O^2-CRITICURL: OFFLINE-ONLINE CURRICULUM REINFORCEMENT LEARNING FOR CRITICAL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models exhibit strong capabilities on reasoning tasks, yet often produce flawed intermediate steps while still yielding correct final answers. This behavior undermines interpretability and reliability, suggesting reliance on spurious shortcuts rather than faithful reasoning. Although earlier efforts have explored step-level supervision, distinguishing decisive steps from redundant ones remains challenging. We propose O^2-CritiCuRL, a novel curriculum reinforcement learning framework that introduces critical-step awareness through an iterative offline–online paradigm. In the offline stage, O^2-CritiCuRL decomposes chain-of-thought trajectories, automatically identifies decisive steps while down-weighting redundant ones, and then restructures the trajectories into difficulty-based tiers for curriculum learning. In the online stage, we employ a progressive step-level reinforcement learning strategy, where truncated chains guide the model to infer missing steps and refine its reasoning. Coupled with the offline stage, this iterative process gradually sharpens the model's focus on critical steps and mitigates the limitations of static supervision. Extensive experiments on multiple reasoning benchmarks show that our method not only achieves state-of-the-art performance but also delivers superior training and inference efficiency.

1 INTRODUCTION

Recent advances in large language models (LLMs) have demonstrated remarkable capabilities in multi-step reasoning across domains such as mathematics, code generation, and scientific discovery (Wang et al., 2023; Guo et al., 2024; Comanici et al., 2025). A common strategy to enhance such abilities is curriculum learning, which, inspired by human education, gradually exposes models to increasingly complex examples (Bengio et al., 2009). By regulating task difficulty, curriculum learning helps stabilize optimization and improves generalization, making it an effective paradigm for complex reasoning tasks (Zeng et al., 2024; El-Kishky et al., 2025). Reinforcement learning (RL) plays a central role in these advances by optimizing models with process- or outcome-based rewards (Hendrycks et al., 2021; Stiennon et al., 2022; Wang et al., 2024b). Among RL methods, Group Relative Policy Optimization (GRPO) has gained prominence for its efficiency and scalability, which encourages "slow thinking" by guiding models to generate and refine multiple reasoning paths(Shakya et al., 2023; Comanici et al., 2025; Guo et al., 2025). As a result, the integration of curriculum learning and reinforcement learning has therefore proven effective in strengthening LLMs' performance on reasoning-intensive tasks(Yuan et al., 2025a; Shao et al., 2024).

Nevertheless, models are often observed to generate flawed intermediate reasoning steps yet still reach the correct final answer (Ling et al., 2023; Lightman et al., 2023b). While this may preserve accuracy, it raises concerns about interpretability and reliability, as the reasoning process may hinge on spurious correlations rather than robust logic (Poursabzi-Sangdeh et al., 2021). Such behavior undermines trust in high-stakes applications that demand transparent step-by-step reasoning, including education, scientific research, and legal or medical decision support (Arrieta et al., 2020; Benda et al., 2022). This phenomenon can be regarded as a form of resembles reward hacking, where a model attains high scores by exploiting shortcuts instead of engaging in faithful reasoning. (Amodei et al., 2016; Stiglic et al., 2020). Unlike traditional reward hacking, which arises from flawed reward design, this issue stems from answer-only supervision that rewards correct outcomes while neglecting the reasoning process (Skalse et al., 2022; Lightman et al., 2023a).

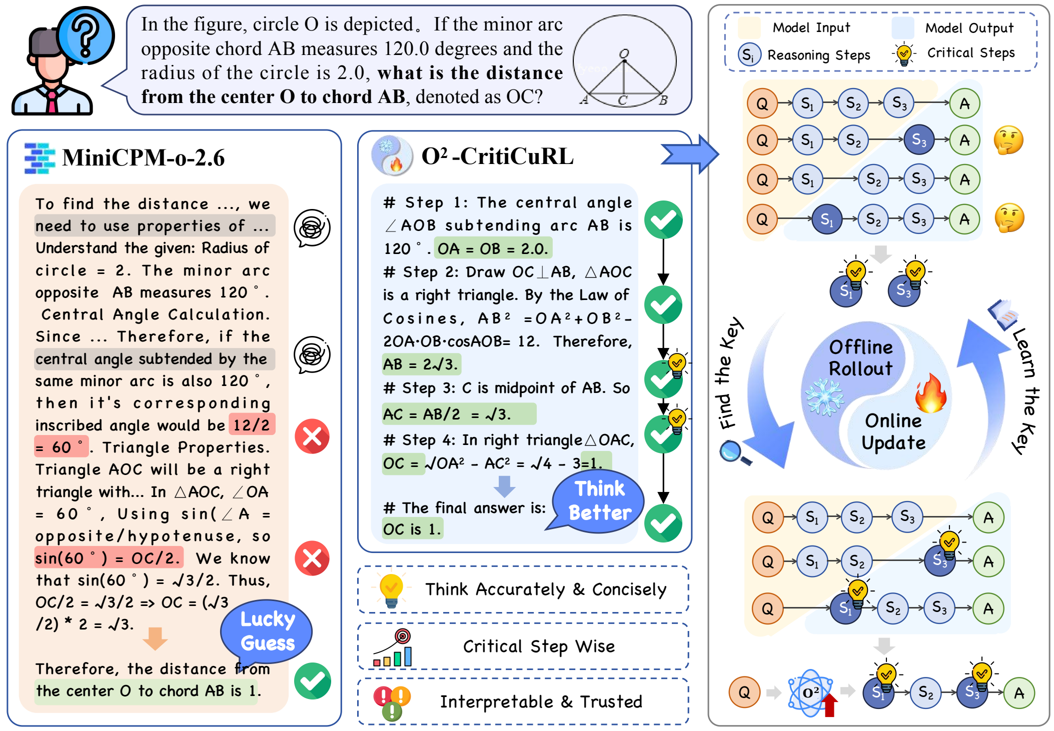


Figure 1: **Overview of O^2-CritiCuRL.** Given a mathematical problem (left), MiniCPM-o-2.6 generates a verbose and potentially redundant chain-of-thought (CoT) trajectory, which may include incorrect or non-critical steps (e.g.,"Lucky Guess"). In contrast, O^2-CritiCuRL identifies critical reasoning steps—those whose removal significantly degrades performance—via offline decomposition and self-designed reward analysis. These critical steps are highlighted (green checkmarks), while redundant or erroneous ones are down-weighted or pruned. The refined trajectories are organized into difficulty tiers for curriculum learning. During the online stage, the model undergoes progressive reinforcement learning: reasoning chains are gradually truncated, prompting the model to infer missing critical steps and the final answer. This iterative loop between offline critical-step identification and online update enables the model to learn accurate, concise, and trustworthy reasoning paths ("Think Better"), ultimately improving both accuracy and interpretability.

A natural solution is to incorporate step-level supervision into training framework. Aligning model behavior with human-annotated reasoning chains is able to guarantee the integrity of intermediate reasoning steps in principle (Cheng et al., 2024; Luo et al., 2024). However, available datasets often contain long, complex, and redundant trajectories, making direct training computationally expensive and cognitively diffuse (Qu et al., 2025; Lu et al., 2025). As every step is treated indiscriminately, the model struggles to identify and prioritize the decisive parts of reasoning. Reinforcement learning with process-level rewards offers an alternative by encouraging models to refine reasoning paths dynamically rather than imitating fixed trajectories, but it incurs high computational cost due to reliance on additional reward models and remains vulnerable to reward hacking (Wang et al., 2024c; Chen et al., 2025).These limitations highlight the need for training paradigms that adaptively emphasize decisive-step awareness while maintaining efficiency and robustness.

To address these challenges, we propose O^2-CritiCuRL, a novel offline–online curriculum reinforcement learning framework. Rather than treating all reasoning steps equally, O^2-CritiCuRL automatically identifies the steps most decisive for deriving correct answers. In the offline stage, long chain-of-thought trajectories are decomposed, and a self-designed reward is used to detect critical steps: those whose removal significantly degrades performance are marked as critical, while redundant steps are down-weighted or reorganized. The resulting trajectories are then restructured into difficulty tiers to ensure curriculum learning focuses on decisive reasoning patterns.

In the online stage, we introduce a progressive step-level reinforcement learning strategy guided by step-level rewards. By gradually truncating reasoning chains, the model is encouraged to infer missing critical steps and final answers, enabling step-wise acquisition of robust reasoning ability.

