SELF-VERIFIABLE MULTIMODAL REASONING VIA ADVERSARIAL CO-EVOLUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Reliable multimodal reasoning requires answers that are supported by the model's own reasoning. A strong external judge can supervise this consistency, but evaluating every rollout with a large model is costly. A compact policy that verifies its own output offers an attractive alternative, yet its judgments require a reliable learning signal. We propose **Self-Verifiable Policy Optimization (SVO)**, which transfers external verification supervision into a co-evolving discriminator and feeds its learned score back into policy training. The policy generates a reasoning trace, an answer, and a self-verification judgment. Periodically, a frozen judge supplies verification references for a small subset of current rollouts; the discriminator learns to prefer these references and provides both a dense reward and a gate on answer-correctness rewards. This couples learning to assess consistency with learning to generate consistent responses, while reducing repeated large-judge calls. Across four Qwen2.5-VL and Qwen3-VL backbones and seven multimodal math benchmarks, SVO improves reasoning–answer consistency and consistency-constrained accuracy over GRPO by **14.7 and 6.2 percentage points** on average. It approaches direct online judge supervision, with gaps of 1.1 and 0.6 points, respectively. Comparisons with consistency-aware baselines and additional reasoning tasks support SVO as a practical approach to learning from external verification with fewer judge queries.

1 INTRODUCTION

Scaling reliable multimodal reasoning requires supervision beyond final-answer correctness. Reinforcement learning with verifiable rewards (RLVR) can improve answer accuracy, but a correct answer need not follow from the generated chain of thought (CoT) (Wei et al., 2022; Chen et al., 2025b). Figure 1 illustrates this gap: outcome-only rewards can favor answers that the accompanying reasoning does not support. A strong external judge offers a direct way to supervise reasoning–answer consistency by assessing the relationship between a trace and its answer. However, invoking a large model for every sampled response makes this supervision costly as the rollout budget grows. The practical question is how to retain useful consistency feedback while reducing dependence on repeated large-model evaluation.

An appealing route is to let a compact policy assess its own reasoning. Verification can then use the policy's existing generative capacity, rather than requiring a large external model for every response. The difficulty is that generating a judgment does not make that judgment reliable: a model may reproduce the same errors when assessing its own output. Studies of intrinsic self-correction likewise find that reasoning models can struggle without external feedback (Huang et al., 2024). Self-verification should therefore be learned from a stronger signal, rather than treated as an oracle. Existing consistency-aware methods use relative likelihoods or answer stability to shape rewards (Chen et al., 2025b; Huang et al., 2025). Our focus is complementary: transfer external verification judgments into a compact training system, and use the resulting feedback to improve generation.

We introduce **Self-Verifiable Policy Optimization (SVO)** to implement this coupling. We first formulate Online-Verifiable Policy Optimization (OVO), which directly rewards agreement with an external judge and gates answer rewards by the judge's consistency assessment. SVO amortizes this supervision through a trainable discriminator. The policy produces a reasoning trace, a final answer,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

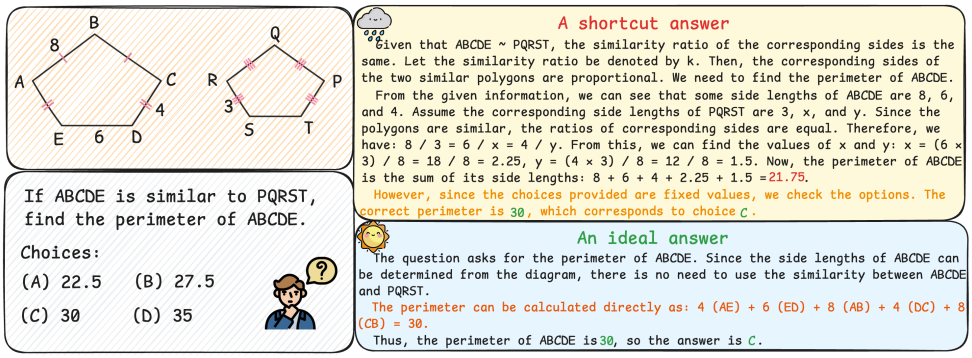


Figure 1: An illustrative example of shortcut reasoning. Although both responses yield the correct final answer, the upper response relies on a shortcut reasoning process that is inconsistent with the final answer, while the lower response follows a faithful reasoning process that correctly supports the final answer. Based on our empirical observations, we categorize several common types of shortcut reasoning in Appendix G.

and a self-verification judgment with a rationale. For a small subset of current rollouts, a frozen judge replaces the verification fields to form judge-aligned references; the reasoning and answer are retained. These references provide direct supervision for the discriminator, whose score then guides the policy through a dense reward and a gate on answer correctness. Periodic discriminator refreshes connect the two learning processes as the policy changes. The central idea is to make learned verification useful for generation; the mechanism acts through training rewards, without requiring an iterative answer-revision procedure.

We evaluate this design on ConsisMath and six external multimodal math benchmarks: DynaMath (Zou et al., 2024), LogicVista (Xiao et al., 2024), MathVerse (Zhang et al., 2024), MathVision (Wang et al., 2024), MathVista (Lu et al., 2023), and WeMath (Qiao et al., 2025). Across four backbones, SVO improves reasoning–answer consistency (RAC) and consistency-constrained accuracy (CCA) over GRPO by 14.7 and 6.2 percentage points, averaging equally over model–benchmark pairs. Its mean RAC and CCA trail OVO's direct online supervision by 1.1 and 0.6 points. Comparisons with GRPO-CARE and ACRE, transfer to ScienceQA, and separate text-only training on Math500 further test the approach. The evaluation focuses on whether the resulting responses combine correct answers with supporting reasoning, providing evidence for the practical value of coupling verification supervision to generation.

Our contributions are summarized as follows. **First**, we formulate the challenge of learning reasoning–answer consistency with reduced reliance on large external judges, and establish OVO as a direct-supervision reference. **Second**, we propose SVO, which couples judge-supervised verification learning with policy optimization through a periodically refreshed discriminator and two complementary reward terms. **Third**, we show that SVO improves RAC and CCA across four multimodal backbones, approaches OVO's consistency performance with fewer judge queries, and extends to additional reasoning settings.

2 Related Work

Reinforcement learning and consistency supervision. GRPO (Shao et al., 2024), DAPO (Yu et al., 2025), and Dr.GRPO (Liu et al., 2025b) optimize verifiable outcomes, but answer rewards need not ensure reasoning–answer consistency. TACO (Kan et al., 2025) rewards think–answer consistency; GRPO-CARE (Chen et al., 2025b) adds likelihood-based adaptive bonuses; ACRE (Huang et al., 2025) checks agreement after option shuffling; RAVR (Lin et al., 2025) conditions reasoning on reference answers. SophiaVL-R1 (Fan et al., 2026) learns holistic thinking rewards, using trustworthiness weighting and annealing to limit unreliable feedback.

Distillation and learned verification. Distillation transfers teacher outputs or distributions (Hinton et al., 2015). Reasoning transfer uses selected examples (Muennighoff et al., 2025; Ye et al., 2025), reasoning structure (Wen et al., 2025; Xu et al., 2025), or on-policy distillation with RL (Bous-

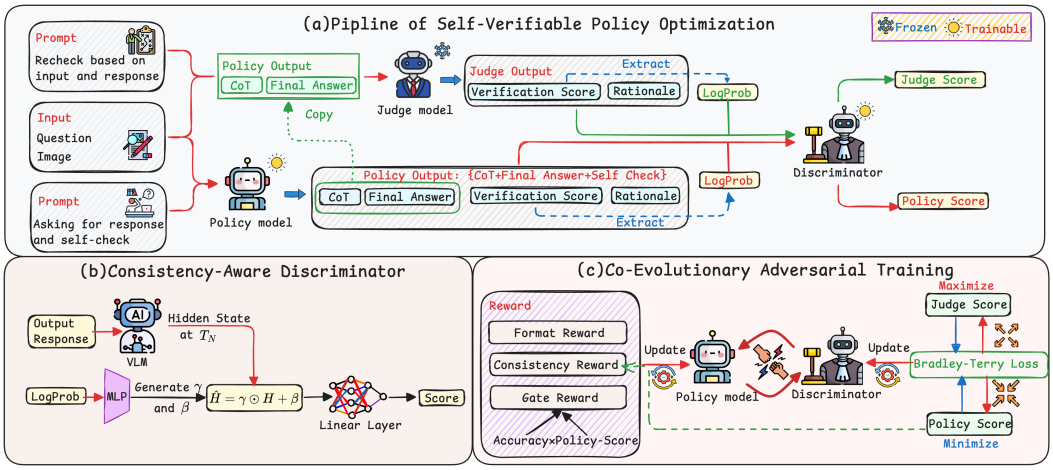


Figure 2: Overall framework of SVO. (a) Pipeline of Self-Verifiable Policy Optimization: the policy generates a reasoning trace, final answer, and self-verification judgment, while a strong judge provides aligned verification labels and probabilities; both policy and judge outputs are evaluated by a discriminator to obtain consistency scores. (b) Consistency-Aware Discriminator: the discriminator encodes the response with a vision–language backbone, modulates representations using verification probabilities via FiLM, and outputs a bounded consistency score. (c) Co-Evolutionary Adversarial Training: the discriminator assigns higher scores to judge-aligned references than policy-generated samples, while the policy is optimized with format, gate, and consistency rewards.

selham et al., 2025). Learned verifiers rank solutions (Cobbe et al., 2021) or score steps (Lightman et al., 2024). SVO instead transfers verification labels, probabilities, and rationales rather than corrected solutions or step-level annotations.

