis that AC-RL teaches the captioner to front-load information that would otherwise trigger follow-up questions. Combined with the maintained generation diversity shown in Appendix D, this suggests the model finds genuinely informative captioning strategies rather than mode-collapsing to a narrow set of templates.

G Statistical Significance

We compute paired bootstrap confidence intervals for the comparisons supporting the main claims of the paper. For each (model A, model B, benchmark) triple, we resample test examples with replacement and compute $\Delta = \text{acc}(A) - \text{acc}(B)$ on the same resampled indices. We report the point estimate of Δ, the 95% bootstrap confidence interval, and a one-sided p-value for the hypothesis $\text{acc}(A) > \text{acc}(B)$, with $B = 10{,}000$ bootstrap resamples. A 95% CI excluding zero corresponds to significance at $\alpha = 0.05$ in either direction; we mark such cells with ★. Resampling is performed

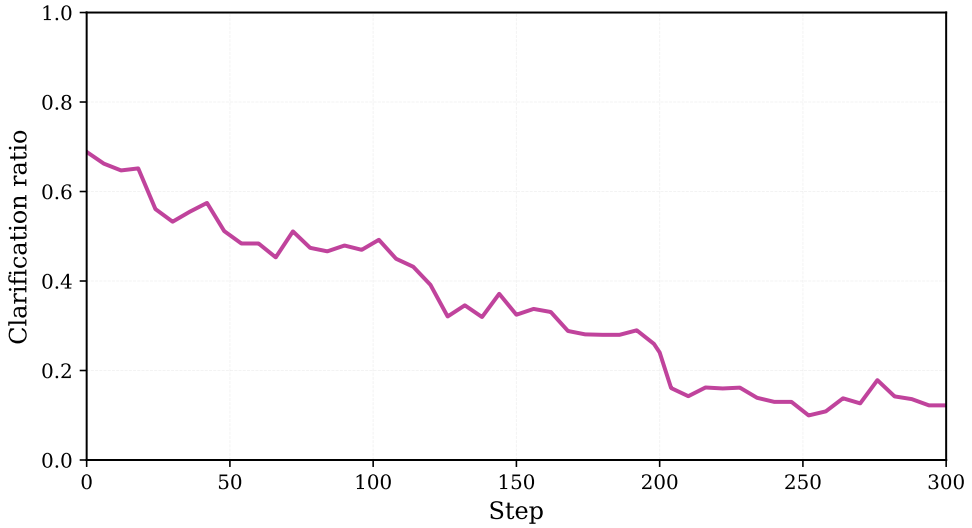


Figure 5: Clarification request frequency during AC-RL training. The rate decreases from approximately 65% at initialization to 20% at convergence, demonstrating that the model learns to produce more self-sufficient initial captions.

at the example level for benchmarks without grouping structure, and at the question-group level for DynaMath (where each seed question generates multiple variants). We do not report variance over training seeds due to compute cost; the bootstrap intervals reported here capture sampling variability of the test sets but not the variability of the training procedure itself.

Table 7 reports the paired bootstrap test for the AC-RL improvement over the pretrained captioner under the open-source 32B reasoner. AC-RL gains are significant at 95% on six of seven benchmarks; only WeMath shows no statistically resolvable difference.

Table 7: AC-RL + 32B Reasoner vs Pretrained + 32B Reasoner (Qwen2.5-VL-3B captioner). Δ values are accuracy point differences.

Benchmark	n	**AC-RL**	**Pretrained**	Δ	**95% CI**	**Sig.**
MathVista	1000	63.82	59.72	$+4.10$	$[+2.90, +5.40]$	★
MathVision	3040	36.84	32.73	$+4.11$	$[+3.42, +4.84]$	★
MathVerse-VO	788	34.38	29.21	$+5.17$	$[+1.65, +8.88]$	★
MMMU	900	57.79	55.24	$+2.56$	$[+1.56, +3.67]$	★
WeMath	1740	33.22	34.71	-1.49	$[-4.02, +1.03]$	—
DynaMath	501	24.73	14.17	$+10.55$	$[+7.78, +13.57]$	★
LogicVista	447	53.01	47.19	$+5.83$	$[+3.80, +8.05]$	★

Table 8 reports the same comparison under the proprietary Gemini reasoner. All seven differences are significant at 95%, including WeMath ($\Delta = +2.83$), where the open-source reasoner showed no resolvable difference.

Table 9 isolates the contribution of the clarification scaffold by comparing AC-RL to Binary-Reward RL under the 32B reasoner. Both methods share architecture, optimizer, and training data and differ only in the reward structure. AC-RL's improvement is significant at 95% on five of seven benchmarks; MathVerse is marginal ($p = 0.041$, just outside the threshold), and WeMath again shows no resolvable difference.

Across the headline comparisons, AC-RL's improvements over the pretrained captioner are statistically significant on 13 of 14 cells, with the only exception being WeMath under the 32B reasoner. The decomposition further shows that AC-RL significantly outperforms Binary-Reward RL on five of

Table 8: AC-RL + Gemini Reasoner vs Pretrained + Gemini Reasoner (Qwen2.5-VL-3B captioner).

Benchmark	n	**AC-RL**	**Pretrained**	Δ	**95% CI**	**Sig.**
MathVista	1000	66.12	61.82	$+4.29$	$[+3.10, +5.60]$	★
MathVision	3040	40.50	37.89	$+2.60$	$[+2.04, +3.16]$	★
MathVerse-VO	788	39.46	31.87	$+7.59$	$[+4.06, +11.17]$	★
MMMU	900	66.79	62.68	$+4.11$	$[+2.89, +5.44]$	★
WeMath	1740	34.43	31.60	$+2.83$	$[+0.57, +5.06]$	★
DynaMath	501	20.16	10.39	$+9.77$	$[+6.39, +13.17]$	★
LogicVista	447	54.35	49.43	$+4.92$	$[+3.13, +6.94]$	★

Table 9: AC-RL + 32B Reasoner vs Binary-Reward RL + 32B Reasoner (Qwen2.5-VL-3B captioner). Both methods use the same captioner, optimizer, and data; only the reward structure differs.