Static supervision and fixed curriculum learning fail to capture the evolving nature of critical reasoning steps. Early-stage models often misidentify these steps, and the truly decisive ones emerge only as the model improves, while noisy reward signals further exacerbate the challenge. To address these challenges, O^2-CritiCuRL adopts an iterative training mechanism that alternates between offline critical-step identification and online reinforcement learning, continuously refining the model's focus as training progresses. This iterative loop yields improved accuracy, enhanced interpretability, and more stable optimization. Our framework alternates between offline identification of critical steps and online reinforcement learning, with the updated model fed back into the offline stage for refined step re-identification. Through this iterative loop, the model gradually sharpens its focus on critical steps, yielding improved accuracy, enhanced interpretability, and greater training stability. In summary, our contributions are threefold:

- We introduce O^2-CritiCuRL, the first curriculum reinforcement learning framework that explicitly models critical-step awareness, automatically detecting decisive reasoning steps and restructuring trajectories for efficient and principled reasoning supervision.
- We develop a novel iterative offline–online training mechanism, where critical-step discovery and reinforcement learning co-evolve, enabling progressive refinement of reasoning with improved accuracy, interpretability, and stability..
- O^2-CritiCuRL substantially improves reasoning performance, while also offering greater training efficiency compared to prevailing step-level reinforcement learning approaches.

2 RELATED WORK

2.1 CURRICULUM LEARNING AND REINFORCEMENT LEARNING FOR REASONING

Curriculum learning is a training paradigm that organizes tasks in increasing order of difficulty to promote smoother and more effective model learning (Bengio et al., 2009). In the domain of LLMs and VLMs, it has been widely adopted to stabilize optimization and enhance reasoning generalization. From the perspective of reward design, existing works can be categorized as result-based or process-based rewards. Result-based methods evaluate the final answer correctness or similarity as the primary criterion, such as Light-R1 (Wen et al., 2025), Curr-ReFT (Deng et al., 2025), VLM-R1 (Shen et al., 2025), Vision-G1 (Zha et al., 2025), Infi-MMR (Liu et al., 2025a), ASTRO (Kim et al., 2025), and JT-Math (Hao et al., 2025). In contrast, process-based reward approaches explicitly assess intermediate steps, as in R-PRM (She et al., 2025), VisualPRM (Wang et al., 2025a), Step-GRPO (Zhang et al., 2025a), DocThinker (Yu et al., 2025), and ALPHALLM-CPL (Wang et al., 2024c). These methods provide finer-grained supervision signals but still face challenges such as high reward modeling costs, limited scalability to multimodal settings, and insufficient ability to differentiate truly critical steps. From the perspective of training phase design, works like Curr-ReFT (Deng et al., 2025) and PCuRL (Yuan et al., 2025a) introduce explicit progressive stages, while Infi-MMR (Liu et al., 2025a), E2D (Parashar et al., 2025), and GHPO (Liu et al., 2025b) employ dynamic scheduling functions or adapt between imitation and reinforcement learning. Although effective in structuring learning, these approaches generally rely on static difficulty annotations or handcrafted schedules, and rarely account for the evolving importance of intermediate reasoning steps. Our work builds on these insights by introducing a curriculum reinforcement learning framework that explicitly integrates step-level supervision with critical-step awareness.

2.2 SUPERVISION STRATEGIES FOR REASONING PROCESSES

Traditional answer-only supervision enables models to bypass reasoning trajectories as long as the final answer is correct, which undermines interpretability and reliability (Lightman et al., 2023a). To mitigate this, recent efforts incorporate intermediate reasoning rewards, such as R-PRM (She et al., 2025), VisualPRM (Wang et al., 2025a), and StepGRPO (Zhang et al., 2025a), which evaluate the quality of intermediate steps. Although these methods provide more transparent supervision, they tend to treat all steps equally, ignoring the fact that many reasoning trajectories contain redundant or low-impact steps (Qu et al., 2025; Arora et al., 2025). Several approaches attempt to address this issue through data selection or trajectory restructuring. For example, DUMP (Wang et al., 2025c), Writing-RL (Lei et al., 2025), and Vision-G1 (Zha et al., 2025) prioritize high-value samples via bandit mechanisms or influence functions, while EduFlow (Zhu et al., 2025a) and VersaPRM (Zeng et al., 2025) leverage large-scale preference annotations. However, these strategies remain limited in their ability to dynamically pinpoint which reasoning steps are critical for problem-solving. Our work differs in explicitly modeling critical-step awareness. We propose an offline–online framework that automatically detects decisive steps, restructures reasoning trajectories accordingly, and iteratively refines supervision as the model evolves. By progressively focusing on the most impactful reasoning components, our method improves accuracy, interpretability, and training efficiency.

3 THEORETICAL PROOFS

Assumption: We define a critical step as the reasoning step that contributes the most to deriving the correct answer. To formalize this notion, we assume that each problem admits a set of possible answers that are uniformly distributed within an answer space. A reasoning process can then be viewed as a sequence of partitions of this space, where each step progressively reduces the set of candidate answers. Within this framework, a critical step corresponds to the partition that achieves the greatest reduction in the answer space, thereby maximizing the probability of selecting the correct answer.

Definition: To establish a theoretical foundation for quantifying critical steps, we begin by defining the probabilistic state space of the reasoning process. Let Ω be the set of all possible answers. The model's reasoning state is represented by a probability distribution over Ω. $P_0 = \text{Uniform}(\Omega)$ denotes the initial distribution without any steps, while P_i denotes the conditional distribution given steps $S_{1:i}$. To measure the uncertainty of the answer space, we introduce the Shannon Entropy, and the contribution of the i-th step can then be quantified as the entropy reduction in Δp:

$$P_i = P(\cdot \mid S_{1:i}), \quad H(P) = -\sum_{a\in\Omega} P(a)\log P(a), \quad \Delta H_i = H(P_{i-1}) - H(P_i) \tag{1}$$

Within our formulation, the reduction in uncertainty of the conditional distribution directly manifests as compression of the probability space, effectively eliminating a substantial portion of incorrect answers. (i.e., entropy reduction). The degree and efficiency of probability distribution changes triggered by each reasoning step are central to measuring its criticality. The **relative entropy** (Kullback-Leibler Divergence) naturally quantifies this distribution change, making it our information-theoretic measure for the criticality of step S_i:

$$\mathcal{K}_i = D_{\text{KL}}(P_i \parallel P_{i-1}) = \sum_{\omega\in\Omega} P_i(\omega)\log\frac{P_i(\omega)}{P_{i-1}(\omega)} \tag{2}$$

Step 1: Decomposition of Relative Entropy. We decompose the relative entropy into contributions from the correct answer ω^* and the space of incorrect answers ω:

$$\mathcal{K}_i = \underbrace{P_i(\omega^*)\log\frac{P_i(\omega^*)}{P_{i-1}(\omega^*)}}_{\text{Contribution from correct answer probability increase}} + \underbrace{\sum_{\omega\neq\omega^*} P_i(\omega)\log\frac{P_i(\omega)}{P_{i-1}(\omega)}}_{\text{Contribution from incorrect answer space compression}} \tag{3}$$

Step 2: Uniform Compression Assumption. The second term remains intractable without additional assumptions. To address this, we assume that all incorrect answers are *uniformly compressed*, i.e., their probabilities decrease by the same proportion. This abstraction eliminates spurious biases and is consistent with the principle of maximum entropy. It further reflects the intuition that a decisive step reduces the probability of all incorrect answers and shifts the mass toward the correct one.

$$\forall\omega \neq \omega^*, \quad \frac{P_i(\omega)}{P_{i-1}(\omega)} = \frac{1-p_i}{1-p_{i-1}}, \quad \text{where } p_i = P_i(\omega^*) \tag{4}$$

Step 3: Simplifying Contributions. Based on the uniform compression assumption, the contribution from incorrect answers simplifies to the logarithmic change of their total probability:

$$\sum_{\omega\neq\omega^*} P_i(\omega)\log\frac{P_i(\omega)}{P_{i-1}(\omega)} = (1-p_i)\log\frac{1-p_i}{1-p_{i-1}} \tag{5}$$

Meanwhile, the contribution from the correct answer remains:

$$P_i(\omega^*) \log \frac{P_i(\omega^*)}{P_{i-1}(\omega^*)} = p_i \log \frac{p_i}{p_{i-1}} \tag{6}$$

Step 4: Criticality Metric Formula. Combining both contributions yields final criticality metric, which is fully characterized by the model's probability assigned to the correct answer at step S_i:

$$\boxed{\mathcal{K}_i = p_i \log \frac{p_i}{p_{i-1}} + (1 - p_i) \log \frac{1 - p_i}{1 - p_{i-1}}} \tag{7}$$

Boundary Behavior and Robustness. The criticality measure $\mathcal{K}i$ exhibits intuitive boundary behavior consistent with its information-theoretic interpretation: it diverges to $+\infty$ when a step raises the probability from near zero to a finite value, and it vanishes as $pi - 1 \rightarrow 1$, since no further entropy reduction is possible. In practice, due to cold-start conditions and the finite capability of the model, the system never reaches these extremes. (e.g. steps are neither infinitely informative nor entirely negligible.) Moreover, the logarithmic form of $\mathcal{K}i$ naturally suppresses the influence of small statistical fluctuations. When $p_i \approx p_{i-1}$ and both are away from the extremes, $\mathcal{K}i \ll |p_i - p_{i-1}|$, which reduces false positives caused by random noise.