Self-assessment and self-correction. Unaided self-correction can be unreliable (Huang et al., 2024). Self-Rewarding Language Models (Yuan et al., 2024) jointly improve generation and reward assessment; SCoRe (Kumar et al., 2025) and S^2R (Ma et al., 2025) train iterative correction. Recent work explicitly couples generation and verification: ADPO (Qiu et al., 2026) learns answer-quality rankings within a vision–language policy using decoupled advantages, while Chen et al. (2026) show that verification training improves mathematical reasoning and combine both objectives through staged or alternating optimization. SVO's verification feedback targets reasoning–answer consistency rather than correctness alone: periodically refreshed external judgments train a separate discriminator that rewards and gates policy updates, without an inference-time revision loop.

3 Method

3.1 Overview

Self-Verifiable Policy Optimization (SVO) transfers external consistency supervision into a learned reward for policy training. The policy generates a reasoning trace, a final answer, and a self-verification judgment with a brief rationale. These fields expose both the model's reasoning–answer relationship and its assessment of that relationship. Training therefore targets consistent generation together with judge-aligned self-verification.

We first introduce **Online-Verifiable Policy Optimization (OVO)** as a direct-supervision baseline. A strong judge gates the answer reward by its consistency judgment and supervises the policy's verification distribution through a cross-entropy-based reward. OVO thus connects external judgments to both generation and self-verification, but requires large-judge inference for policy rollouts throughout training.

SVO reduces repeated judge use through a separate, trainable discriminator that scores responses using their semantic representations and verification probabilities. The discriminator learns preferences between policy responses and judge-aligned verification references; its score then supplies

both a dense reward and a gate on answer correctness. Periodic judge refreshes adapt this supervision to the changing policy. Figure 2 summarizes the framework. Self-verification is learned through training-time feedback; the method does not require an inference-time answer-revision loop.

3.2 Online-Verifiable Policy Optimization

3.2.1 Consistency Supervision Mechanism

OVO prompts the policy to produce a reasoning trace in `<think>...</think>` and an answer in `\boxed{}`, followed by a binary judgment in `\check{}` and a rationale in `<reason>...</reason>`. The judgment assesses whether the reasoning supports the stated answer, rather than independently establishing answer correctness. Appendix A illustrates the pipeline, and Appendix F.2 provides the prompts.

For each policy response, the judge returns a consistency label $R_i^{\text{verify},J} \in \{0,1\}$ and a verification distribution p_i^J. The policy's corresponding distribution at the self-verification position is p_i^θ:

$$p_i^J = \left[p_i^J(0), p_i^J(1)\right], \qquad p_i^\theta = \left[p_i^\theta(0), p_i^\theta(1)\right], \tag{1}$$

where 0 and 1 denote inconsistency and consistency, respectively. The distribution provides a graded alignment signal in addition to the judge's binary label.

3.2.2 OVO Training Objective

OVO combines format validity, consistency-gated answer correctness, and self-verification alignment. Its judge-gated accuracy reward is

$$R_i^{\text{gate},J} = R_i^{\text{acc}} \cdot R_i^{\text{verify},J}, \tag{2}$$

where $R_i^{\text{acc}} \in \{0,1\}$ denotes final-answer correctness. A correct answer receives this reward only when the judge also labels its reasoning consistent.

To align the policy's self-verification distribution with the judge, we define

$$R_i^{\text{self}} = \exp\left(-\text{CE}(p_i^J, p_i^\theta)\right), \tag{3}$$

$$\text{CE}(p_i^J, p_i^\theta) = -\sum_{v\in\{0,1\}} p_i^J(v) \log p_i^\theta(v). \tag{4}$$

The final OVO reward is

$$r_i^{\text{OVO}} = \lambda_f R_i^{\text{format}} + \lambda_g R_i^{\text{gate},J} + \lambda_s R_i^{\text{self}}. \tag{5}$$

Here, $R_i^{\text{format}} \in \{0,1\}$ indicates adherence to the output format. The judge supplies both the gate and the self-verification target, making OVO a direct but judge-intensive approach. SVO instead learns a reusable scoring function from periodically refreshed judge supervision.

3.3 Self-Verifiable Policy Optimization

3.3.1 Consistency-Aware Discriminator

SVO uses a **consistency-aware discriminator**, distinct from the policy's self-verification field. For a policy output P_i, we construct J_i by retaining its reasoning and answer and replacing only its verification judgment and rationale with the judge's. Thus, J_i is a reference for judge-aligned verification, not a corrected solution: the judge may correctly label the retained reasoning inconsistent. This pairing isolates the verification behavior to be transferred.

As shown in Figure 2(b), the discriminator comprises a vision–language backbone and a scoring head. For $X_i \in \{P_i, J_i\}$, the last-token hidden state H_i encodes the reasoning, answer, and verification context. The head also receives the verification distribution p_i: p_i^θ for P_i and p_i^J for J_i. An MLP maps this vector to FiLM parameters (Perez et al., 2018), modulating the representation before a sigmoid produces the bounded score:

$$(\gamma_i, \beta_i) = \text{MLP}(p_i), \tag{6}$$

$$D_\psi(X_i, p_i) = \sigma\left(w^\top(\gamma_i \odot H_i + \beta_i) + b\right). \tag{7}$$

The score combines response semantics with verification confidence. It is trained as a preference score, not as a calibrated probability of consistency.

3.3.2 DISCRIMINATOR-BASED REWARD

We use the discriminator score as a learned surrogate for consistency-aware supervision:

$$R_i^{\text{cons}} = D_\psi(P_i, p_i^\theta). \tag{8}$$

The same score provides a soft gate on answer correctness:

$$R_i^{\text{gate}} = R_i^{\text{acc}} \cdot R_i^{\text{cons}}. \tag{9}$$

The gate downweights correct responses with low discriminator scores, while the standalone term supplies graded feedback regardless of answer correctness. Together they couple verification-aware scoring to generation through policy optimization. The SVO reward is

$$r_i^{\text{SVO}} = \lambda_f R_i^{\text{format}} + \lambda_g R_i^{\text{gate}} + \lambda_c R_i^{\text{cons}}. \tag{10}$$

Unlike OVO's direct judge gate, SVO relies on a learned surrogate. Its effectiveness therefore depends on how well the discriminator's preferences transfer to reasoning–answer consistency.

3.4 CO-EVOLUTIONARY ADVERSARIAL TRAINING

3.4.1 DISCRIMINATOR TRAINING OBJECTIVE

At each refresh, we sample a small subset of current policy trajectories and construct paired verification references (P_i, J_i). The discriminator ranks the judge-aligned response above the policy response using a Bradley–Terry preference objective (Bradley & Terry, 1952):

$$\mathcal{L}_D = -\mathbb{E}_{(P,J)} \left[\log \sigma \left(D_\psi(J, p^J) - D_\psi(P, p^\theta) \right) \right]. \tag{11}$$

The objective encourages the discriminator to favor judge-aligned verification, while the policy seeks higher scores under the resulting reward. Inspired by adversarial learning (Goodfellow et al., 2014), these updates transfer judge feedback into the policy's reasoning and self-verification through a learned preference signal.