Benchmark	n	**AC-RL**	**Binary-RL**	Δ	**95% CI**	**Sig.**
MathVista	1000	63.82	62.62	$+1.20$	$[+0.60, +1.90]$	★
MathVision	3040	36.84	34.31	$+2.53$	$[+1.97, +3.09]$	★
MathVerse-VO	788	34.38	31.09	$+3.29$	$[-0.38, +6.98]$	—
MMMU	900	57.79	55.46	$+2.34$	$[+1.33, +3.33]$	★
WeMath	1740	33.22	33.44	-0.22	$[-2.70, +2.24]$	—
DynaMath	501	24.73	17.56	$+7.17$	$[+3.99, +10.38]$	★
LogicVista	447	53.01	47.41	$+5.60$	$[+3.58, +7.83]$	★

seven benchmarks, isolating the contribution of the clarification scaffold beyond what RL training alone provides.

Table 10 reports the same test for the clarifier controls of Section 4.2.4 and the decaying-α run of Appendix E, with AC-RL as model A. The MathVision differences are computed from per-example scores, which differ from the aggregate scores in Table 4 by up to 0.13 points. All runs are significantly below AC-RL on MathVision. On LogicVista, all runs except the weaker clarifier and the decaying schedule are significantly below AC-RL.

Table 10: AC-RL vs the clarifier controls, the decaying-α run and binary-reward RL trained in the same configuration (32B reasoner). Δ = AC-RL minus the control.

	LogicVista ($n = 447$)			**MathVision** ($n = 3040$)		
Control	Δ	**95% CI**	**Sig.**	Δ	**95% CI**	**Sig.**
Weaker clarifier	$+2.91$	$[-1.57, +7.38]$	—	$+2.27$	$[+0.56, +4.01]$	★
Permuted rewards	$+5.59$	$[+0.89, +10.29]$	★	$+2.14$	$[+0.39, +3.95]$	★
Uninformative clarifier	$+8.95$	$[+4.47, +13.42]$	★	$+3.19$	$[+1.38, +5.00]$	★
Decaying α	$+4.92$	$[+0.00, +9.84]$	—	$+2.43$	$[+0.69, +4.18]$	★
Binary reward	$+7.83$	$[+2.91, +12.75]$	★	$+2.20$	$[+0.46, +3.98]$	★

Table 11 reports the comparisons on the additional benchmarks (Appendix N). Confidence intervals for the caption filtering and the multi-round analysis are given in Appendices P and Q.

Table 11: AC-RL vs pretrained captioner on the additional benchmarks (32B reasoner). WorldMedQA-V and TableVQA-Bench use the checkpoint from Appendix E, ScienceQA the checkpoint of Table 1.

Benchmark	n	**AC-RL**	**Pretrained**	Δ	**95% CI**	**Sig.**
WorldMedQA-V	568	58.27	50.18	$+8.10$	$[+3.87, +12.32]$	★
TableVQA-Bench	500	42.40	31.80	$+10.60$	$[+6.60, +15.00]$	★
ScienceQA	2017	86.17	85.03	$+1.14$	$[-0.25, +2.53]$	—

H WeMath: Per-Concept Analysis

WeMath is organized around 67 hierarchical textbook knowledge concepts (e.g., "area of triangle", "Pythagorean theorem", "coordinate reading"). Each problem is annotated with the number of distinct concepts it requires (1, 2, or 3), and composite multi-concept problems are additionally decomposed into per-concept sub-problems that are evaluated separately. We use the strict accuracy as in our main results.

The most direct test of whether the clarification signal causes the WeMath regression is the AC-RL vs. Binary-RL comparison, since both methods share architecture, optimizer, and data and differ only in the reward structure. Across all other six benchmarks, AC-RL outperforms Binary-RL by $+3.68$ points on average. On WeMath, the gap is only -0.23 points (Binary-RL: 33.45%, AC-RL: 33.22%). Both RL methods regress by similar amounts relative to the pretrained baseline (Binary-RL: -1.26, AC-RL: -1.49). The clarification signal is therefore approximately neutral on WeMath rather than harmful.

Decomposing by required-concept count provides further evidence. On 1-concept problems, both methods regress by similar amounts. On 2-concept problems, AC-RL outperforms Binary-RL by $+2.78$ points. On 3-concept problems, both methods regress more sharply, but in tandem. The regression scales with concept-chaining complexity for both reward structures.

To test whether caption quality is the bottleneck for WeMath at all, we hold the 32B reasoner fixed and replace the 3B captioner with Qwen2.5-VL-32B without applying any RL. This isolates the caption-quality axis. Under this scaling, MathVision improves by $+11.9$ points and DynaMath by $+22.2$ points, while WeMath moves by only $+2.3$ points. The cross-benchmark pattern is consistent: methods that improve caption quality (whether through better captioner scale or through better reward signals) help most on benchmarks where visual extraction is the bottleneck and barely move benchmarks bottlenecked by the reasoner's domain knowledge. AC-RL optimizes what the reasoner sees, not what it knows.

I Captioner Scaling: 7B Inference

To verify that the caption-then-reason interface remains beneficial at larger captioner scales, we evaluate Qwen2.5-VL-7B in two configurations without applying any AC-RL training: as a standalone model answering directly, and as a captioner paired with the same DeepSeek-R1-Qwen-32B reasoner used throughout the paper.