4 METHOD

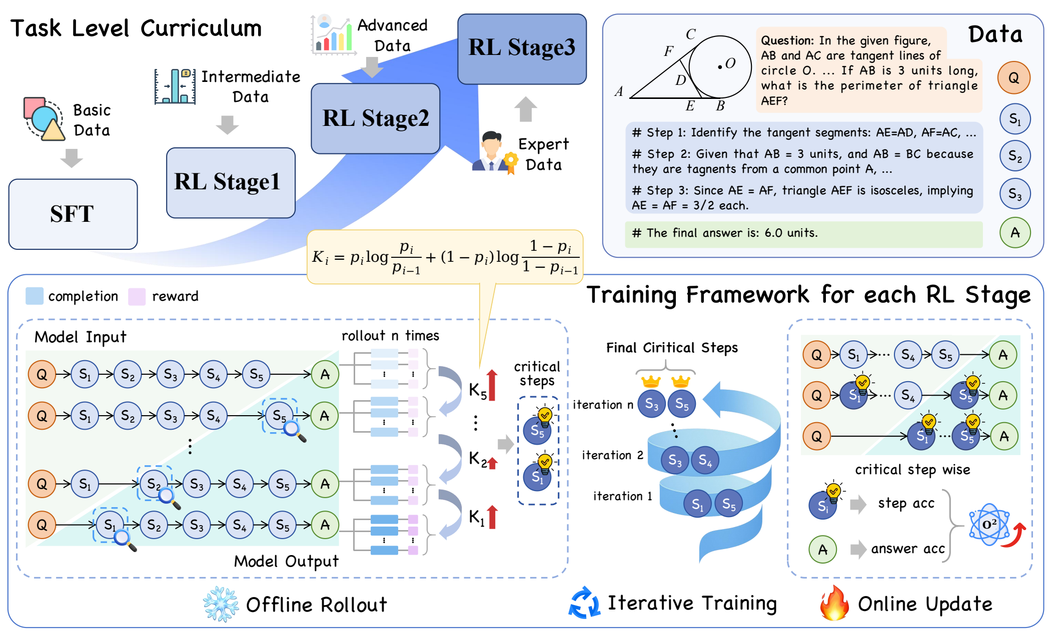


Figure 2: **The overview of our proposed O**^{2}**-CritiCuRL.** Our method conducts curriculum learning at the stage level according to task difficulty. Within each stage, we further split training into two phases including offline rollout and online update, and train between these two phases iteratively, thereby achieving curriculum learning at the critical step level.

In this section, we introduce our proposed framework "O^2-CritiCuRL". The methodology begins with supervised fine-tuning on existing data to provide a cold-start initialization for the model. Subsequently, large language models (LLMs) are employed to segment reasoning trajectories, which serve as the basis for identifying potential critical steps. Building on this foundation, we adopt an iterative optimization strategy: in the offline phase, critical steps are identified, while in the online phase, these identified steps are leveraged to update the model. The updated model is then fed back into the offline phase for refined identification, forming a closed loop of alternating offline and online optimization. This process is repeated until clear convergence criteria are met, ensuring both the stability of critical step identification and the effectiveness of reinforcement learning.

4.1 DATA PROCESS

First, we leverage large language models to decompose the long-chain reasoning process into explicit steps. Specifically, to mitigate the risk of reward hacking during reinforcement learning, we first convert all multiple-choice questions in the original dataset into open-ended question-answering format. Subsequently, we employ the GPT to analyze the converted data and perform an initial segmentation of critical steps within the long-chain CoT reasoning trajectories as illustrated in equation 8, where P_i represents the i-th reasoning step in the long-chain CoT process. Finally, we categorized the data into four difficulty levels (e.g., *basic*, *intermediate*, *advanced*, and *expert*) based on the number of reasoning steps, thereby distinguishing samples with varying levels of complexity.

$$f_{\text{seg}}^{\text{GPT}} : \{P_1, P_2, \dots, P_n\} \;\; \longrightarrow \;\; \{\text{Step 1: } P_1,\ \text{Step 2: } P_2,\ \dots,\ \text{Step n: } P_n\} \tag{8}$$

4.2 MODEL INITIALIZATION

We adopt Qwen-2.5-VL as the backbone model and initiate the training process with a cold-start strategy. During the cold-start phase, we employ *basic* data for supervised fine-tuning (SFT) to facilitate the initial development of the model's fundamental reasoning capabilities. To enable curriculum learning that gradually exposes the model to harder samples, we employ the remaining data within our framework, structuring training into three progressive stages. Unlike conventional curriculum learning approaches that primarily adjust task difficulty in a coarse-grained manner, we introduce a two-phase progressive refinement framework in each stage.

4.3 OFFLINE STEP DISTILLATION

In the offline phase, the model autonomously identifies and extracts critical reasoning steps, effectively reducing redundancy and capturing essential structures of long-chain reasoning. As shown in Figure 2, we perform offline rollouts over the entire dataset to identify critical reasoning steps. For each instance, the segmented reasoning steps are concatenated with the original question in a sequential manner. Specifically, each step is appended to the question one by one, forming inputs of the model, until the complete chain of reasoning is reconstructed:

$$\mathcal{I} = \{\, q,\ q + s_1,\ q + s_1 + s_2,\ \dots,\ q + s_1 + s_2 + \cdots + s_T \,\} \tag{9}$$

We sequentially feed the constructed inputs into the model and evaluate the reward based on whether the predicted answer matches the ground truth. For each input, we perform n independent rollouts, obtaining n reward samples. These samples form an empirical distribution that approximates the model's behavior over the answer space for the given input. Following the practical refinement of Eq. (7), we introduce several adjustments to ensure stability under extreme probability cases. First, both p_i and p_{i-1} are clipped into the interval $[\varepsilon, 1-\varepsilon]$ to avoid degenerate values at 0 or 1. Second, a small lower bound is imposed when p_i falls below a threshold (e.g., 0.01), preventing disproportionately large contributions to the metric. Third, if p_i and p_{i-1} are nearly indistinguishable (e.g., $|p_i - p_{i-1}| < \varepsilon$) or if p_i exceeds $1-$tol, the corresponding contribution is set to zero, reflecting the absence of meaningful information gain in such cases:

$$m_i = \min\Big(\max(p_i, \max(\varepsilon, 0.01)),\ 1-\varepsilon\Big) \tag{10}$$

Under these refinements, we use the stabilized probabilities m_i and m_{i-1} to compute the criticality score K between adjacent steps as following:

$$K_i = m_i \log \frac{m_i}{m_{i-1}} + (1-m_i) \log \frac{1-m_i}{1-m_{i-1}} \tag{11}$$

where m_i and m_{i-1} denote the probability values after the adjustments. Based on the ranking of computed criticality scores, we select the top_k steps with the highest scores as the identified critical steps, where K is adaptively determined according to the total number of steps in each stage.

4.4 ONLINE STEP OPTIMIZATION

As illustrated in Figure 2, we construct paired training samples for reinforcement learning. Each pair consists of a truncated input sequence and its corresponding target, where critical steps identified during offline distillation are explicitly marked by K. This formulation enables the model to receive supervision not only from the final answer but also from intermediate reasoning checkpoints, thereby emphasizing the contribution of critical steps during optimization:

$$(q + s_1 \cdots + s_5 \;\Rightarrow\; A),\; (q + s_1 \cdots s_3 \;\Rightarrow\; A + s_5 + s_{k_4}),\; (q + s_1 \;\Rightarrow\; A + s_5 + \cdots s_{k_2}) \tag{12}$$

To encourage the model to derive correct answers by accurately reasoning through critical steps, we incorporate a reward component that explicitly accounts for step-wise correctness. Inspired by the notion of gated control, the reward for answer correctness is further modulated by the step score: if the step score is low (i.e., the reasoning steps are incorrect), the overall reward is substantially attenuated even when the final answer is correct. This mechanism imposes a structured constraint on the reasoning trajectory, thereby enhancing both the reliability and interpretability of the model's reasoning process. The reward function is formally defined as follows, where α,β, and γ are used to maintain the balance among different rewards.

$$\text{score} = \Big[\underbrace{R_{\text{fmt}}}_{\text{format score}} \cdot\alpha + (1-\alpha)\cdot\big(\underbrace{R_{\text{stp\_acc}}}_{\text{step accuracy}} \cdot\beta + (1-\beta)\cdot \underbrace{R_{\text{stp}}}_{\text{step format}} \big)\Big] \cdot \Big[(1-\gamma) + \gamma\cdot \underbrace{R_{\text{acc}}}_{\text{answer accuracy}} \Big] \tag{13}$$

R_{fmt} denotes the format score (excluding step formatting), $R_{\text{stp\_acc}}$ measures the correctness of reasoning steps, R_{stp} evaluates step format, where s_i is the model-predicted step index and R_{acc} represents the correctness of the final answer. The online strategy design enables the model to internalize both the correctness of the final answer and the validity of intermediate reasoning steps, cultivating the ability to derive correct answers through correctly reasoning about the critical steps.