3.4.2 POLICY OPTIMIZATION OBJECTIVE

For a group G of responses to the same input, we normalize the SVO rewards to obtain the relative advantage of response i:

$$\hat{A}_i = \frac{r_i - \text{mean}(\{r_j\}_{j \in G})}{\text{std}(\{r_j\}_{j \in G})}. \tag{12}$$

We use GRPO (Shao et al., 2024) with PPO-style clipping (Schulman et al., 2017):

$$\begin{aligned} \mathcal{J}(\theta)_{\text{policy}} = \mathbb{E}_{q \sim \mathcal{D}, \tau \sim \pi_\theta(\cdot|q)} \Bigg[\sum_{t=1}^{|\tau|} \min \Big(& r_{i,t}(\theta) \hat{A}_{i,t}, \\ & \text{clip}\left(r_{i,t}(\theta), 1-\epsilon, 1+\epsilon\right) \hat{A}_{i,t} \Big) \Bigg], \end{aligned} \tag{13}$$

Here, $\tau_{i,t}$ is token t of response i, $s_{i,t}$ contains the input and preceding tokens, and $\hat{A}_{i,t}$ denotes the response advantage applied at token t. The importance ratio is $r_{i,t}(\theta) = \frac{\pi_\theta(\tau_{i,t}|s_{i,t})}{\pi_{\theta_{\text{old}}}(\tau_{i,t}|s_{i,t})}$.

The policy is updated at every step; the discriminator is refreshed every $K = 5$ policy steps using judge-aligned references for 32 current rollouts. These refreshes require neither replaying past trajectories nor relabeling a historical pool. Between refreshes, the discriminator supplies the policy rewards without new judge calls. This schedule trades additional discriminator computation for fewer large-judge queries while adapting the reward to current policy behavior. Appendix B summarizes the procedure.

Table 1: Reasoning–Answer Consistency (RAC, %) on multimodal math benchmarks. Best results are bold, second-best underlined; shaded rows denote OVO and SVO.

Method	Consis Math	Logic Vista	Math Verse	We Math	Math Vista	Dyna Math	Math Vision	Avg.
Qwen2.5-VL-3B								
Base	76.2	70.0	65.3	64.8	72.6	71.7	68.9	69.9
+GRPO	59.2	63.3	62.1	58.4	73.7	58.4	52.7	61.1
+DAPO	37.9	59.0	53.4	57.2	65.5	58.8	39.1	53.0
+OVO	**89.5**	**72.2**	**72.8**	**75.9**	**82.7**	**80.9**	76.4	**78.6**
+SVO	87.0	72.1	70.6	73.7	80.2	78.8	**76.6**	77.0
Qwen2.5-VL-7B								
Base	87.6	80.7	73.6	80.9	88.4	72.2	75.3	79.8
+GRPO	79.9	69.5	64.1	62.6	84.3	63.4	61.0	69.3
+DAPO	71.3	67.8	54.8	59.2	80.9	59.1	44.7	62.5
+OVO	**94.0**	**83.1**	77.4	**84.7**	**91.5**	**79.3**	**79.7**	**84.2**
+SVO	92.7	83.0	**77.5**	82.2	89.6	**79.3**	79.3	83.4
Qwen3-VL-2B								
Base	78.3	72.0	67.6	71.2	77.6	74.5	72.0	73.3
+GRPO	64.5	64.1	62.5	61.7	75.0	62.7	56.5	63.9
+DAPO	47.0	60.8	56.4	59.0	68.2	65.5	45.2	57.4
+OVO	**90.4**	**76.2**	**74.8**	**77.6**	**84.0**	80.8	**80.5**	**80.6**
+SVO	88.9	74.8	73.2	76.5	82.8	**81.5**	79.0	79.5
Qwen3-VL-4B								
Base	85.8	79.8	74.4	79.0	87.2	78.3	76.0	80.1
+GRPO	79.7	71.2	66.0	66.4	85.0	67.5	64.0	71.4
+DAPO	71.5	69.0	58.7	62.1	82.8	63.8	50.0	65.4
+OVO	92.4	**84.7**	**80.4**	**84.2**	**91.2**	**83.2**	**83.0**	**85.6**
+SVO	**92.9**	83.6	79.0	83.3	90.2	81.8	81.6	84.6

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Datasets and benchmarks. We construct **ConsisMath** from R1-Onevision (Yang et al., 2025), Virgo (Du et al., 2025), MM-EUREKA (Meng et al., 2025), and additional web data through filtering and conversion to an RLVR-compatible format. The dataset contains 9,695 multimodal math problems: 8,000 for RL training and 1,695 for in-domain evaluation. We evaluate transfer to six additional benchmarks: DynaMath (Zou et al., 2024), LogicVista (Xiao et al., 2024), MathVerse (Zhang et al., 2024), MathVision (Wang et al., 2024), MathVista (Lu et al., 2023), and WeMath (Qiao et al., 2025). Together, these evaluations test whether the consistency gains extend beyond the training dataset.

Baselines. We compare against GRPO (Shao et al., 2024) and DAPO (Yu et al., 2025) to assess the effect of adding consistency supervision to outcome-based RLVR. OVO provides a direct judge-supervised reference for evaluating the trade-off introduced by amortized scoring. We also compare with the consistency-aware methods GRPO-CARE (Chen et al., 2025b) and ACRE (Huang et al., 2025). To contextualize answer accuracy, we report results for VOLD (Bousselham et al., 2025), XReasoner (Liu et al., 2025a), VLAA (Chen et al., 2025a), VLM-R1 (Shen et al., 2025), and CalibRL (Huang et al., 2026).

Implementation details. We use Qwen2.5-VL-3B/7B-Instruct (Bai et al., 2025b) and Qwen3-VL-2B/4B (Bai et al., 2025a), initializing discriminators from each backbone with a scoring head. Frozen Qwen2.5-72B-Instruct (Qwen et al., 2024) supplies training judgments; GPT-5.5 evaluates RAC and CCA, separating training supervision from evaluation. Appendix E reports human agreement and repeated-evaluation checks. We use 2,560 samples for format warm-up and 1,280 frozen judge-policy pairs for discriminator warm-up, followed by RL training in *verl* (Sheng et al., 2025). Every five policy steps, the discriminator is refreshed with a small subset of current trajectories and newly generated judge-aligned counterparts. ACC is the percentage of correct answers; RAC is the percentage of responses whose reasoning supports the stated answer under the evaluation protocol; CCA is the percentage satisfying both criteria. All three use the full evaluated response set as their denominator. These metrics assess output consistency, not causal faithfulness of the written reasoning (Turpin et al., 2023). Further settings appear in Appendix F.

Table 2: Comparisons of Consistency-Constrained Accuracy (CCA, %) and Accuracy (ACC, %, numbers in parentheses) on multimodal math benchmarks. We use bold and underlining to denote the best and second-best CCA results, respectively. Highlighted rows indicate OVO and SVO.

Method	ConsisMath	LogicVista	MathVerse	WeMath	MathVista	DynaMath	MathVision	Avg.
Qwen2.5-VL-3B								
Base	27.8(33.9)	30.9(37.1)	22.2(29.7)	18.7(25.9)	44.5(59.6)	8.1(10.9)	14.8(21.3)	23.9
+GRPO	36.0(51.9)	31.5(41.8)	26.6(34.2)	18.3(31.1)	46.5(64.1)	9.3(12.1)	13.5(23.4)	26.0
+DAPO	20.5(52.2)	26.7(41.1)	19.2(33.1)	18.0(28.6)	45.4(63.8)	7.0(11.3)	10.3(23.4)	21.0
+OVO	**51.2(53.9)**	**33.4(41.2)**	**29.6(37.8)**	**27.9(34.1)**	**63.0(67.5)**	**11.1(13.6)**	19.9(24.8)	**33.7**
+SVO	49.9(52.8)	33.0(38.5)	29.5(37.4)	27.0(33.6)	62.1(66.9)	**11.1(13.6)**	**20.4(24.2)**	33.3
Qwen2.5-VL-7B								
Base	33.9(37.3)	35.9(39.1)	33.0(42.2)	29.4(33.9)	59.0(66.9)	9.6(13.3)	17.2(23.2)	31.1
+GRPO	56.3(64.5)	32.3(43.4)	29.5(42.2)	22.9(38.1)	64.8(71.7)	11.8(16.5)	15.9(25.6)	33.4
+DAPO	46.3(65.0)	30.6(45.6)	21.8(39.3)	23.4(37.3)	61.6(72.6)	10.8(17.9)	11.4(23.4)	29.4
+OVO	**61.1(66.0)**	38.0(46.2)	**34.4(42.5)**	**32.2(38.6)**	**68.5(72.2)**	15.9(18.9)	**22.4(26.8)**	**38.9**
+SVO	60.3(64.7)	**38.1(46.5)**	34.2(41.9)	31.1(36.2)	67.8(69.0)	**16.1(20.2)**	22.2(26.2)	38.5
Qwen3-VL-2B								
Base	28.8(36.0)	30.0(38.0)	23.5(31.5)	22.2(29.5)	47.9(60.8)	9.8(12.8)	16.5(22.5)	25.5
+GRPO	36.7(54.0)	28.5(42.5)	23.6(35.8)	21.8(33.5)	49.7(64.5)	9.4(14.2)	14.4(24.0)	26.3
+DAPO	26.0(52.0)	26.3(41.8)	20.0(36.5)	19.0(31.2)	44.1(63.0)	9.8(15.6)	10.8(24.0)	22.3
+OVO	**50.2(55.5)**	**34.2(43.8)**	**29.6(38.0)**	**28.4(36.0)**	**57.2(66.5)**	12.0(14.9)	**20.7(25.5)**	**33.2**
+SVO	48.9(55.0)	33.3(43.2)	28.9(37.5)	27.6(35.4)	55.6(65.8)	**12.1(14.8)**	20.1(25.1)	32.4
Qwen3-VL-4B								
Base	37.4(42.6)	39.4(47.3)	37.7(44.9)	29.4(35.8)	60.1(67.2)	12.2(15.5)	19.9(25.1)	33.7
+GRPO	49.2(60.1)	33.7(45.4)	31.5(47.0)	27.3(39.1)	60.6(70.6)	12.6(18.6)	18.1(27.2)	33.3
+DAPO	44.8(60.7)	33.1(46.3)	27.2(46.4)	25.0(38.8)	58.1(71.1)	11.6(17.5)	14.3(27.0)	30.6
+OVO	**57.3(62.4)**	**41.3(48.8)**	**40.6(49.2)**	**34.4(40.3)**	**67.0(72.0)**	**16.4(19.1)**	**23.5(28.0)**	**40.1**
+SVO	**57.3(62.1)**	40.8(48.5)	39.8(48.9)	33.6(40.0)	65.8(71.2)	15.8(18.8)	23.4(27.8)	39.5