Table 12: Qwen2.5-VL-7B as standalone solver vs. as a (non-RL-trained) captioner paired with the 32B reasoner. The two-stage configuration improves on 6 of 7 benchmarks despite no captioner fine-tuning, with the largest gains on the same extraction-heavy benchmarks where AC-RL provides the most value at 3B scale.

Config	MVista	MVision	MVerse	MMMU	WeM	DynaM	LVista	AVG
7B Standalone	70.3	25.8	41.0	57.3	34.4	19.4	46.1	42.0
7B + Reasoner	66.4	37.1	45.6	59.1	36.2	21.6	53.0	45.6
Gap	-3.9	+11.3	+4.6	+1.8	+1.8	+2.2	+6.9	+3.6

The two-stage setup improves over the standalone 7B on 6 of 7 benchmarks without any captioner training, with an average gain of $+3.6$ points. The largest improvements are on MathVision, LogicVista, and MathVerse, the same benchmarks where AC-RL provides the most value at 3B scale. We do not apply AC-RL training to the 7B model; this experiment establishes only that the interface bottleneck addressed by AC-RL persists at larger captioner scales.

J Image-Text Similarity Densification

This appendix details the alternative-densification ablation summarized in Section 4.2.4. The setup tests whether AC-RL's gains are attributable to reward densification broadly or to the clarification signal specifically.

We augment the binary correctness reward with image-text similarity computed by Qwen3-VL-Embedding-2B [26]: $R_i = R_{\text{correct},i} + \lambda \cdot \text{clamp}(\text{sim}(I, c_i), 0, 1)$, where c_i is the caption and I is the image. We calibrate λ on a held-out sample of 256 training traces by sweeping $\lambda \in \{0.1, 0.2, 0.4, 0.6\}$ and selecting the value where the correctness component contributes approximately 70% of total reward and the similarity component approximately 30% at convergence, following established practice for mixing task rewards with embedding-based caption quality signals [25]. This yields $\lambda = 0.4$.

Before training, we verified that the similarity signal is discriminative: matched image-caption pairs from the training set average 0.60, captions paired with mismatched images average 0.42, and random text averages 0.20. The signal therefore has the expected ordering, even if it does not directly measure task-relevance.

All other training details (BNPO optimizer, ViRL-39K data, decoupled architecture, KL regularization, hyperparameters) match Binary-RL and AC-RL. Checkpoints were evaluated every 25 steps on the average across all seven benchmarks; we report the checkpoint with the highest held-out average.

Table 13: Per-benchmark comparison of binary, similarity-densified, and clarification-densified reward structures, all using the same captioner, optimizer, and training data.

Method	**MVista**	**MVision**	**MVerse**	**MMMU**	**WeM**	**DynaM**	**LVista**
Binary-RL	62.6	34.3	31.1	55.4	33.5	17.6	47.4
Binary + Similarity	62.0	36.5	23.7	57.4	32.1	19.8	51.9
AC-RL	63.8	36.8	34.4	57.7	33.2	24.8	53.0

AC-RL outperforms similarity-densified RL on benchmarks designed to probe task-specific visual extraction (DynaMath, MathVerse, LogicVista, MathVista), while the two methods are comparable on benchmarks where generic caption quality is closer to the bottleneck (MathVision, MMMU, WeMath). The improvement of similarity-densified RL over plain Binary-RL on benchmarks like LogicVista ($+4.5$) confirms that some of AC-RL's gain is attributable to densification per se; the further $+5.0$ gap to AC-RL on DynaMath, the most extraction-heavy benchmark, establishes that the clarification signal carries information beyond what generic visual fidelity provides. The pattern is most extreme on MathVerse, where similarity-densified RL (23.7) underperforms even plain Binary-RL (31.1): when problems hinge on extracting specific values from a sparse diagram, optimizing for whole-image fidelity actively pulls the captioner away from what the reasoner needs. Similarity rewards optimize for whether the caption matches the image, whereas clarification-based rewards optimize for whether the caption contains what the reasoner specifically needs. The two signals are not in opposition; combining them is a natural direction for future work.

Two further observations, from the checkpoints of Table 1, point the same way. First, the problems that AC-RL and Binary-RL newly solve overlap only partly. Across LogicVista, DynaMath, MathVerse-VO, MMMU and WeMath (9,035 problems, counting DynaMath variants separately), AC-RL solves 1,272 problems that the pretrained captioner fails, and 623 of them (49.0%) are not solved by Binary-RL. If the clarification reward only made the binary signal denser, we would expect the two methods to improve on largely the same problems. Second, on the non-mathematical subjects of MMMU, AC-RL improves over the pretrained captioner while Binary-RL lowers accuracy (Appendix N).

K Clarifier Controls

This appendix details the clarifier controls summarized in Section 4.2.4. Each control changes one part of the clarification scaffold and keeps the rest fixed. The three controls, the binary-reward run in Table 4 and the decaying-α run of Appendix E share the training data and hyperparameters of AC-RL, use an effective batch size of 256, and are evaluated after 200 optimizer steps. The AC-RL reference is the checkpoint from Appendix E, which is also evaluated after 200 steps but trained with an effective batch size of 512.

The weaker-clarifier run replaces the frozen clarifier with Qwen2-VL-2B-Instruct [43], a previous-generation model, while the captioner is still initialized from Qwen2.5-VL-3B. Since the partial reward requires the final answer to be correct, a weaker clarifier mainly lowers how often the partial reward is earned, which moves training toward the binary-reward setting.

The uninformative-clarifier run replaces every clarification response with the fixed string "I cannot provide additional details about the image." The reasoner can still request clarification, and the reward rule and tier values are unchanged, so only the information in the response is removed.

The permuted-reward run computes the rewards as in AC-RL and then, at every step, randomly shuffles the positive values (1 and α) among the successful rollouts of each group. Failed rollouts keep a reward of 0. A caption that the reasoner solved directly can thus receive α, and one that needed clarification can receive 1. Each group keeps the same set of reward values, so its mean and spread, the sizes of its advantages, and whether it contributes a gradient at all are the same as under AC-RL. Only the correspondence between the partial reward and clarification is removed.