4.5 Offline–Online Iteration

As training progresses, the model's ability to identify critical steps becomes more accurate, yielding step selections that better align with the true pivotal nodes of the reasoning process. Importantly, the set of steps regarded as critical is not static: as the model improves, previously difficult steps may no longer be bottlenecks, while new decisive steps emerge as the primary contributors to reasoning success. This dynamic evolution provides increasingly precise and informative learning signals for subsequent optimization. This provides more precise and informative learning signals for subsequent optimization. For each epoch, critical steps within all samples are first identified through Offline Step Distillation. The data is then reorganized during the Online Step Optimization phase and used to update the model, marking the completion of one iteration. In the subsequent iteration, the updated model is applied to the next batch, where Offline Step Distillation and Online Step Optimization are repeated. After several epochs, steps that have consistently never been identified as critical are pruned from subsequent training, thereby reducing redundant computation and accelerating convergence without sacrificing performance. Through this iterative cycle, the model progressively improves both its capacity to recognize critical steps and its accuracy in reasoning. The iterative process proceeds until model performance reaches a stable point, at which point the reinforcement learning stage is deemed converged and training advances to the subsequent phase.

5 Experiments

5.1 Implementation Details

Evaluation. We adopt a hierarchical soft matching framework for both final answers and critical reasoning steps. For answers, we first normalize predictions and references with LaTeX processing and unit unification. They are compared using layered strategies, including exact, equality-based soft, numerical, and canonical containment matching. Multiple candidates are allowed only on the reference side to capture annotation diversity, while enumerated predictions are disallowed to prevent guessing. For reasoning steps, reference keywords are extracted with Qwen3-Max(Team, 2025b), and outputs are segmented into "# Step k:" blocks. A strict–soft matching scheme is applied, relying on LaTeX normalization for strict matching and falling back to the answer-level matcher otherwise. Enumeration suppression, dynamic length control, and step count limits further ensure precise and efficient detection of critical conclusions.

Datasets. We train on three multimodal mathematical benchmarks: GEO3K (Lu et al., 2021), GEOQA (Chen et al., 2022), and GEOs(Seo et al., 2015). In total, 44k samples are used for SFT-based cold start, and 5.1k samples are employed in the Offline–Online RL stages.

Training. All experiments are conducted on $8\times$ A800 GPUs. Training with 1k questions per stage for one epoch takes about 2 hours. We adopt `Qwen-2.5-vl-Instruct-7B` as the backbone. In the SFT stage, we train for 3 epochs with a batch size of 32 and learning rate $2\mathrm{e}{-6}$. In the RL stage, the learning rate is set to $1\mathrm{e}{-6}$, and we experiment with subsets of 2k samples trained for a single epoch. Reward weights are set to $\alpha = 0.3$, $\beta = 0.75$, and $\gamma = 0.15$.

5.2 Main Results

Table 1: Comparison across Various Benchmarks

Model	AIME 25	MathVista	MathVision	MathVerse	MM-Math	OlympiadBench	DynaMath	MathScape
Qwen-2.5-VL-7B(Bai et al., 2025)	10.7	68.2	25.4	47.9	34.1	22.3	-	27.3
InternVL3-8B(Zhu et al., 2025b)	13.3	71.6	29.3	39.8	12.2	12.0	15.9	-
R1-VL-7B(Zhang et al., 2025b)	16.7	74.3	28.2	52.2	24.6	10.3	-	47.3
Vision-R1-7B(Huang et al., 2025)	23.3	73.5	-	52.4	36.2	11.9	16.4	-
ThinkLite-VL-7B(Wang et al., 2025b)	9.5	75.1	32.3	50.7	30.4	16.7	19.8	29.0
MiniCPM-o-2.6-7B(Team, 2025a)	11.4	71.9	21.6	35.0	22.0	9.3	12.2	67.0
VL-COGITO-7B(Yuan et al., 2025b)	3.3	74.8	30.7	53.3	35.8	22.4	15.9	45.8
Mulberry-7B(Yao et al., 2024)	14.2	63.1	19.1	41.6	23.7	9.5	-	40.3
R1-Onevision-7B(Yang et al., 2025)	19.9	64.1	29.9	40.0	-	18.3	10.2	**71.5**
Ours (O^2-CritiCuRL-7B)	**25.6**	**78.3**	**33.1**	**55.2**	**38.9**	**25.3**	**22.7**	43.2

We evaluate O^2-CritiCuRL on several benchmarks, including the text-only AIME-25 and multi-modal datasets. As shown in Table 1, MathVista(Lu et al., 2024), MathVisionWang et al. (2024a), and MathVerse(Zhang et al., 2024) assess visual mathematical reasoning across charts, diagrams, and natural images. MM-Math(Sun et al., 2024) covers open-ended middle-school problems, OlympiadBench(He et al., 2024) and DynaMathZou et al. (2025) focus on advanced and dynamic reasoning, and MathScape(Liang et al., 2025) provides $1,369$ real-world math photos. O^2-CritiCuRL achieves state-of-the-art performance across most benchmarks, while achieves relatively lower score due to its hand-captured images. It scores 78.3 on MathVista, outperforming the prior best 7B model ThinkLite-VL(Wang et al., 2025b) by 3.2 points, surpasses VL-COGITO(Yuan et al., 2025b) by 1.9 points on MathVerse, and improves upon ThinkLite-VL by 0.8 points on MathVision. O^2-CritiCuRL also sets a new standard on AIME-25 25.6, highlighting its strong generalization across both multimodal and text-only reasoning tasks.

Table 2: Different Design for Reward Functions

Reward Function	MathVista	MathVision	MathVerse
Add	77.5	33.1	54.8
Multiply	**78.3**	**33.5**	**55.2**

Table 3: Different Training Stage Performance

Training Stages	MathVista	MathVision	MathVerse
SFT	67.0	25.2	45.9
RL-Stage1	72.4	27.3	49.5
RL-Stage2	76.2	29.6	53.7
RL-Stage3	**78.3**	**33.1**	**55.2**

5.3 Ablation Study of Reward and Training Stages

We evaluate both our reward design and training stages on three benchmarks (MathVista, MathVision, MathVerse). As shown in Table 2, the multiplicative gating reward consistently outperforms the additive formulation (average $+0.53$), as it enforces stronger dependencies between step correctness and final answer accuracy, thereby discouraging degenerate guessing. Table 3 further shows steady gains across successive stages: RL-Stage1 improves over the SFT baseline, RL-Stage2 yields the largest boost by consolidating step correctness, and RL-Stage3 provides additional robustness with expert-level data. Together, these results verify that our step-aware curriculum and reward design complement each other, progressively enhancing both reasoning quality and final accuracy.

5.4 Ablation Study of Reward Function and Inference Speed

We first validate the effectiveness of our design choices through ablation studies. As shown in Table 4, our proposed metric K_i surpasses two intuitive baselines (e,g, absolute change Δp and relative improvement) by $3-7$ points across MathVista, MathVision, and MathVerse, confirming its advantage in capturing step importance. Table 5 shows that incorporating step accuracy (stepAcc)

Table 4: Metrics for Critical Step

metric for critical step	MathVista	MathVision	MathVerse
Δp	71.8	28.5	47.7
$\frac{p_i - p_{i-1}}{1 - p_{i-1}}$	74.3	30.1	49.3
K_i	**78.3**	**33.5**	**55.2**

Table 5: Step Accuracy Ablation

	MathVista	MathVision	MathVerse
w/o stepAcc	74.2	31.6	52.0
w/ stepAcc	**78.3**	**33.5**	**55.2**

as a reward yields consistent gains of $3 - 4$ points over answer-only supervision, demonstrating the benefit of rewarding correct intermediate reasoning steps for accuracy. In addition to accuracy, O^2-CritCuRL also improves efficiency (Table 6): on MM-Math it is 2× faster than R1-VL-7B and over 4× faster than Vision-R1-7B, while on MathScape it reduces inference time to 37min compared to 59min and 3h14min. These results highlight that O^2-CritCuRL not only enhances reasoning quality but also achieves superior efficiency, making it well-suited for practical deployment.

6 Case Study

Figure 3 presents a representative example to illustrate the reasoning ability of our O^2-CritiCuRL. In this geometry problem, the model is required to reason over multiple angle relations involving both bisection and proportional constraints. As shown in the step-by-step reasoning process, our model successfully identified and executed the critical reasoning steps (Steps 3 and 4), it derived the relation $\angle BON = 2\angle NOC$, correctly transformed it into the proportional form $\angle NOC = \frac{1}{3}\angle BOC$ and consequently obtained the intermediate value $t = 67.5°$. Building on this accurate intermediate conclusion, the model then proceeded to compute $\angle AOM = 45°$, which matches the ground-truth answer. This case demonstrates that O^2-CritiCuRL reaches correct answer through faithful and logically consistent reasoning steps.