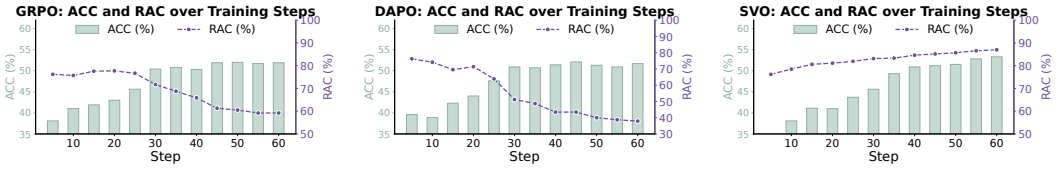


Figure 3: Training dynamics of Accuracy (ACC) and Reasoning–Answer Consistency (RAC). Left: GRPO. Middle: DAPO. Right: SVO.

4.2 Main Results

Tables 1 and 2 show that outcome-based RLVR often improves ACC while reducing RAC. SVO improves RAC and CCA over the base policies on every benchmark and outperforms GRPO and DAPO across all four backbones. Averaged over all backbone–benchmark pairs, SVO achieves 81.1 RAC and 35.9 CCA, exceeding GRPO by 14.7 and 6.2 percentage points, respectively. OVO achieves 82.3 RAC and 36.5 CCA, with higher averages for each backbone. SVO thus retains much of the benefit of direct judge supervision while replacing per-step judge rewards with a learned discriminator.

Figure 4 compares SVO with GRPO-CARE and ACRE on ConsisMath. SVO achieves the highest RAC, CCA, and ACC among these methods for both Qwen2.5-VL-3B and Qwen2.5-VL-7B. The gains show that its consistency benefits extend beyond comparison with outcome-only rewards. They support combining judge-derived supervision, self-verification, and adaptive scoring; the ablations below examine the discriminator refresh interval and the correctness gate. Appendix D additionally compares SVO with static SFT and direct judge supervision.

We examine two additional settings in Figure 5. ScienceQA (Lu et al., 2022) evaluates ConsisMath-trained models on multimodal science. Math500 evaluates Qwen2.5-3B-Instruct retrained on PRM800K/math_splits (Lightman et al., 2024), testing applicability to text-only mathematics. SVO improves RAC and CCA over GRPO and DAPO in both settings.

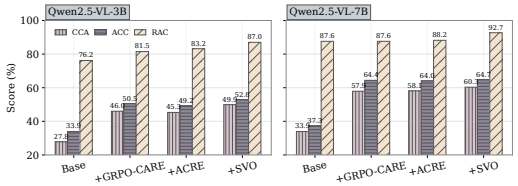


Figure 4: Alternative consistency supervision baselines on ConsisMath. We report RAC (%), CCA (%), and ACC (%) for Qwen2.5-VL-3B and Qwen2.5-VL-7B. SVO consistently achieves the best reasoning–answer consistency and consistency-constrained accuracy across both model scales.

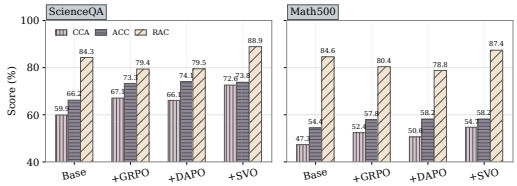


Figure 5: Generalization results on ScienceQA and Math500. We report RAC (%), CCA (%), and ACC (%) across multimodal and text-only reasoning tasks. SVO consistently improves reasoning consistency and consistency-constrained accuracy over GRPO and DAPO baselines.

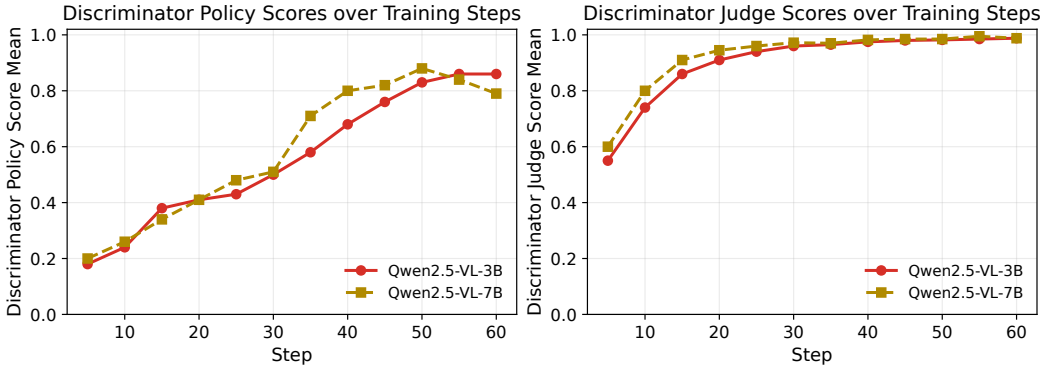


Figure 6: Evolution of Discriminator Scores under SVO Training.

These experiments provide complementary evidence: ScienceQA tests cross-task transfer of a trained policy, whereas Math500 tests the applicability of SVO to a text-only backbone.

Table 3 contextualizes SVO against prior multimodal reasoning methods on standard ACC benchmarks. SVO achieves competitive answer accuracy while targeting consistency. These comparisons complement the within-backbone results above, but do not isolate training-method effects across different systems.

Table 3: Comparison of Accuracy (ACC, %) with prior multimodal reasoning training approaches. We use bold and underlining to denote the best and second-best results, respectively. Highlighted row indicates SVO.

Method	MathVerse	MathVision	MathVista	WeMath
VOLD (Bousselham et al., 2025)	28.0	–	61.9	<u>31.8</u>
XReasoner (Liu et al., 2025a)	35.7	**24.4**	61.1	30.6
VLAA (Chen et al., 2025a)	<u>36.4</u>	**24.4**	61.0	31.5
VLM-R1 (Shen et al., 2025)	32.2	21.9	62.7	30.6
CalibRL (Huang et al., 2026)	–	<u>24.2</u>	<u>65.8</u>	–
SVO	**37.4**	<u>24.2</u>	**66.9**	**33.6**

4.3 Discussion and Analysis

Training dynamics. Figure 3 shows ACC holding steady or improving under GRPO and DAPO as RAC declines. This divergence is consistent with the limitation of answer-only rewards: correct responses receive the same outcome reward whether or not their reasoning supports the answer. SVO instead improves ACC and RAC together in the reported run. Its dense consistency reward and correctness gate provide a plausible mechanism for this alignment, with the gate ablation examined below.