Table 10 in Appendix G gives paired bootstrap tests of each run against AC-RL.

L Cross-Reasoner Generalization

We test whether AC-RL learns reasoner-specific shortcuts or general captioning strategies by pairing the AC-RL captioner with reasoners unseen during training. To keep the cross-reasoner comparison internally consistent, we use a single AC-RL checkpoint trained at a higher effective batch size (the same checkpoint used for the sensitivity sweep in Appendix E). AC-RL values are slightly above the corresponding cells in Table 1 but we focus on the relative deltas.

We pair the pretrained captioner and the AC-RL captioner with three reasoners: the training reasoner DeepSeek-R1-Qwen-32B, an unseen smaller open-weights reasoner DeepSeek-R1-0528-Qwen3-8B, and OpenAI's GPT-5 mini (gpt-5-mini-2025-08-07) as a proprietary API reasoner. The pairings with GPT-5 mini are distinct from the end-to-end Gemini training reported in Table 1 and detailed in Appendix M: here the captioner was trained against the open-source 32B reasoner and is only paired with GPT-5 mini at inference, which tests whether the learned captioning strategy transfers to an unseen proprietary reasoner without retraining.

Table 14: Cross-reasoner generalization with the checkpoint from Appendix E. AC-RL improves over the pretrained captioner under all three reasoners. The gaps under the unseen 8B reasoner are comparable to those under the training 32B reasoner, and they are smaller under GPT-5 mini.

	32B Reasoner		**8B Reasoner (unseen)**		**GPT-5 mini (unseen)**	
Captioner	LVista	MVision	LVista	MVision	LVista	MVista
Pretrained	47.2	32.8	44.3	30.0	48.1	62.2
AC-RL	55.7	40.2	53.0	36.5	51.0	66.1
Gap	+8.5	+7.4	+8.7	+6.5	+2.9	+3.9

The deltas are positive across all three reasoners. AC-RL transfers cleanly to the unseen 8B reasoner with gains comparable to the training reasoner, and it transfers to a proprietary API reasoner whose internals are entirely opaque. Combined with the end-to-end training against Gemini reported in Table 1 and detailed in Appendix M, this indicates the captioning strategies AC-RL learns are not idiosyncratic to a single reasoner.

M End-to-End Training with Gemini

This appendix details the training setup for the AC-RL + Gemini Reasoner row in Table 1. We replaced DeepSeek-R1-Qwen-32B with Google's Gemini 3.1 Flash-Lite [11] (accessed via OpenRouter) as the frozen reasoner during both training and evaluation, while keeping all other configuration identical to the open-source setup. The clarification captioner is local (Qwen2.5-VL-3B frozen reference), so only the reasoner traffic touches the API.

Each rollout makes one Gemini call for the initial reasoner pass and, when the reasoner requests clarification, one additional call to produce the post-clarification answer. Over 480 optimizer steps, the run made approximately 25,000 Gemini API calls in total; the clarification-path calls account for roughly 23% of this volume, an overhead that is front-loaded since the clarification rate decreases from $\sim$42% early in training to $\sim$24% near convergence (similar to the 32B reasoner path reported in Appendix F). Total API cost was approximately $15 on Gemini 3.1 Flash-Lite pricing.

The reasoner's temperature was set to 0.6 (matching the open-source reasoner setup) and we used a fixed system prompt matching the open-source reasoner protocol (Appendix R). Throughput was capped by a self-imposed parallel-worker limit of 16, held conservatively to avoid provider throttling; with this cap, the run took roughly 9 hours on the same $8\times$ NVIDIA RTX 6000 Ada hardware used for the open-source reasoner training. Since throughput is reasoner-bound, the wall-clock overhead of the clarification path is similar to its share of API calls.

The per-benchmark improvement pattern (largest gains on DynaMath, MathVerse, LogicVista; smallest on MathVision) is the same one observed when training against DeepSeek-R1-Qwen-32B; the average gain is slightly larger ($+5.2$ vs. $+4.4$), and WeMath flips from a small regression to a modest gain. This is consistent with the central claim of the paper: AC-RL teaches the captioner to anticipate what the reasoner needs, regardless of which specific reasoner is on the other side of the interface.

N Additional Benchmarks

All results in this appendix use the two-stage system with the DeepSeek-R1-Qwen-32B reasoner and single-pass evaluation. Significance tests are in Table 11.

Table 15 reports the two benchmarks of Section 4.1, evaluated with the AC-RL checkpoint from Appendix E. WorldMedQA-V [31] contains medical licensing exam questions, each paired with a clinical image. We use its full English set of 568 questions, which neither captioner saw during training or development. For TableVQA-Bench [21], we evaluate a uniform random sample of 500 of its 1,500 questions, drawn across all four table domains, and score it with the official evaluators.

Table 15: Two-stage accuracy on WorldMedQA-V and TableVQA-Bench.

Benchmark	n	**Pretrained**	**AC-RL**
WorldMedQA-V	568	50.2	**58.3**
TableVQA-Bench	500	31.8	**42.4**

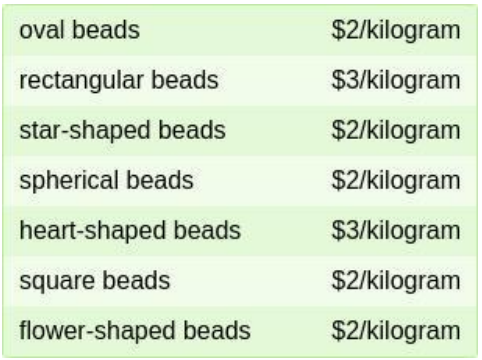
Table 16 reports ScienceQA [28], the non-mathematical subjects of MMMU and the non-numerical skills of LogicVista, evaluated with the checkpoints of Table 1, which allows a comparison with Binary-RL. The MMMU subset covers Art & Design, Business, Health & Medicine and Humanities on the dev and validation splits. The LogicVista subset covers its deductive, inductive, spatial and mechanical problems. AC-RL improves over the pretrained captioner on every subset, while Binary-RL lowers accuracy on the MMMU subjects. The subsets are small, and among the gains of AC-RL over the pretrained captioner only the one on deductive LogicVista problems has a 95% confidence interval that excludes zero.