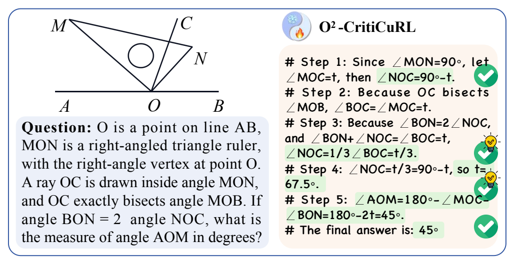


Figure 3: **Case Study.**

7 Conclusion

In this work, we propose O^2-CritiCuRL, an offline–online curriculum reinforcement learning framework that emphasizes critical-step awareness. By adaptively identifying decisive steps and iteratively refining them through offline decomposition and online reinforcement learning, our method improves accuracy, interpretability, and efficiency over conventional step-level RL. Experiments across multiple reasoning benchmarks demonstrate state-of-the-art performance. Looking ahead, we envision that the principle of critical-step awareness is not limited to mathematical reasoning, but can be broadly applied to domains that demand transparent and trustworthy decision-making, such as scientific discovery, legal analysis, and medical diagnosis. Future work will explore integrating human feedback into step identification, scaling to larger multimodal datasets, and designing adaptive curricula that co-evolve with model capability, further bridging the gap between human-like reasoning and practical deployment in real-world scenarios.

Table 6: Inference Speed of different models on MM-Math and MathScape datasets.

Model	MM-Math (1k-subset)	MathScape (0.4k-subset)
Vision-R1-7B	4h39min	3h14min
R1-VL-7B	2h11min	59min
Ours (O^2-CritiCuRL-7B)	1h03min	37min

References

Amodei, Dario, Olah, Chris, Steinhardt, Jacob, Christiano, Paul, Schulman, John, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

Arora, Daman, Zanette, and Andrea. Training language models to reason efficiently. *CoRR*, 2025.

Arrieta, Alejandro Barredo, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information fusion*, 58:82–115, 2020.

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. URL `https://arxiv.org/abs/2502.13923`.

Natalie C Benda, Laurie L Novak, Carrie Reale, and Jessica S Ancker. Trust in ai: why we should be designing for appropriate reliance. *Journal of the American Medical Informatics Association*, 29(1):207–212, 2022.

Bengio, Yoshua, Louradour, Jérôme, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48, 2009.

Jiaqi Chen, Jianheng Tang, Jinghui Qin, Xiaodan Liang, Lingbo Liu, Eric P. Xing, and Liang Lin. Geoqa: A geometric question answering benchmark towards multimodal numerical reasoning, 2022. URL `https://arxiv.org/abs/2105.14517`.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *CoRR*, 2025.

Xiaoxue Cheng, Junyi Li, Wayne Xin Zhao, and Ji-Rong Wen. Chainlm: Empowering large language models with improved chain-of-thought prompting. *CoRR*, 2024.

Comanici, Gheorghe, Bieber, Eric, Schaekermann, Mike, Pasupat, Ice, Sachdeva, Noveen, Dhillon, Inderjit, Blistein, Marcel, Ram, Ori, Zhang, Dan, Rosen, Evan, 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.

Deng, Huilin, Zou, Ding, Ma, Rui, Luo, Hongchen, Cao, Yang, Kang, and Yu. Boosting the generalization and reasoning of vision language models with curriculum reinforcement learning. *arXiv preprint arXiv:2503.07065*, 2025.

El-Kishky, Ahmed, Wei, Alexander, Saraiva, Andre, Minaiev, Borys, Selsam, Daniel, Dohan, David, Song, Francis, Lightman, Hunter, Gilaberte, Ignasi Clavera, Pachocki, Jakub, et al. Competitive programming with large reasoning models. *CoRR*, 2025.

Guo, Daya, Zhu, Qihao, Yang, Dejian, Xie, Zhenda, Dong, Kai, Zhang, Wentao, Chen, Guanting, Bi, Xiao, Wu, Y, Li, YK, et al. Deepseek-coder: When the large language model meets programming-the rise of code intelligence. *CoRR*, 2024.

Guo, Daya, Yang, Dejian, Zhang, Haowei, Song, Junxiao, Zhang, Ruoyu, Xu, Runxin, Zhu, Qihao, Ma, Shirong, Wang, Peiyi, Bi, Xiao, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Yifan Hao, Fangning Chao, Yaqian Hao, Zhaojun Cui, Huan Bai, Haiyu Zhang, Yankai Liu, Chao Deng, and Junlan Feng. Jt-math: A multi-stage framework for advanced mathematical reasoning in large language models. *arXiv preprint arXiv:2507.19748*, 2025.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems, 2024. URL `https://arxiv.org/abs/2402.14008`.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset, 2021. URL https://arxiv.org/abs/2103.03874.

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, 2025. URL https://arxiv.org/abs/2503.06749.

Joongwon Kim, Anirudh Goyal, Liang Tan, Hannaneh Hajishirzi, Srini Iyer, and Tianlu Wang. Astro: Teaching language models to reason by reflecting and backtracking in-context. *arXiv preprint arXiv:2507.00417*, 2025.

Xuanyu Lei, Chenliang Li, Yuning Wu, Kaiming Liu, Weizhou Shen, Peng Li, Ming Yan, Ji Zhang, Fei Huang, and Yang Liu. Writing-rl: Advancing long-form writing via adaptive curriculum reinforcement learning. *arXiv preprint arXiv:2506.05760*, 2025.

Hao Liang, Linzhuang Sun, Minxuan Zhou, Zirong Chen, Meiyi Qiang, Mingan Lin, Tianpeng Li, Fan Yang, Zenan Zhou, and Wentao Zhang. Mathscape: Benchmarking multimodal large language models in real-world mathematical contexts, 2025. URL https://arxiv.org/abs/2408.07543.

Lightman, Hunter, Kosaraju, Vineet, Burda, Yuri, Edwards, Harrison, Baker, Bowen, Lee, Teddy, Leike, Jan, Schulman, John, Sutskever, Ilya, Cobbe, and Karl. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023a.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step, 2023b. URL https://arxiv.org/abs/2305.20050.

Ling, Zhan, Fang, Yunhao, Li, Xuanlin, Huang, Zhiao, Lee, Mingu, Memisevic, Roland, Su, and Hao. Deductive verification of chain-of-thought reasoning. *Advances in Neural Information Processing Systems*, 36:36407–36433, 2023.

Zeyu Liu, Yuhang Liu, Guanghao Zhu, Congkai Xie, Zhen Li, Jianbo Yuan, Xinyao Wang, Qing Li, Shing-Chi Cheung, Shengyu Zhang, Fei Wu, and Hongxia Yang. Infi-mmr: Curriculum-based unlocking multimodal reasoning via phased reinforcement learning in multimodal small language models. *arXiv preprint arXiv:2505.23091*, 2025a.

Ziru Liu, Cheng Gong, Xinyu Fu, Yaofang Liu, Ran Chen, Shoubo Hu, Suiyun Zhang, Rui Liu, Qingfu Zhang, and Dandan Tu. Ghpo: Adaptive guidance for stable and efficient llm reinforcement learning. *arXiv preprint arXiv:2507.10628*, 2025b.

Lu, Jinghui, Yu, Haiyang, Xu, Siliang, Ran, Shiwei, Tang, Guozhi, Wang, Siqi, Shan, Bin, Fu, Teng, Feng, Hao, Tang, Jingqun, et al. Prolonged reasoning is not all you need: Certainty-based adaptive routing for efficient llm/mllm reasoning. *arXiv preprint arXiv:2505.15154*, 2025.

Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. Inter-gps: Interpretable geometry problem solving with formal language and symbolic reasoning, 2021. URL https://arxiv.org/abs/2105.04165.

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, 2024. URL https://arxiv.org/abs/2310.02255.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, et al. Improve mathematical reasoning in language models by automated process supervision. *CoRR*, 2024.

Shubham Parashar, Shurui Gui, Xiner Li, Hongyi Ling, Sushil Vemuri, Blake Olson, Eric Li, Yu Zhang, James Caverlee, Dileep Kalathil, and Shuiwang Ji. Curriculum reinforcement learning from easy to hard tasks improves llm reasoning. *arXiv preprint arXiv:2506.06632*, 2025.

Poursabzi-Sangdeh, Forough, Goldstein, Daniel G, Hofman, Jake M, Wortman Vaughan, Jennifer Wortman, Wallach, and Hanna. Manipulating and measuring model interpretability. In *Proceedings of the 2021 CHI conference on human factors in computing systems*, pp. 1–52, 2021.

Qu, Xiaoye, Li, Yafu, Su, Zhaochen, Sun, Weigao, Yan, Jianhao, Liu, Dongrui, Cui, Ganqu, Liu, Daizong, Liang, Shuxian, He, Junxian, et al. A survey of efficient reasoning for large reasoning models: Language, multimodality, and beyond. *arXiv preprint arXiv:2503.21614*, 2025.

Min Joon Seo, Hannaneh Hajishirzi, Ali Farhadi, Oren Etzioni, and Clint Malcolm. Solving geometry problems: Combining text and diagram interpretation. In *EMNLP*, volume 1, pp. 3, 2015.