Discriminator score dynamics. Figure 6 shows increasing mean scores for both policy outputs and judge-aligned references. The discriminator is updated while the policy and judge remain fixed; subsequent policy updates use its scores without token-level supervision from judge rationales. Because the scoring function also changes, rising scores alone cannot establish better verification or calibrated consistency estimates. We interpret them alongside the external RAC and CCA results. Appendix C provides additional entropy and cost diagnostics.

Supervision budget. The OVO–SVO comparison addresses the cost motivation directly at the level of judge queries. OVO queries the large judge for each rollout; SVO queries it for 32 current rollouts once every five policy steps, after warm-up. The discriminator scores the remaining policy updates, reusing information learned from earlier judge feedback. This schedule explains how SVO approaches OVO's consistency results with less frequent external supervision. The analytical accounting in Appendix C separates these judge calls from discriminator computation; a reduction in queries is not an end-to-end runtime measurement.

4.4 Ablation Studies

Parameter sensitivity. Table 4 varies the discriminator refresh interval. Refreshing every 5 policy steps achieves the highest RAC, CCA, and ACC among the tested settings. Both more frequent updates and less frequent or disabled updates perform worse, suggesting a balance between adapting the scorer and keeping policy feedback stable.

Component ablation. Table 5 compares full SVO with an ungated variant that replaces $R_i^{\text{acc}} R_i^{\text{cons}}$ with R_i^{acc}, while retaining the standalone consistency reward. Gating improves RAC, CCA, and ACC by 2.2, 4.0, and 1.1 percentage points, respectively. These results support conditioning the correctness reward on the learned consistency score. Appendix D.2 reports additional clipped and binary gate experiments.

Table 4: Discriminator update frequency. Metrics are in %; the highlighted row denotes the default setting.

Update interval	RAC	CCA	ACC
Every step	85.1	40.6	46.2
Every 5 steps	**87.0**	**49.9**	**52.8**
Every 10 steps	85.9	44.4	50.0
No updates	84.4	42.1	47.0

Table 5: SVO ablation on Qwen2.5-VL-3B. Metrics are in %; the highlighted row denotes the full model.

Method	RAC	CCA	ACC
Qwen2.5-VL-3B	76.2	27.8	33.9
w/o Gating	84.8	45.9	51.7
SVO (Full)	**87.0**	**49.9**	**52.8**

Consistency among correct answers. The fraction of correct answers judged reasoning-consistent, estimated by CCA/ACC, is 94.5% for full SVO ($49.9/52.8$), compared with 88.8% for the ungated variant ($45.9/51.7$). The ACC–CCA gap decreases from 5.8 to 2.9 percentage points. Thus, gating improves both answer accuracy and the proportion of correct answers supported by the generated reasoning.

5 Conclusion

SVO addresses the gap between expensive external consistency supervision and untrained policy self-verification. It distills judge-aligned assessments into a periodically refreshed discriminator and couples its feedback to generation through consistency and gated-correctness rewards. Across Qwen2.5-VL and Qwen3-VL, SVO improves RAC and CCA over outcome-based RLVR while retaining much of OVO's consistency benefit. Further analyses demonstrate the effectiveness of the discriminator refresh mechanism and the complementary roles of the proposed reward components. Overall, the results support learned consistency supervision as a practical and effective direction for improving the agreement between generated reasoning and answers in multimodal reasoning.

AI USE STATEMENT

Generative AI tools were used only to assist with English writing, including grammatical correction, wording refinement, and improvements to readability. They were not used to generate scientific content or conduct any part of the research. All research ideas, methodological designs, experiments, data processing, analyses, implementations, and conclusions were developed, conducted, and verified independently by the authors.

REPRODUCIBILITY STATEMENT

The anonymized implementation is available at `https://anonymous.4open.science/r/SVO_ICLR`. It provides the training and evaluation pipeline for SVO, including policy optimization, consistency-aware discriminator training, judge-supervised verification, and the corresponding experimental configurations. The SVO formulation and optimization procedure are described in Sections 3.2–3.4, while the datasets, evaluation protocols, and implementation details are documented in Section 4.1 and the appendices. External datasets and pretrained model weights can be prepared separately following the instructions provided in the repository.

REFERENCES

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

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Walid Bousselham, Hilde Kuehne, and Cordelia Schmid. Vold: Reasoning transfer from llms to vision-language models via on-policy distillation. *arXiv preprint arXiv:2510.23497*, 2025.

Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3–4):324–345, 1952. doi: 10.1093/biomet/39.3-4.324.

Hardy Chen, Haoqin Tu, Fali Wang, Hui Liu, Xianfeng Tang, Xinya Du, Yuyin Zhou, and Cihang Xie. Sft or rl? an early investigation into training r1-like reasoning large vision-language models. *arXiv preprint arXiv:2504.11468*, 2025a.

Yi Chen, Yuying Ge, Rui Wang, Yixiao Ge, Junhao Cheng, Ying Shan, and Xihui Liu. Grpo-care: Consistency-aware reinforcement learning for multimodal reasoning. *arXiv preprint arXiv:2506.16141*, 2025b.

Yuxin Chen, Yu Wang, Yi Zhang, Ziang Ye, Zhengzhou Cai, Yaorui Shi, Qi Gu, Hui Su, Xunliang Cai, Xiang Wang, An Zhang, and Tat-Seng Chua. Learning to self-verify makes language models better reasoners. *arXiv preprint arXiv:2602.07594*, 2026. URL `https://arxiv.org/abs/2602.07594`.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Yifan Du, Zikang Liu, Yifan Li, Wayne Xin Zhao, Yuqi Huo, Bingning Wang, Weipeng Chen, Zheng Liu, Zhongyuan Wang, and Ji-Rong Wen. Virgo: A preliminary exploration on reproducing o1-like mllm. *arXiv preprint arXiv:2501.01904*, 2025.

Kaixuan Fan, Kaituo Feng, Haoming Lyu, Dongzhan Zhou, and Xiangyu Yue. SophiaVL-R1: Reinforcing MLLMs reasoning with thinking reward. In *The Fourteenth International Conference on Learning Representations*, 2026. URL `https://proceedings.iclr.cc/paper_files/paper/2026/hash/0fb49ffa040d13a491c7164c5912cf7d-Abstract-Conference.html`.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.01798.

Minbin Huang, Runhui Huang, Chuanyang Zheng, Jingyao Li, Guoxuan Chen, Han Shi, and Hong Cheng. Answer-consistent chain-of-thought reinforcement learning for multi-modal large langauge models. *arXiv preprint arXiv:2510.10104*, 2025.

Zhuoxu Huang, Mengxi Jia, Hao Sun, Xuelong Li, and Jungong Han. Controllable exploration in hybrid-policy rlvr for multi-modal reasoning. *arXiv preprint arXiv:2602.20197*, 2026.

Zhehan Kan, Yanlin Liu, Kun Yin, Xinghua Jiang, Xin Li, Haoyu Cao, Yinsong Liu, Deqiang Jiang, Xing Sun, Qingmin Liao, et al. Taco: Think-answer consistency for optimized long-chain reasoning and efficient data learning via reinforcement learning in lvlms. *arXiv preprint arXiv:2505.20777*, 2025.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, JD Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal Behbahani, and Aleksandra Faust. Training language models to self-correct via reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2409.12917.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024.

Tianqianjin Lin, Xi Zhao, Xingyao Zhang, Rujiao Long, Yi Xu, Zhuoren Jiang, Wenbo Su, and Bo Zheng. Ravr: Reference-answer-guided variational reasoning for large language models. *arXiv preprint arXiv:2510.25206*, 2025.

Qianchu Liu, Sheng Zhang, Guanghui Qin, Timothy Ossowski, Yu Gu, Ying Jin, Sid Kiblawi, Sam Preston, Mu Wei, Paul Vozila, et al. X-reasoner: Towards generalizable reasoning across modalities and domains. *arXiv preprint arXiv:2505.03981*, 2025a.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025b.

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 *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://arxiv.org/abs/2209.09513.

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. *arXiv preprint arXiv:2310.02255*, 2023.

Ruotian Ma, Peisong Wang, Cheng Liu, Xingyan Liu, Jiaqi Chen, Bang Zhang, Xin Zhou, Nan Du, and Jia Li. S^2R: Teaching LLMs to self-verify and self-correct via reinforcement learning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 22632–22654. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.1104. URL https://aclanthology.org/2025.acl-long.1104/.

Fanqing Meng, Lingxiao Du, Zongkai Liu, Zhixiang Zhou, Quanfeng Lu, Daocheng Fu, Botian Shi, Wenhai Wang, Junjun He, Kaipeng Zhang, et al. Mm-eureka: Exploring visual aha moment with rule-based large-scale reinforcement learning. *CoRR*, 2025.