Table 16: Two-stage accuracy on ScienceQA, the non-mathematical subjects of MMMU and the non-numerical skills of LogicVista (checkpoints of Table 1).

Subset	n	**Pretrained**	**Binary-RL**	**AC-RL**
ScienceQA (test)	2017	85.0	—	**86.2**
MMMU, non-math subjects	630	63.0	61.0	**64.8**
Health & Medicine	175	56.6	53.7	**62.3**
LogicVista, non-numerical skills	352	40.3	41.8	**45.7**
Mechanical	74	47.3	54.1	**59.5**
Deductive	93	72.0	71.0	**82.8**

O Qualitative Caption Comparisons

To illustrate how AC-RL modifies captioning behavior, we present a representative example comparing captions from the pretrained baseline and AC-RL on a table-reading problem (Figure 6).



Question: Chase wants to buy 4 kilograms of oval beads and 5 kilograms of star-shaped beads. How much will he spend? (Unit: $)

Baseline Caption

The image appears to be a table listing various types of beads along with their respective prices per kilogram. The table has two columns and lists different types of beads available for purchase. I can see oval beads, rectangular beads, star-shaped beads, spherical beads, heart-shaped beads, square beads, and flower-shaped beads. Each type has a corresponding price per kilogram shown in the second column. The table appears to be from a craft supply store or similar retailer.

AC-RL Caption

The image appears to be a table listing various types of beads along with their respective prices per kilogram. The table is as follows:

Beads	Price/kg
oval beads	$2
rectangular beads	$3
star-shaped beads	$2
spherical beads	$2
heart-shaped beads	$3
square beads	$2
flower-shaped beads	$2

(Model outputs Markdown; rendered for readability.)

Figure 6: Qualitative comparison on a table-reading problem. The baseline describes the table structure but omits numerical values. AC-RL extracts all prices needed to compute the answer.

The baseline caption correctly identifies the table structure and all bead types, but refers to prices abstractly without extracting the actual values. AC-RL, having learned from clarification requests during training that numerical details are essential, extracts every price explicitly.

P Caption Content Analysis

This appendix details the caption analysis summarized in Section 4.3. We compare the pretrained Qwen2.5-VL-3B captioner and the AC-RL captioner from Appendix E on MathVerse-VO (788 captions per model) and DynaMath (5,010 captions per model, ten variants for each of the 501 seed questions). We release the labeling rubric and the labeled test sets with the code.

P.1 Sentence Labels

Each caption is split into sentences, and the frozen 32B reasoner labels each sentence under a fixed rubric. A sentence is answer-like if it states a final answer, performs a solution step (a calculation or a deduction toward the answer), or contains a directive addressed to the reasoner. Transcribing a value that is visible in the image counts as description, also when that value is the answer. We validate the labeler on a constructed test set where answer-like sentences are inserted into clean captions. It flags 114 of the 116 inserted sentences (recall 0.98), 1 of 118 clean sentences, and none of 104 clean sentences that transcribe visible numbers. We also labeled a random sample of 100 captions ourselves, with model identity and labeler output hidden. On the 96 captions we could judge with confidence, our judgment of whether a caption contains any answer-like sentence agrees with the

labeler on 93.8% of captions (Cohen's $\kappa = 0.87$). Five MathVerse-VO captions per model were generated after the labeling run, so the labeled MathVerse-VO set has 783 captions per model.

P.2 Numeric Claims

Qwen2.5-VL-72B [2] checks each numeric claim in a caption against the image. It judges whether the value is visible in the image, derived from visible values, or not present, and whether a visible value is transcribed correctly. It also checks a stratified sample of 100 captions per model for unsupported non-numeric claims. On 154 hand-labeled claims, the verifier separates visible from derived values in all 123 applicable cases and judges 70 of 72 visible values correctly. We run this check on MathVerse-VO only.

Table 17: Caption content of the pretrained and the AC-RL captioner. Rates are percentages of captions unless noted. Gold answers are matched by string, which gives a high base rate on DynaMath, where many answers are short numbers.

	MathVerse-VO		**DynaMath**	
	Pretrained	**AC-RL**	**Pretrained**	**AC-RL**
States a final answer	47.6	18.0	58.1	28.7
Contains a solution step	76.3	35.0	89.9	54.7
Contains a directive	18.4	18.9	28.3	15.9
Any answer-like sentence	81.5	48.0	95.8	63.3
No answer-like sentence	18.5	52.0	4.2	36.7
Contains the gold answer	9.6	3.3	28.9	22.7
Numeric claims that are derived (% of claims)	19.2	8.5	—	—
Contains a derived value	39.6	15.1	—	—
Visible values transcribed correctly (% of values)	60.4	68.6	—	—
Unsupported non-numeric claims (% of claims)	17.9	16.8	—	—
Mean length (characters)	1410	850	867	765

Table 17 shows the results. On both benchmarks, AC-RL lowers the rate of final answers and solution steps. On MathVerse-VO, its captions also contain fewer derived values and transcribe visible values more accurately, with no increase in unsupported claims. A tiered reward could in principle also reward captions that discourage the reasoner from asking for clarification, for example by stating that the description is complete. Such sentences are directives, and their rate does not increase on MathVerse-VO and decreases on DynaMath.