Shakya, Ashish Kumar, Pillai, Gopinatha, Chakrabarty, and Sohom. Reinforcement learning algorithms: A brief survey. *Expert Systems with Applications*, 231:120495, 2023.

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

Shuaijie She, Junxiao Liu, Yifeng Liu, Jiajun Chen, Xin Huang, and Shujian Huang. R-prm: Reasoning-driven process reward modeling. *arXiv preprint arXiv:2503.21295*, 2025.

Haozhan Shen, Peng Liu, Jingcheng Li, Chunxin Fang, Yibo Ma, Jiajia Liao, Qiaoli Shen, Zilun Zhang, Kangjia Zhao, Qianqian Zhang, Ruochen Xu, and Tiancheng Zhao. VLM-R1: A stable and generalizable R1-style large vision-language model. *arXiv preprint arXiv:2504.07615*, 2025.

Skalse, Joar Max Viktor, Howe, Nikolaus HR, Krasheninnikov, Dmitrii, Krueger, and David. Defining and characterizing reward gaming. In *Advances in Neural Information Processing Systems*, 2022.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. Learning to summarize from human feedback, 2022. URL https://arxiv.org/abs/2009.01325.

Stiglic, Gregor, Kocbek, Primoz, Fijacko, Nino, Zitnik, Marinka, Verbert, Katrien, Cilar, and Leona. Interpretability of machine learning-based prediction models in healthcare. *Wiley Interdisciplinary Reviews-Data Mining And Knowledge Discovery*, 10(5), 2020.

Kai Sun, Yushi Bai, Ji Qi, Lei Hou, and Juanzi Li. Mm-math: Advancing multimodal math evaluation with process evaluation and fine-grained classification, 2024. URL https://arxiv.org/abs/2404.05091.

OpenBMB Team. Minicpm-o 2.6: A gpt-4o-level mllm for vision, speech, and multimodal live streaming on your phone. https://github.com/OpenBMB/MiniCPM-o, 2025a. Online; OpenBMB Notion Page.

Qwen Team. Qwen3-max: Just scale it, September 2025b.

Wang, Hanchen, Fu, Tianfan, Du, Yuanqi, Gao, Wenhao, Huang, Kexin, Liu, Ziming, Chandak, Payal, Liu, Shengchao, Van Katwyk, Peter, Deac, Andreea, et al. Scientific discovery in the age of artificial intelligence. *Nature*, 620(7972):47–60, 2023.

Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with math-vision dataset, 2024a. URL https://arxiv.org/abs/2402.14804.

Peiyi Wang, Lei Li, Zhihong Shao, R. X. Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations, 2024b. URL https://arxiv.org/abs/2312.08935.

Weiyun Wang, Zhangwei Gao, Lianjie Chen, Zhe Chen, Jinguo Zhu, Xiangyu Zhao, Yangzhou Liu, Yue Cao, Shenglong Ye, Xizhou Zhu, Lewei Lu, Haodong Duan, Yu Qiao, Jifeng Dai, and Wenhai Wang. Visualprm: An effective process reward model for multimodal reasoning. *arXiv preprint arXiv:2503.10291*, 2025a.

Xiyao Wang, Linfeng Song, Ye Tian, Dian Yu, Baolin Peng, Haitao Mi, Furong Huang, and Dong Yu. Towards self-improvement of llms via mcts: Leveraging stepwise knowledge with curriculum preference learning. *arXiv preprint arXiv:2410.06508*, 2024c.

Xiyao Wang, Zhengyuan Yang, Chao Feng, Hongjin Lu, Linjie Li, Chung-Ching Lin, Kevin Lin, Furong Huang, and Lijuan Wang. Sota with less: Mcts-guided sample selection for data-efficient visual reasoning self-improvement, 2025b. URL https://arxiv.org/abs/2504.07934.

Zhenting Wang, Guofeng Cui, Yu-Jhe Li, Kun Wan, and Wentian Zhao. Dump: Automated distribution-level curriculum learning for rl-based llm post-training. *arXiv preprint arXiv:2504.09710*, 2025c.

Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, Haosheng Zou, Yongchao Deng, Shousheng Jia, and Xiangzheng Zhang. Light-r1: Curriculum sft, dpo and rl for long cot from scratch and beyond. *arXiv preprint arXiv:2503.10460*, 2025. URL https://github.com/Qihoo360/Light-R1.

Yi Yang, Xiaoxuan He, Hongkun Pan, Xiyan Jiang, Yan Deng, Xingtao Yang, Haoyu Lu, Dacheng Yin, Fengyun Rao, Minfeng Zhu, Bo Zhang, and Wei Chen. R1-onevision: Advancing generalized multimodal reasoning through cross-modal formalization, 2025. URL https://arxiv.org/abs/2503.10615.

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, 2024. URL https://arxiv.org/abs/2412.18319.

Wenwen Yu, Zhibo Yang, Yuliang Liu, and Xiang Bai. Docthinker: Explainable multimodal large language models with rule-based reinforcement learning for document understanding. *arXiv preprint arXiv:2508.08589*, 2025.

Yuan, Ruifeng, Xiao, Chenghao, Leng, Sicong, Wang, Jianyu, Li, Long, Xu, Weiwen, Chan, Hou Pong, Zhao, Deli, Xu, Tingyang, Wei, Zhongyu, et al. Vl-cogito: Progressive curriculum reinforcement learning for advanced multimodal reasoning. *arXiv preprint arXiv:2507.22607*, 2025a.

Ruifeng Yuan, Chenghao Xiao, Sicong Leng, Jianyu Wang, Long Li, Weiwen Xu, Hou Pong Chan, Deli Zhao, Tingyang Xu, Zhongyu Wei, Hao Zhang, and Yu Rong. Vl-cogito: Progressive curriculum reinforcement learning for advanced multimodal reasoning, 2025b. URL https://arxiv.org/abs/2507.22607.

Zeng, Zhiyuan, Cheng, Qinyuan, Yin, Zhangyue, Wang, Bo, Li, Shimin, Zhou, Yunhua, Guo, Qipeng, Huang, Xuanjing, Qiu, and Xipeng. Scaling of search and learning: A roadmap to reproduce o1 from reinforcement learning perspective. *CoRR*, 2024.

Thomas Zeng, Shuibai Zhang, Shutong Wu, Christian Classen, Daewon Chae, Ethan Ewer, Minjae Lee, Heeju Kim, Wonjun Kang, Jackson Kunde, Ying Fan, Jungtaek Kim, Hyung Il Koo, Kannan Ramchandran, Dimitris Papailiopoulos, and Kangwook Lee. Versaprm: Multi-domain process reward model via synthetic reasoning data. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

Yuheng Zha, Kun Zhou, Yujia Wu, Yushu Wang, Jie Feng, Zhi Xu, Shibo Hao, Zhengzhong Liu, Eric P. Xing, and Zhiting Hu. Vision-g1: Towards general vision language reasoning with multi-domain data curation. *arXiv preprint arXiv:2508.12680*, 2025. URL https://github.com/yuh-zha/Vision-G1.

Jingyi Zhang, Jiaxing Huang, Huanjin Yao, Shunyu Liu, Xikun Zhang, Shijian Lu, and Dacheng Tao. R1-vl: Learning to reason with multimodal large language models via step-wise group relative policy optimization. *arXiv preprint arXiv:2503.12937*, 2025a.

Jingyi Zhang, Jiaxing Huang, Huanjin Yao, Shunyu Liu, Xikun Zhang, Shijian Lu, and Dacheng Tao. R1-vl: Learning to reason with multimodal large language models via step-wise group relative policy optimization, 2025b. URL https://arxiv.org/abs/2503.12937.

Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Peng Gao, and Hongsheng Li. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems?, 2024. URL `https://arxiv.org/abs/2403.14624`.

Chenglin Zhu, Tao Zhang, Chong Li, Mingan Lin, Zenan Zhou, and Jian Xie. Eduflow: Advancing mllms' problem-solving proficiency through multi-stage, multi-perspective critique. *arXiv preprint arXiv:2507.09374*, 2025a.

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, Zhangwei Gao, Erfei Cui, Xuehui Wang, Yue Cao, Yangzhou Liu, Xingguang Wei, Hongjie Zhang, Haomin Wang, Weiye Xu, Hao Li, Jiahao Wang, Nianchen Deng, Songze Li, Yinan He, Tan Jiang, Jiapeng Luo, Yi Wang, Conghui He, Botian Shi, Xingcheng Zhang, Wenqi Shao, Junjun He, Yingtong Xiong, Wenwen Qu, Peng Sun, Penglong Jiao, Han Lv, Lijun Wu, Kaipeng Zhang, Huipeng Deng, Jiaye Ge, Kai Chen, Limin Wang, Min Dou, Lewei Lu, Xizhou Zhu, Tong Lu, Dahua Lin, Yu Qiao, Jifeng Dai, and Wenhai Wang. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models, 2025b. URL `https://arxiv.org/abs/2504.10479`.