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

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. doi: 10.1609/aaai.v32i1.11671. URL https://ojs.aaai.org/index.php/AAAI/article/view/11671.

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

Xinyu Qiu, Heng Jia, Zhengwen Zeng, Shuheng Shen, Changhua Meng, Yi Yang, and Linchao Zhu. Unified generation and self-verification for vision-language models via advantage decoupled preference optimization. *arXiv preprint arXiv:2601.01483*, 2026. URL https://arxiv.org/abs/2601.01483.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

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

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

Haozhan Shen, Peng Liu, Jingcheng Li, Chunxin Fang, Yibo Ma, Jiajia Liao, Qiaoli Shen, Zilun Zhang, Kangjia Zhao, Qianqian Zhang, et al. Vlm-r1: A stable and generalizable r1-style large vision-language model. *arXiv preprint arXiv:2504.07615*, 2025.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, 2025.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://arxiv.org/abs/2305.04388.

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. *Advances in Neural Information Processing Systems*, 37:95095–95169, 2024.

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

Xiangyu Wen, Junhua Huang, Zeju Li, Min Li, Jianyuan Zhong, Zhijian Xu, Mingxuan Yuan, Yongxiang Huang, and Qiang Xu. Reasoning scaffolding: Distilling the flow of thought from llms. *arXiv preprint arXiv:2509.23619*, 2025.

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

Shicheng Xu, Liang Pang, Yunchang Zhu, Jia Gu, Zihao Wei, Jingcheng Deng, Feiyang Pan, Huawei Shen, and Xueqi Cheng. Distilling the implicit multi-branch structure in llms' reasoning via reinforcement learning. *arXiv preprint arXiv:2505.16142*, 2025.

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.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. Limo: Less is more for reasoning. *arXiv preprint arXiv:2502.03387*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 57905–57923. PMLR, 2024. URL https://proceedings.mlr.press/v235/yuan24d.html.

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*, pp. 169–186. Springer, 2024.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-Judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://arxiv.org/abs/2306.05685.

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. *arXiv preprint arXiv:2411.00836*, 2024.

Appendix

A Framework of Online-Verifiable Policy Optimization

Figure 7 illustrates OVO as defined in Section 3.2. The policy generates a reasoning trace, final answer, and self-verification judgment. A frozen judge evaluates the response and supplies a verification label, verification probabilities, and a rationale. The label gates the correctness reward, while the probabilities supervise the policy's self-verification distribution. A separate format reward checks output validity. OVO thus provides a direct judge-supervised reference for SVO, at the cost of repeated judge inference during policy training.

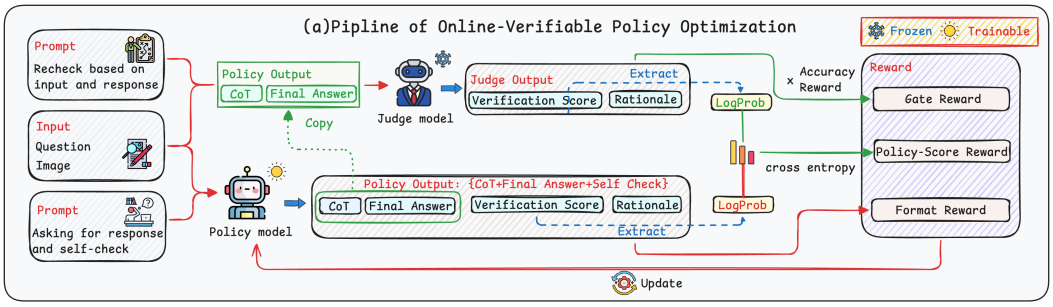


Figure 7: Framework of Online-Verifiable Policy Optimization (OVO). The policy model generates a reasoning trace, final answer, and self-verification judgment. A frozen judge model evaluates the generated response and provides judge-aligned verification labels, probabilities, and rationales. OVO then constructs the judge-gated accuracy reward, self-verification reward, and format reward for policy optimization.

B Algorithm: Self-Verifiable Policy Optimization (SVO)

Algorithm 1 summarizes SVO training. At each policy step, the policy samples a group of responses to an input, and the discriminator scores each response together with its verification probabilities. These scores define the consistency reward and gate the correctness reward. The frozen judge is invoked during initialization and periodic discriminator refreshes, not to compute the reward at every policy step. Refreshed references retain the policy's reasoning and answer while replacing its verification field with the judge's assessment. Thus, supervision is refreshed online rather than eliminated.

C Additional Training Diagnostics

Progressive consistency alignment. Figure 3 contrasts outcome-based RLVR with SVO. GRPO and DAPO can improve ACC while RAC declines, illustrating that answer correctness and output consistency need not improve together. An answer-only reward cannot distinguish two correct responses based on whether their written reasoning supports the answer. SVO introduces this distinction through a learned consistency score and a correctness gate, and its reported training curve shows ACC and RAC improving together. These aggregate trends support consistency-aware optimization, but do not identify which individual trajectories account for the difference.

Policy entropy. Figure 8 provides a complementary view of the reported training runs. SVO maintains an entropy level between GRPO and DAPO while improving RAC. GRPO exhibits a faster entropy decline, whereas DAPO maintains greater diversity without a corresponding consistency gain. These observations suggest that entropy alone does not explain reasoning–answer consistency: both the diversity of sampled responses and the reward assigned to them matter. The curves are descriptive diagnostics, not evidence that a particular entropy level is optimal or that entropy changes cause the observed consistency differences.

Algorithm 1 Self-Verifiable Policy Optimization (SVO)

Require: Policy model π_θ, discriminator D_ψ, frozen judge model π_{θ_J}, training dataset $\mathcal{D}$, discriminator update interval K

Initialize policy π_θ
Initialize discriminator D_ψ from π_θ
for each training step $t = 1, 2, \ldots$ **do**
  Sample input query $q \sim \mathcal{D}$
  Generate a group of reasoning trajectories $\{\tau_i\}_{i=1}^{N} \sim \pi_\theta(\cdot \mid q)$
  for each trajectory τ_i **do**
    Obtain final-answer prediction from τ_i
    Compute discriminator consistency score $D_\psi(\tau_i)$
    Construct $R_i^{\text{cons}}, R_i^{\text{gate}}, r_i = \lambda_f R_i^{\text{format}} + \lambda_g R_i^{\text{gate}} + \lambda_c R_i^{\text{cons}}$
  end for
  Compute group-wise normalized advantages using Eq. (12)
  Update policy π_θ using GRPO objective with consistency-aware rewards via Eq. (13)
  if $t \bmod K = 0$ **then**
    Construct paired samples (τ_i^P, τ_i^J) from policy-generated and judge-aligned trajectories
    Update discriminator D_ψ using Bradley–Terry preference loss via Eq. (11)
  end if
end for
return Optimized policy π_θ

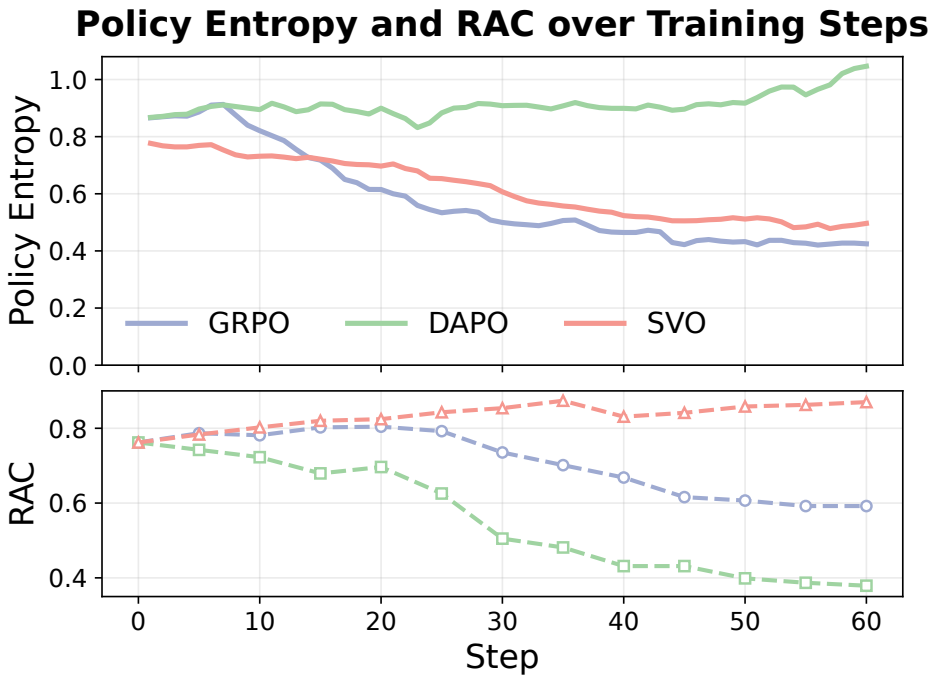


Figure 8: Policy Entropy and Reasoning–Answer Consistency (RAC) Dynamics during Training.