P.3 Filtering Answer-Like Content

We remove all answer-like sentences from each caption and run the frozen reasoner again. As a control, we remove the same number of randomly chosen sentences from each caption. Table 18 shows the accuracies on MathVerse-VO. Removing any content lowers accuracy, and for both models part of the loss comes from removing content in general. Removing answer-like sentences costs the pretrained captioner 10.4 points and AC-RL 4.7 points. With answer-like content removed from both, AC-RL leads by 6.4 points, and on the 100 problems where neither captioner produces an answer-like sentence it leads by 9.0 points. If the gains of AC-RL were carried by answer-like content, these gaps would shrink.

P.4 Examples

This subsection shows three captions from MathVerse-VO, reproduced verbatim with Markdown rendered for readability. In MathVerse-VO the question is part of the image, and we add it above each caption.

The first AC-RL caption describes the square and the marked angles. Its closing sentence, which states that the description is sufficient, is the only sentence the labeler marks as answer-like (a directive). The verifier finds one mistranscribed value, the side length of the square. The reasoner answers this problem incorrectly.

Table 18: MathVerse-VO accuracy after removing caption sentences, on the problems judged for both models, with paired differences (AC-RL minus pretrained). The last row is restricted to problems where neither captioner produces an answer-like sentence.

Captions	n	**Pretrained**	**AC-RL**	Δ	**95% CI**
Unfiltered	772	31.2	31.9	$+0.65$	$[-2.98, +4.27]$
Answer-like sentences removed	766	20.8	27.2	$+6.40$	$[+2.74, +10.05]$
Random sentences removed	772	24.2	29.4	$+5.18$	$[+1.30, +8.94]$
No answer-like sentence in either	100	24.0	33.0	$+9.00$	$[-1.00, +19.00]$

AC-RL caption

Question: Find m$\angle 2$. Choices: A: 39, B: 42, C: 48, D: 51.

The image is a geometric diagram featuring a square $ABCD$ with additional points E and F forming right triangles within the square. Here's a detailed description:

1. **Square** $ABCD$: The square has vertices labeled A, B, C, and D in clockwise order. Each side of the square is marked with a length of 3 units (as indicated by the number 3).
2. **Right Triangles**: Triangle AEF is formed by extending line segment EF from the square. Triangle BCF is formed by extending line segment BF from the square.
3. **Angles**: $\angle EAF = 90^\circ$ (right angle). $\angle BCF = 48^\circ$ (given). $\angle BDC = 39^\circ$ (given).
4. **Lines and Points**: Line segments EF and BF extend from the square to form the triangles. Point F lies on the extension of EF and BF.
5. **Text and Labels**: The problem asks to find the measure of $\angle 2$. The choices provided are: A: 39, B: 42, C: 48, D: 51.
6. **Visual Elements**: The diagram includes labeled angles and sides of the square. The angles are marked as 48° at point B and 39° at point D.

This description provides all the necessary details to analyze the image and answer the question about finding the measure of $\angle 2$.

The second AC-RL caption has the most final-answer and directive sentences among the AC-RL captions on MathVerse-VO. It reads the marked angle as 30°, finds that no option matches, and revises its reasoning until it states option D, which is the gold answer. Captions with at least two final-answer or directive sentences are 2.6 times more frequent for the pretrained captioner (138 against 53 of 783 captions).

AC-RL caption

Question: As shown in the figure, angle 1 = angle 2, then angle 3 is equal to (). Choices: A: 120°, B: 130°, C: 145°, D: 150°.

The image shows a geometric figure with two parallel lines a and b. Line c intersects line a at point C and line d intersects line b at point D. The angles formed by these intersections are labeled as follows:

- Angle 1 is labeled as $\angle 1$.
- Angle 2 is labeled as $\angle 2$.
- Angle 3 is labeled as $\angle 3$.
- Angle 4 is labeled as $\angle 4$.

Angle 2 is given as 30°. The problem states that $\angle 1 = \angle 2$, which means $\angle 1 = 30^\circ$.

To find $\angle 3$, we need to use the properties of parallel lines and corresponding angles. Since a and b are parallel lines, and c and d are transversals, the corresponding angles formed by the intersection of c with a and d with b will be equal.

Given that $\angle 1 = 30^\circ$, and knowing that $\angle 1$ and $\angle 3$ are corresponding angles (formed by the intersection of c with a and d with b), we can conclude that $\angle 3 = \angle 1 = 30^\circ$.

However, none of the provided choices match 30°. This suggests there might be an error in the problem statement or the choices provided. Given the choices, let's re-evaluate the problem with the assumption that the angle given is not necessarily $\angle 2$ but rather another angle related to the figure.

If we consider the supplementary angles formed by the transversal c intersecting the parallel lines a and b, we can deduce the following:

- $\angle 1 + \angle 2 = 180^\circ$ because they are supplementary angles.
- Therefore, $\angle 1 = 180^\circ - \angle 2 = 180^\circ - 30^\circ = 150^\circ$.

Since $\angle 1 = \angle 2$, and $\angle 1 = 150^\circ$, then $\angle 3$ must also be 150° because it is the same angle formed by the intersection of c with a and d with b.

Thus, the correct choice is: D: 150°

In the last example, both captioners describe a translated circle as centered at the origin, and the reasoner answers incorrectly with either caption. Both captions also work through the answer options.

AC-RL caption (abridged)

Question: Describe the translation to get from $x^2 + y^2 = 4^2$ to the graph shown. Choices: A: The graph has been translated 4 units downwards. B: The graph has been translated 4 units right. C: The graph has been translated 4 units left. D: The graph has been translated 4 units upwards.