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, 2025. URL `https://arxiv.org/abs/2411.00836`.

8 Appendix

8.1 ETHICS STATEMENT

This work adheres to the ICLR Code of Ethics. In this study, no human subjects or animal experimentation was involved. All datasets used were sourced in compliance with relevant usage guidelines, ensuring no violation of privacy. We have taken care to avoid any biases or discriminatory outcomes in our research process. No personally identifiable information was used, and no experiments were conducted that could raise privacy or security concerns. We are committed to maintaining transparency and integrity throughout the research process.

8.2 REPRODUCIBILITY STATEMENT

We have made every effort to ensure that the results presented in this paper are reproducible. All code and datasets have been made publicly available in an anonymous repository to facilitate replication and verification. The experimental setup, including training steps, model configurations, and hardware details, is described in detail in the paper, We have also provided a full description of experiments, to assist others in reproducing our experiments. Additionally, datasets, such as MathVista(Lu et al., 2024), MathVisionWang et al. (2024a), are publicly available, ensuring consistent and reproducible evaluation results. We believe these measures will enable other researchers to reproduce our work and further advance the field.

8.3 LLM Usage

Large Language Models (LLMs) were used to aid in the writing and polishing of the manuscript. Specifically, we used an LLM to assist in refining the language, improving readability, and ensuring clarity in various sections of the paper. The model helped with tasks such as sentence rephrasing,grammar checking, and enhancing the overall flow of the text. It is important to note that the LLM was not involved in the ideation, research methodology, or experimental design. All research concepts, ideas, and analyses were developed and conducted by the authors. The contributions of the LLM were solely focused on improving the linguistic quality of the paper, with no involvement in the scientific content or data analysis. The authors take full responsibility for the content of the manuscript, including any text generated or polished by the LLM. We have ensured that the LLM-generated text adheres to ethical guidelines and does not contribute to plagiarism or scientific misconduct.

Theoretical Foundation and Interpretation

This section provides the theoretical foundation, detailing the derivation and practical implementation of the criticality metric $\mathcal{K}_i$. In our approach, the criticality of a reasoning step is assessed based on the change in the model's accuracy after sequentially introducing each step. Assuming each step is correct, adding a step will not decrease the model's accuracy; it can either improve it or leave it unchanged (if the step provides no additional information).

Introducing a critical step inevitably causes a significant shift in the answer probability distribution: the probability of the correct answer increases, while the probabilities of other answers decrease. To describe this process in a simple yet principled way, we introduce the **uniform compression assumption**:

1. When the model has no reasoning ability, all answers are assigned approximately equal initial probabilities; if the answer space is very large, these probabilities naturally approach zero.
2. When a critical step is introduced, the correct answer probability increases, while the remaining probability mass of other answers is assumed, in an idealized sense, to be uniformly redistributed. This assumption provides a tractable and reasonable approximation of the probability shift.

To quantify the difference between the prior and posterior distributions induced by a reasoning step, we use the **Kullback-Leibler (KL) divergence**, defined as

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{\omega \in \Omega} P(\omega) \log \frac{P(\omega)}{Q(\omega)}. \tag{14}$$

Its fundamental interpretation is the additional information required to represent the target distribution P using a code optimized for the distribution Q. In our context, the KL divergence measures the extra information needed for the current distribution to align with a distribution that successfully leads to the correct answer.

Within this framework, the criticality of step S_i is quantified as the KL divergence between the posterior P_i and the prior P_{i-1}:

$$\mathcal{K}_i = D_{\mathrm{KL}}(P_i \parallel P_{i-1}). \tag{15}$$

Intuitively:

- **Large** $\mathcal{K}_i$: Indicates a substantial difference between P_i and P_{i-1}, meaning the step S_i provides significant new information, which is highly critical for correcting the model when it was previously far from the correct answer.
- **Small** $\mathcal{K}_i$: Indicates that P_i is similar to P_{i-1}, meaning the step S_i offers only minor refinements, acting more like a fine-tuning adjustment when the model is already close to the correct answer.

Derivation of the Computable Metric

The definition of $\mathcal{K}_i$ is general but computationally complex as it requires a sum over the entire answer space Ω. To derive a tractable metric, we introduce a structural assumption about how reasoning steps affect the probability distribution.

We begin by decomposing the sum into the contribution from the correct answer ω^* and the space of all incorrect answers:

$$\mathcal{K}_i = P_i(\omega^*) \log \frac{P_i(\omega^*)}{P_{i-1}(\omega^*)} + \sum_{\omega \neq \omega^*} P_i(\omega) \log \frac{P_i(\omega)}{P_{i-1}(\omega)}. \tag{16}$$

Let $p_i = P_i(\omega^*)$ denote the probability assigned to the correct answer at step i. Consequently, the total probability mass of all incorrect answers is $1 - p_i = \sum_{\omega \neq \omega^*} P_i(\omega)$.

Assumption (Uniform Compression). We assume that the effect of a reasoning step on the probability of all incorrect answers is uniform. Formally, the probability ratio is identical for every incorrect answer $\omega \neq \omega^*$:

$$\forall \omega \neq \omega^*, \quad \frac{P_i(\omega)}{P_{i-1}(\omega)} = \frac{1-p_i}{1-p_{i-1}}. \tag{17}$$

This assumption implies that the reasoning step homogeneously compresses the entire incorrect answer space, redistributing the probability mass toward the correct answer.

Under this assumption, the second term simplifies as follows:

$$\sum_{\omega \neq \omega^*} P_i(\omega) \log \frac{P_i(\omega)}{P_{i-1}(\omega)} = (1-p_i) \log \frac{1-p_i}{1-p_{i-1}}. \tag{18}$$

The first term remains unchanged:

$$p_i \log \frac{p_i}{p_{i-1}}. \tag{19}$$

Combining these contributions yields our final computable metric for step criticality:

$$\boxed{\mathcal{K}_i = p_i \log \frac{p_i}{p_{i-1}} + (1-p_i) \log \frac{1-p_i}{1-p_{i-1}}}. \tag{20}$$

BOUNDARY BEHAVIOR, ROBUSTNESS, AND SMOOTHING

The metric $\mathcal{K}_i$ exhibits extreme and theoretically justified behavior at the boundaries of the probability space, which necessitates careful handling in practice.

Boundary Analysis:

Case 1: $p_{i-1} \to 0^+$ Consider

$$\mathcal{K}_i = p_i \log \frac{p_i}{p_{i-1}} + (1-p_i) \log \frac{1-p_i}{1-p_{i-1}}.$$

As $p_{i-1} \to 0^+$, the second term behaves as

$$(1-p_i) \log \frac{1-p_i}{1-p_{i-1}} = (1-p_i) \log \left(1 + \frac{p_{i-1}-p_i}{1-p_{i-1}}\right) \to (1-p_i) \log(1-p_i),$$

which remains finite. The first term behaves as

$$p_i \log \frac{p_i}{p_{i-1}} = p_i(\log p_i - \log p_{i-1}) \sim -p_i \log p_{i-1} \to +\infty,$$

since $-\log p_{i-1} \to +\infty$. Therefore, $\mathcal{K}_i \to +\infty$, reflecting an infinite information gain when updating from near-zero prior probability to a finite posterior.

Case 2: $p_{i-1} \to 1^-$ Let $q_i = 1 - p_i$ and $q_{i-1} = 1 - p_{i-1}$. Then

$$\mathcal{K}_i = (1-q_i) \log \frac{1-q_i}{1-q_{i-1}} + q_i \log \frac{q_i}{q_{i-1}}.$$

As $q_{i-1} \to 0^+$, the first term behaves as

$$(1-q_i) \log \frac{1-q_i}{1-q_{i-1}} \to \log 1 = 0,$$

and the second term

$$q_i \log \frac{q_i}{q_{i-1}} = q_i(\log q_i - \log q_{i-1}) \sim -q_i \log q_{i-1} \to 0,$$

because $q_i \to 0$ along with q_{i-1}. Hence $\mathcal{K}_i \to 0$, indicating negligible information gain when the model is already nearly certain.