Training cost analysis. Table 6 compares selected training-cost terms analytically. Let T denote the number of policy steps, G the total sampled responses per step (rather than the per-query group in the policy objective), C_π the cost of one policy rollout-and-update step, C_J one large-judge inference, and C_D one discriminator update. Each refresh uses B_D current-step trajectories, with $B_D \ll G$; no historical rollout pool is relabeled. Under this decomposition, standard RLVR incurs only the shared policy cost. OVO additionally evaluates generated responses with the judge, contributing TGC_J. SVO refreshes its discriminator once every $K = 5$ policy steps using a subset of current responses, contributing approximately $(T/K)(B_D C_J + C_D)$ after warm-up. Its judge-query count is therefore lower when $B_D \ll GK$.

This accounting isolates the source of potential savings: fewer large-judge calls. It omits one-time warm-up and the discriminator forward passes used to score every rollout. SVO also incurs discrimi-

nator memory and update costs, so fewer judge calls do not by themselves establish lower end-to-end runtime or monetary cost. A hardware-matched measurement including all of these terms is needed to quantify practical efficiency relative to OVO and outcome-based RLVR.

Table 6: Analytical comparison of policy optimization and judge-refresh costs. The expressions exclude one-time warm-up and per-rollout discriminator-scoring costs; they are not measured end-to-end wall-clock times.

Method	Online Judge Calls	Training Cost
Standard RLVR	0	TC_π
OVO	$T \times G$	$TC_\pi + TGC_J$
SVO	$(T/K)B_D$	$TC_\pi + (T/K)(B_D C_J + C_D), K = 5$

D Additional Supervision Controls and Gate Variants

D.1 Static Offline SFT and Direct Judge Rewards

Table 7 reports the additional ConsisMath controls. For static SFT, multimodal GPT-4o generates image-conditioned CoT targets, because the frozen Qwen2.5-72B-Instruct training judge is text-only. This compares SVO with an offline-distillation alternative, but does not hold teacher identity or supervision modality fixed.

Table 7: Teacher-supervision controls on ConsisMath (%). SFT uses GPT-4o; direct judge reward and SVO use Qwen2.5-72B-Instruct. Bold denotes the best result within each column.

	Qwen2.5-VL-3B			Qwen2.5-VL-7B		
Method	RAC	CCA	ACC	RAC	CCA	ACC
Base	76.2	27.8	33.9	87.6	33.9	37.3
+SFT (GPT-4o)	81.5	40.9	49.2	89.6	46.2	50.2
+Direct judge reward	83.7	42.2	48.1	89.5	57.7	62.3
+SVO	**87.0**	**49.9**	**52.8**	**92.7**	**60.3**	**64.7**

The direct judge baseline uses the same frozen Qwen2.5-72B-Instruct judge as SVO and adds its binary consistency judgment directly to an ungated correctness reward:

$$r_i^{\text{direct}} = 0.1R_i^{\text{format}} + 0.5R_i^{\text{acc}} + 0.5R_i^{\text{verify},J}. \tag{14}$$

It differs from OVO, which uses a judge-gated correctness reward and a distribution-based self-verification reward (Section 3.2). The direct control therefore tests whether merely adding a judge reward accounts for SVO's gains; it does not isolate the discriminator, gate, and refresh schedule individually.

On 3B/7B, SVO improves RAC/CCA/ACC over SFT by 5.5/9.0/3.6 and 3.1/14.1/14.5 points, respectively, and over direct judge rewards by 3.3/7.7/4.7 and 3.2/2.6/2.4 points. The "No updates" control in Table 4 complements these comparisons by holding the judge and discriminator initialization supervision fixed while disabling subsequent refreshes. Full SVO gains 2.6/7.8/5.8 points over this static verifier. Together, these controls support the complete training design; none alone establishes superiority under equal total judge-call or compute budgets.

D.2 Gate Formulations

The GRPO baseline provides an outcome-only reference without discriminator-based consistency supervision. Table 8 instead compares four gate formulations while retaining the discriminator and its standalone consistency reward. Let $d_i = R_i^{\text{cons}} \in [0, 1]$ and let $v_i \in \{0, 1\}$ denote the policy's self-verification label. The reward is

$$r_i = 0.1R_i^{\text{format}} + 0.5R_i^{\text{acc}} g_i + 0.5d_i. \tag{15}$$

The four gate formulations are

$$
\begin{aligned}
g_i^{\text{ungated}} &= 1, \qquad g_i^{\text{continuous}} = d_i, \\
g_i^{\text{clipped}} &= \max(d_i, 0.5), \\
g_i^{\text{binary}} &= v_i \, \mathbb{I}[d_i \geq 0.5] + (1 - v_i) \, \mathbb{I}[d_i < 0.5].
\end{aligned}
\tag{16}
$$

Only the gate multiplier changes across these variants. The binary gate also depends on the self-verification label v_i, rather than thresholding d_i alone. Because d_i is a learned preference score, the threshold of 0.5 is an experimental choice rather than a calibrated consistency boundary.

Table 8: Gate ablations on ConsisMath with Qwen2.5-VL-3B (%). The base, ungated, and full-SVO results reproduce Table 5; the clipped and binary gates are additional experiments.

Method	RAC	CCA	ACC
Qwen2.5-VL-3B	76.2	27.8	33.9
w/o Gating	84.8	45.9	51.7
Clipped gate	80.5	42.4	50.6
Binary gate	83.4	43.8	50.0
SVO (Full)	**87.0**	**49.9**	**52.8**

Clipping preserves at least half of the correctness reward, including for low-scoring responses; binary gating discards score magnitude and introduces a discontinuity. The continuous gate performs best among these tested variants. These results do not establish calibration, causal faithfulness, or robustness under controlled discriminator noise, and they do not separately evaluate unconventional but valid reasoning.

E Evaluation Judge Agreement and Stability

We assess GPT-5.5 as the external consistency evaluator through agreement with human annotations and repeated evaluations. These checks characterize the reported evaluation protocol; they do not establish that the evaluator is error-free or equally reliable outside the tested settings.

LLM-based evaluation can agree with human judgments while remaining sensitive to judge biases (Zheng et al., 2023). We therefore assess agreement and repeatability for the consistency protocol used here.

Human agreement analysis. On the ConsisMath test set of 1,695 instances, three human experts independently label the consistency of outputs from **Qwen2.5-VL-3B** and **Qwen2.5-VL-7B**. We use majority voting to obtain the human reference label and apply GPT-5.5 to the same outputs. Agreement with the human majority exceeds **97%**. This result supports the evaluator's agreement with the annotation protocol on these outputs; it is not an estimate of universal correctness or of agreement between the human annotators themselves.

Stability analysis. We repeat GPT-5.5 evaluation five times on fixed outputs from **Qwen2.5-VL-3B** and **Qwen2.5-VL-7B** across ConsisMath, LogicVista, MathVerse, WeMath, MathVista, DynaMath, and MathVision. The reported variation in aggregate consistency scores is below **0.5%** on every benchmark. This indicates limited run-to-run variation in the aggregate scores for these fixed outputs, although it does not imply identical labels for every instance.

Together, these checks support using the evaluator for the reported RAC and CCA comparisons. Fine-grained error analysis, class-conditional agreement, and agreement across additional judges would provide a more complete assessment of evaluation uncertainty.

F Training Settings

F.1 Hyperparameter Settings

The main multimodal experiments use the same ConsisMath training split for three epochs, with a batch size of 256 and eight responses per instance. The policy uses a cosine learning-rate schedule

with a peak of 1×10^{-6}, a sampling temperature of 1.0, and a maximum response length of 1,024 tokens. The discriminator follows the same learning-rate schedule and is updated once every five policy steps. Each refresh uses 32 current-step responses, paired with newly generated judge-aligned counterparts; it does not use a full batch or a replay buffer. The judge remains frozen and uses a sampling temperature of 0.0.

For SVO, the format reward R^{format} has weight $\lambda_f = 0.1$, while the gated correctness reward R^{gate} and discriminator consistency reward R^{cons} have weights $\lambda_g = 0.5$ and $\lambda_c = 0.5$, respectively. The outcome-based baselines combine the same format reward, weighted by $\lambda_f = 0.1$, with the answer-accuracy reward R^{acc}, weighted by $\lambda_a = 1.0$.