The image shows a graph with a circle centered at the origin (0, 0) on a coordinate plane. The equation of the circle is $x^2 + y^2 = 4$, which represents a circle with a radius of 2 units. The circle is centered at the origin, meaning its center is at the point (0, 0).
The choices provided are: A: The graph has been translated 4 units downwards. B: The graph has been translated 4 units right. C: The graph has been translated 4 units left. D: The graph has been translated 4 units upwards.
[...] Therefore, the correct translation is 4 units upwards. The correct answer is: D: The graph has been translated 4 units upwards.

Pretrained caption for the same problem (abridged)

The image shows a graph with a circle centered at the origin (0, 0) on a coordinate plane. The circle has a radius of 4 units, as indicated by the distance from the center to the edge of the circle along both the x-axis and y-axis. The circle is drawn using a solid black line.
The x-axis ranges from -10 to 10, and the y-axis ranges from -10 to 10. The grid lines are evenly spaced, with each unit marked clearly on both axes.
[...] Therefore, the correct answer is: None of the given options accurately describe the translation of the circle from $x^2 + y^2 = 4^2$ to the graph shown.

Q Multi-Round Clarification

To test whether clarification saturates, we let the reasoner request up to R rounds of clarification at evaluation time. In each round, the accumulated dialogue (the initial description and all previous requests and responses) re-enters the decision stage of Appendix R, and the reasoner either answers or asks again until the budget is used. The frozen Qwen2.5-VL-3B answers the requests, as in training. $R = 0$ is single-pass evaluation, and $R = 1$ is the protocol of Table 3. We evaluate the AC-RL captioner from Appendix E on MathVerse-VO and on MathVision-500, a uniform random sample of 500 MathVision problems, and the pretrained captioner on MathVision-500. These accuracies come from separate runs of the clarification-enabled pipeline, so the single-pass values differ slightly from those elsewhere in the paper. For $R \geq 2$, the decision prompt differs from the one in Appendix R in a single instruction, which reads "*You may request one specific piece of additional visual information per round. You have {r_remaining} clarification request(s) remaining for this problem. Make each request very specific and focused; do not repeat information already provided.*" Here {r_remaining} starts at R and decreases after each response. $R = 1$ uses the prompt of Appendix R unchanged.

Table 19: Accuracy under a budget of R clarification rounds. "Asked" = fraction of problems with at least one request. Δ = paired accuracy difference to $R = 0$, with 95% bootstrap confidence intervals.

	MathVerse-VO, AC-RL			MathVision-500, AC-RL			MathVision-500, pretrained		
R	Acc.	Asked	Δ	Acc.	Asked	Δ	Acc.	Asked	Δ
0	33.0	—	—	44.4	—	—	36.8	—	—
1	31.3	48.0	-1.80 $[-4.51, +0.90]$	45.7	39.7	$+1.63$ $[-2.04, +5.10]$	39.5	36.6	$+2.61$ $[-0.80, +6.01]$
2	33.6	50.1	$+0.51$ $[-2.06, +3.09]$	44.5	39.2	$+0.82$ $[-3.28, +4.71]$	41.9	38.5	$+5.21$ $[+1.80, +8.62]$
3	32.6	47.2	-0.65 $[-3.25, +1.95]$	43.4	39.8	-0.61 $[-4.08, +2.86]$	40.6	37.6	$+3.61$ $[+0.40, +6.83]$

For the AC-RL captioner, accuracy stays flat across budgets on both benchmarks, and no budget differs significantly from single-pass evaluation, although 39–50% of problems request clarification. The pretrained captioner gains from additional rounds, by up to 5.2 points at $R = 2$, and its gains at $R = 2$ and $R = 3$ are significant. Among the problems where clarification is requested at $R = 3$ on MathVision-500, 23 pretrained answers change from wrong to right and 7 from right to wrong,

against 14 and 14 for AC-RL. Clarification at evaluation time therefore helps when captions miss information, and AC-RL training removes most of this headroom by front-loading the information into the initial caption.

R Prompt Templates

We present the complete set of prompts used in our AC-RL framework. The prompts are structured to maintain clear role separation between the captioner (visual description) and reasoner (problem-solving), while enabling controlled interaction during training.

R.1 Vision-Language Model Prompts

Initial Caption Generation. The following prompt instructs the VLM to generate comprehensive visual descriptions without solving the problem:

vlm_initial_description_prompt

```
I need your help analyzing this image to prepare for answering the following
question:
{question}
IMPORTANT: DO NOT answer the question directly.  Instead, provide a
comprehensive and detailed description of everything visible in the image
that could be relevant for answering this question.
Focus on describing:
      • All objects, people, text, and visual elements in the image
      • Spatial relationships between different elements
      • Any text content that is visible, transcribed exactly
      • Colors, shapes, patterns, and visual attributes
      • Relevant contextual details and background information
Your description should be detailed enough that someone could mentally
reconstruct the image without seeing it, but DO NOT provide step-by-step
instructions on how to recreate it.
```

Clarification Response. When the reasoner requests specific visual information, the frozen reference model uses this prompt:

vlm_focused_description_prompt

```
Original Question:  {question}
Previous Description:  {previous_descriptions}
CONTEXT: The description above was provided for this image, but some details
might be missing or unclear.  We are asking this specific follow-up question
to gather additional visual details.
Your specific task:  {focus_request}
CRITICAL INSTRUCTIONS:
      • You are a VISUAL DESCRIBER only - DO NOT attempt to answer the
        original question
      • DO NOT solve the problem or provide calculations
      • DO NOT give step-by-step solutions or reasoning
      • ONLY describe what you can see in the image that relates to the
        specific request
      • Focus solely on visual elements:  objects, text, numbers, shapes,
        spatial relationships
      • If asked about measurements, describe what you see but don't
        calculate or solve
      • If asked about equations, transcribe what's visible but don't solve
        them
      • Be thorough and precise in your description since this is to clarify
        specific missing details
```

R.2 Reasoner Prompts

Adaptive Decision Mechanism. The reasoner evaluates whether the initial caption is sufficient or requires clarification:

reasoner_adaptive_decision_prompt

```
You are an expert visual reasoning assistant. Your task is to analyze the
given image description and decide if you can solve the problem directly or
if you need one specific piece of additional visual information.
Image Description: {description}
Question: {question}
ANALYSIS INSTRUCTIONS:
    1. CAREFUL EVALUATION: Analyze if the description contains all specific
       visual details needed to solve completely and accurately.
    2. BE CONSERVATIVE: If missing ANY crucial visual detail, request MORE
       information rather than guess.
    3. ONE CLARIFICATION ONLY: You can request specific additional visual
       information if needed.
    4. DECISION CRITERIA:
        • If you have ALL visual details needed: Status = SOLVED
        • If missing crucial visual information: Status = NEED_MORE_INFO
    5. AVOID ASSUMPTIONS: Don't guess numbers, assume "typical" values, or
       fill in missing details.
CRITICAL PRINCIPLES:
    • BE SPECIFIC in requests: Ask for exact details you need
    • SOLVE CONFIDENTLY when possible: If you have enough information,
      provide the complete solution
    • REQUEST STRATEGICALLY: Make your one request count - ask for the most
      crucial missing details
OUTPUT FORMAT (all fields required):
Reasoning: [Your detailed analysis of what information you have and what
might be missing]
Status: [SOLVED or NEED_MORE_INFO]
Answer: [Your complete final answer if Status is SOLVED - use \boxed{answer}
format, otherwise N/A]
Request: [Your specific request for additional visual information if Status
is NEED_MORE_INFO, otherwise N/A]
```

Final Answer Generation. For both direct solving and post-clarification scenarios:

reasoner_final_prompt

```
You are an expert mathematical reasoning assistant. Based on the complete
image description below, please solve the mathematical problem step-by-step.
Complete Image Description: {description}
Question: {question}
INSTRUCTIONS:
    1. Analyze the complete image description carefully
    2. Work through the problem step-by-step with clear mathematical
       reasoning
    3. Show all calculations and logical steps
    4. Provide your final answer in the required format
    5. Use \boxed{answer} notation. For multiple choice, use \boxed{letter}
       format
You MUST follow this format:
<think>
Your detailed reasoning and thought process here...
</think>
<answer> Final Answer: your final answer here </answer>
```

S Training Hyperparameters

We provide complete hyperparameter specifications to ensure reproducibility. All experiments use the same random seed for dataset sampling to enable fair comparisons.

Table 20: Hyperparameter settings for AC-RL training across different model sizes.

Hyperparameter	**2B Models**	**3B Models**
Optimization		
Learning rate	3×10^{-6}	2×10^{-6}
Effective batch size	256	256
KL divergence weight (β)	0.001	0.001
LoRA Configuration		
LoRA rank (r)	128	256
LoRA alpha (α)	256	512
LoRA dropout	0.05	0.05
BNPO Settings		
Group size (M)	8	8
Number of iterations	6	6
Remaining parameters	TRL library defaults	
Generation Parameters		
Captioner temperature	1.0	
Captioner max tokens	800	
Reasoner temperature	0.6	
Reasoner top-p	0.95	
Reasoner max tokens	100,000	
Reward Configuration		
Clarification penalty ($1 - \alpha$)	0.3	

Implementation Details. We implement AC-RL using the Transformers Reinforcement Learning (TRL) library [39] with Beta-Normalization Policy Optimization (BNPO). The LoRA [18] adapters

are applied to all linear layers in the vision-language models. Training typically converges within 1,000 steps on the ViRL-39K dataset. All experiments use mixed precision training (fp16). The AC-RL checkpoint from the sensitivity analysis (Appendix E), which we also use for the cross-reasoner, additional-benchmark, caption and multi-round analyses, is trained with an effective batch size of 512 and evaluated after 200 steps. The clarifier controls (Appendix K) use an effective batch size of 256.

Computational Requirements. Training a 3B parameter captioner with AC-RL requires approximately 50 hours on 8 NVIDIA RTX 6000 Ada GPUs. The 2B models require 40 hours on the same hardware configuration. The reasoner is hosted on a $4\times$ AMD MI300A node, though it is never fully saturated during training. The full research project required additional compute beyond the runs reported in the paper, primarily for development iterations during implementation and a brief hyperparameter search over learning rate and clarification penalty α; we estimate this additional compute at roughly $3\times$ the cost of the reported experiments.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims in the abstract and the introduction are fully supported by the experimental setup and results.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 4.1 and Appendix H discuss the WeMath regression and its cause; the Conclusion notes that extension to non-mathematical domains is left to future work.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Theorem B.1 and its proof appear in Appendix B, with assumptions stated in the theorem.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Algorithm 1 specifies the training procedure, Appendix S provides hyperparameters, Appendix R provides exact prompts, and Appendix C lists evaluation benchmarks.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Code is provided as supplementary material. All datasets used are publicly available (Appendix C).

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4 describes training configurations, evaluation protocol, and benchmarks; Appendix S provides full hyperparameters.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Paired bootstrap confidence intervals for all main-text comparisons are reported in Appendix G. We do not report variance over training seeds due to compute cost.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Compute requirements, including additional compute used for development and hyperparameter search beyond the reported experiments, are reported in Appendix S.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: The work studies interface learning between vision and reasoning modules; it has no direct path to deployment-specific societal impacts beyond those of general vision-language modeling.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: We do not release new pretrained models or scraped datasets that pose misuse risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All models and datasets are cited in the main text and Appendix C, with original sources credited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We do not release new datasets or pretrained models with this submission.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: No crowdsourcing or human subjects research.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human subjects research.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs were used only for editing and formatting assistance.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.