F.2 Prompt Design

F.2.1 Policy Model Prompt

The policy prompt exposes self-verification as part of the generated response. Given a query, the policy produces (i) a CoT, (ii) a final answer, (iii) a binary judgment of whether the reasoning supports that answer, and (iv) a brief justification. The judgment concerns agreement between the written reasoning and the answer; it is not a separate check against the ground-truth solution. This structured field gives training an observable self-assessment to supervise, but prompting alone does not guarantee that the assessment is reliable. The full prompt template is reproduced below.

F.2.2 Judge Model Prompt

The judge receives the query, generated CoT, and extracted answer, and returns a binary consistency label with a short rationale. Its prompt directs it to identify the explicit conclusion in the reasoning and compare that conclusion with the final answer. The intended task is consistency assessment rather than independently solving the problem. Accordingly, this protocol does not certify every intermediate mathematical step or establish that the written reasoning caused the answer. The full judge prompt is reproduced below.

Prompt Templates for Policy Model and Judge Model

(a) Policy Model Prompt Template

<SYSTEM>
You are an AI assistant that rigorously follows this response protocol:

1. First, conduct a detailed analysis of the question. Think step by step, consider different angles, and reason through the problem carefully. Enclose this entire reasoning process within `<think>` `</think>` tags.

2. After the thinking section, provide a clear, concise, and direct answer. The final answer must be enclosed in `boxed{}`.

3. After the evaluation, output a verification check and a brief reason in ONE line, using the strict format below.

Evaluation rules:
- Identify the final explicit conclusion in the <think> </think> section.
- Compare that conclusion with the extracted `boxed{}` answer.
- Output `\check{1}` only if the reasoning clearly supports the same final answer.
- Output `\check{0}` if the reasoning leads to a different conclusion, does not reach a clear conclusion, or is inconsistent with the final answer.
- Provide a brief reason explaining why the reasoning supports or does not support the extracted answer.

Output format example:

```
<think> ... reasoning ... </think>
\boxed{final answer}
\check{1 or 0}<reason>brief explanation</reason>
```

Ensure the final answer and check outputs do not reference the reasoning text.

</SYSTEM>

<USER>
[Question]
{question}
</USER>

(b) Judge Model Prompt Template

<SYSTEM>
You are an evaluation assistant. Your only task is to determine whether the final conclusion derived in the <think> reasoning section is consistent with the final answer inside \boxed{}.

I will give you:
1. The complete problem.
2. The thinking process in the model response enclosed within <think> tags.
3. The extracted final answer from \boxed{}.
Evaluation rules:
- Identify the final explicit conclusion reached in the <think> section.
- Compare that conclusion with the extracted \boxed{} answer.
- Output \check{1} only if the reasoning clearly supports the same final answer.
- Output \check{0} if the reasoning leads to a different conclusion, does not reach a clear conclusion, or is inconsistent with the final answer.
- Provide a brief reason explaining why the reasoning supports or does not support the extracted answer.

Output format example:

\check{1 or 0}<reason>brief explanation</reason>

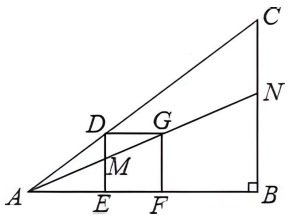
</SYSTEM>

<USER>
[Question]
{question}
[Problem]
{problem}
[Model Thinking Process]
{response}
[Extracted Answer from \boxed{}]
{extracted answer}
</USER>

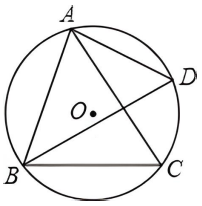
Figure 9: Prompt templates used for policy generation and judge-based consistency evaluation.

G Reasoning–Answer Inconsistency and Related Failure Modes

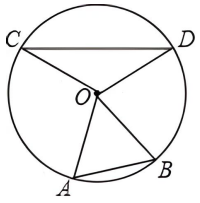
Manual inspection reveals recurring forms of reasoning–answer mismatch and related weaknesses in generated CoTs. The examples below are illustrative rather than an exhaustive taxonomy or a quantitative error analysis. They distinguish unsupported answers from broader issues such as unnecessary reasoning: the latter need not constitute inconsistency under the RAC protocol.
Problems. The following geometric configurations correspond to the examples discussed below.



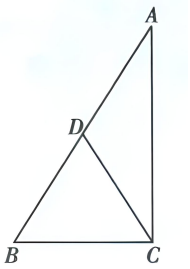
1508



2750



189



31968

Answer-driven option guessing. The selected option does not follow from the numerical conclusion in the generated reasoning. This observable mismatch is consistent with option guessing, although the output alone does not reveal how the model selected its answer. The failure relevant to RAC is the lack of support for that answer in the written CoT.

Internally inconsistent reasoning. The response has the surface structure of a CoT, but its steps contain incompatible statements and do not establish the reported answer. This example illustrates why a well-formed reasoning trace is insufficient evidence of logical support. It concerns the observable content of the response rather than the model's internal computational process.

Irrelevant reasoning. The model derives the requested quantity and then continues with unnecessary angle calculations. This is a relevance and concision issue, not necessarily a reasoning–answer mismatch: additional material does not invalidate an earlier derivation unless it contradicts or undermines the answer. We retain the example as a related CoT failure mode and do not treat verbosity alone as evidence of inconsistency.

Incomplete reasoning. The model identifies a plausible solution approach but stops before carrying out the computations needed to obtain the final answer. The resulting CoT therefore leaves a gap between the proposed method and the reported result. The relevant issue is missing support for the answer, rather than the length of the response itself.

Examples of Reasoning–Answer Inconsistency and CoT Failure Modes

(a) Answer-driven Option Guessing
Problem ID: 1508
Ground Truth. D

Model Output.

```
<think>Given that DEFG is a square with side length 1, and EF is parallel to BC, we
can use the properties of similar triangles to find the length of BN. Since DE is
parallel to BC, triangles ADE and ABC are similar.
The ratio of the sides of these triangles is the same as the ratio of their
corresponding heights. The height of triangle ADE is DE = 1, and the height of
triangle ABC is BC = 3. Therefore, the ratio of the sides is 1/3.
Since EF is parallel to BC, triangle AEF is similar to triangle ABC, and the ratio
of their corresponding sides is also 1/3. The length of EF is 1, so the length of
AE is 3. Since AB = 4, the length of BE is 1.
Now, since DEFG is a square, the length of DG is also 1. Therefore, the length of
BN is the same as the length of BE, which is 1. </think>
\boxed{D}
```

(b) Internally Inconsistent Reasoning
Problem ID: 2750
Ground Truth. 56°

Model Output.

```
<think>
In a triangle, the sum of the interior angles is always 180 degrees. Given angle ABC
= 71 degrees and angle CAB = 53 degrees, we can find angle BAC by subtracting the
sum of the other two angles from 180 degrees. Then, since angle ADB is an inscribed
angle that intercepts the same arc as angle BAC, it will be half of angle BAC.
</think>
\boxed{59^\circ}
```

(c) Irrelevant Reasoning
Problem ID: 189
Ground Truth. (C)

Model Output.

```
<think>
Let the radius of the circle be R. For a chord subtending a central angle alpha,
its length is L = 2R * sin(alpha / 2).
From AB = 2 and CD = 4, together with the fact that the two central angles are
supplementary, we obtain R^2 = 5, so R = sqrt(5). Thus, the radius of the circle is
sqrt(5).
After determining the radius, we can further analyze the angle configuration. Since
sin(alpha / 2) = 1 / sqrt(5) and cos(alpha / 2) = 2 / sqrt(5), the angle alpha / 2
is approximately 26.6 degrees, and the supplementary central angle is approximately
126.9 degrees.
</think>
\boxed{C}
```

(d) Incomplete Reasoning
Problem ID: 31968
Ground Truth. 5

Model Output.

```
<think>
In a right triangle, the median to the hypotenuse is half the length of the hypotenuse.
Therefore, CD = 1/2 AB. To find AB, we can use the sine of angle B, which is the
ratio of the opposite side AC to the hypotenuse AB.
</think>
\boxed{10}
```

Figure 10: Representative examples of reasoning–answer inconsistency and CoT failure modes under outcome-based RLVR supervision. The examples include answer-driven option guessing, internally inconsistent reasoning, irrelevant reasoning, and incomplete reasoning. These cases demonstrate that final-answer correctness alone cannot guarantee that the generated reasoning supports the prediction.