Robustness Property via Taylor Expansion:

Let $\Delta p = p_i - p_{i-1}$. Expanding $\mathcal{K}_i$ around $p_i = p_{i-1}$ using $\log(1+x) \approx x - \frac{x^2}{2} + \mathcal{O}(x^3)$:

$$\begin{aligned}
\mathcal{K}_i &= p_{i-1} \log \frac{p_{i-1} + \Delta p}{p_{i-1}} + (1 - p_{i-1}) \log \frac{1 - p_{i-1} - \Delta p}{1 - p_{i-1}} \\
&= p_{i-1} \log \left(1 + \frac{\Delta p}{p_{i-1}}\right) + (1 - p_{i-1}) \log \left(1 - \frac{\Delta p}{1 - p_{i-1}}\right) \\
&\approx p_{i-1} \left(\frac{\Delta p}{p_{i-1}} - \frac{(\Delta p)^2}{2p_{i-1}^2}\right) + (1 - p_{i-1}) \left(-\frac{\Delta p}{1 - p_{i-1}} - \frac{(\Delta p)^2}{2(1 - p_{i-1})^2}\right) \\
&= \Delta p - \frac{(\Delta p)^2}{2p_{i-1}} - \Delta p - \frac{(\Delta p)^2}{2(1 - p_{i-1})} \\
&= \frac{(\Delta p)^2}{2} \left(\frac{1}{1 - p_{i-1}} + \frac{1}{p_{i-1}}\right) \\
&= \frac{(\Delta p)^2}{2p_{i-1}(1 - p_{i-1})}.
\end{aligned}$$

Summing both contributions:

$$\mathcal{K}_i \approx \frac{(\Delta p)^2}{2} \left(\frac{1}{1 - p_{i-1}} + \frac{1}{p_{i-1}}\right) = \frac{(\Delta p)^2}{2p_{i-1}(1 - p_{i-1})}.$$

This shows that for small Δp, $\mathcal{K}_i$ is of order $(\Delta p)^2$, making it inherently robust to small fluctuations in the probability estimates.

Smoothing Technique: To ensure numerical stability, smoothing is applied at both ends of the probability range. At the lower end, if p_i or p_{i-1} equals 0, this is effectively a numerical error since the model's initial reasoning ability cannot be exactly zero. To avoid excessively large KL divergence values in practice, these probabilities are replaced by a small positive constant ϵ, e.g., $\epsilon = 0.1$, which is much smaller than the minimal sampling probability precision.

At the upper end, if p_i or p_{i-1} equals 1, although the result does not theoretically diverge, computing $\log 0$ could occur during calculations, which leads to numerical instability. Therefore, these probabilities are also capped below $1 - \epsilon$ to ensure stability:

$$\begin{aligned}
\tilde{p}_i &= \min\Big(\max(p_i, \epsilon),\, 1 - \epsilon\Big), \\
\tilde{p}_{i-1} &= \min\Big(\max(p_{i-1}, \epsilon),\, 1 - \epsilon\Big), \\
&\epsilon \ll \text{minimal sampling probability}.
\end{aligned} \tag{21}$$

The smoothed criticality metric is then computed as:

$$\mathcal{K}_i^{(\text{smooth})} = \tilde{p}_i \log \frac{\tilde{p}_i}{\tilde{p}_{i-1}} + (1 - \tilde{p}_i) \log \frac{1 - \tilde{p}_i}{1 - \tilde{p}_{i-1}}. \tag{22}$$

8.4 Reward Function

Our reward function is designed to guide the model towards three key objectives simultaneously:

1. Correctly output the final answer.
2. Perform step-wise, logical reasoning.
3. Recognize whether the reasoning process is complete: if complete, directly output the answer; if incomplete, correctly continue the reasoning steps.

The entire training process is divided into two distinct phases, each with its own reward function. They work in tandem to achieve our ultimate goal.

Offline Rollout Phase: This phase does not involve model training. Instead, it serves as an analytical tool to identify and evaluate critical steps within a Chain-of-Thought (CoT) demonstration. By using the reward function in this stage, we can prune redundant or less essential steps, creating a more concise and efficient training dataset.

Online Update Phase: This is the active model training phase. We use the refined dataset from the offline stage and provide real-time, granular feedback with the online reward function. This guides the model to learn and execute a correct reasoning process. This two-stage approach ensures that we first obtain high-quality training data and then use that data effectively to train the model.

8.4.1 OFFLINE ROLLOUT REWARD FUNCTION

This phase is focused on evaluating the informational value of each step in a demonstration. The reward function here helps in identifying which steps are most crucial for reaching the correct final answer.

Answer Accuracy: $R_{\textbf{acc}}$ This metric provides a foundational measure of the model's ability to produce the correct final answer after a given number of reasoning steps.

- For each input sample, we execute multiple simulated ”rollouts.” In each rollout, we extract the final answer (the text following `# The final answer is:`) and compare it to the ground truth.
  - If the answer is correct, we assign a reward of $R_{\text{acc}} = 1$.
  - If the answer is incorrect, the reward is $R_{\text{acc}} = 0$.
- The mean correctness probability, p_i, is then calculated by averaging the R_{acc} values over a series of n rollouts. This is performed after step i has been provided to the model, meaning the model has access to all steps from 1 to i.

  $$p_i = \frac{1}{n}\sum_{j=1}^{n} R_{\text{acc}}^{(j)}$$

  Here, p_i represents the mean probability that the model can correctly solve the problem after having access to the first i reasoning steps.

Key Step Metric K_i The Key Step Metric, K_i, is the core of our offline analysis. It quantifies the informational value of a specific step by measuring how significantly it alters the model's probability of producing the correct final answer.

- We compute the Kullback-Leibler (KL) divergence between the correctness probabilities of consecutive steps, p_i and p_{i-1}. A large K_i value signifies that the step provides crucial new information that dramatically improves the chances of getting the final answer correct.

  $$K_i = p_i \log \frac{p_i}{p_{i-1}} + (1 - p_i) \log \frac{1 - p_i}{1 - p_{i-1}}$$

  Conversely, a small K_i indicates the step is less essential or even redundant.

8.4.2 ONLINE UPDATE REWARD FUNCTION

This phase involves granular, real-time feedback to guide the model towards producing a correct and well-formatted reasoning process.

Format Score: $R_{\textbf{fmt}}$ This score ensures that the model's output strictly adheres to a predefined format, which is essential for consistent and automated evaluation throughout the training process.

- If the model output strictly follows the specified format (`# Step # The final answer is: ...\n`), $R_{\text{fmt}} = 1$.
- Any deviation from the required format results in $R_{\text{fmt}} = 0$.

Step Format: R_{stp} This component evaluates whether the model's predicted step number aligns with the expected sequential order and whether it correctly decides to stop or continue the process. We use a variable, `judging_step`, to denote the expected next step number that the model should output.

- When reasoning is incomplete (`judging_step` $\neq$ `0`):
  - Correct Continuation: If the model's output begins with `# Step judging_step`, it receives the highest reward, $R_{\text{stp}} = 1$.
  - Incorrect Step Number: If the model's output begins with `# Step pred_step` where `pred_step` is not equal to `judging_step`, the reward is calculated using a decay coefficient k: $R_{\text{stp}} = \max(0, 1 - k \cdot |\text{pred\_step} - \text{judging\_step}|)$. The larger the deviation, the smaller the reward.
  - Early Answer Output: If the model outputs the final answer before the reasoning is complete, the reward is $R_{\text{stp}} = 0$.
- When reasoning is complete (`judging_step = 0`, indicating the model has reached the final step from the reference CoT):
  - Correct Termination: If the model correctly outputs the final answer, it receives the highest reward, $R_{\text{stp}} = 1$.
  - Redundant Step: If the model generates a superfluous step after the reasoning is complete, it receives a small, token reward of $R_{\text{stp}} = 0.05$. This encourages early answer output and reduces redundancy.
  - Other Incorrect Output: Any other incorrect output results in $R_{\text{stp}} = 0$.

Step Accuracy: $R_{\text{stp\_acc}}$ This reward is crucial for ensuring the quality of the reasoning content itself.

- We check if the model's output for the current step contains a pre-extracted key string (`key_str`), which represents a crucial piece of information for that specific step.
- If the output contains the key string, $R_{\text{stp\_acc}} = 1.0$.
- Otherwise, $R_{\text{stp\_acc}} = 0.0$.

Final Reward Score (Multiplicative Gating) The total reward score is a composite of all components, orchestrated by a multiplicative gating mechanism. This structure ensures a holistic evaluation of the model's performance, balancing process quality with final correctness.

$$\text{score} = \Big[\underbrace{R_{\text{fmt}}}_{\text{format score}} \cdot \alpha + (1-\alpha) \cdot \big(\underbrace{R_{\text{stp\_acc}}}_{\text{step accuracy}} \cdot \beta + (1-\beta) \cdot \underbrace{R_{\text{stp}}}_{\text{step format}} \big) \Big] \cdot \Big[(1-\gamma) + \gamma \cdot \underbrace{R_{\text{acc}}}_{\text{answer accuracy}} \Big] \tag{23}$$

The formula is structured into two main parts connected by multiplication. The first part evaluates the quality of the reasoning process itself. The second part serves as a modulator based on the final answer accuracy. This multiplicative gating ensures that if the final answer is incorrect ($R_{\text{acc}} = 0$), the overall reward is significantly penalized, reinforcing that process quality, while vital, must ultimately lead to a correct solution.

Parameter Explanations

- α (Format Weight): Set to 0.3. This value is chosen to give a significant, but not overwhelming, weight to format correctness.
- β (Step Accuracy Weight): Set to 0.75. This high value deliberately emphasizes the paramount importance of content accuracy within each step.
- γ (Answer Accuracy Weight): Set to 0.15. This low weight is a critical design choice. It signals that while the final answer's correctness is important, the quality and integrity of the intermediate reasoning steps are given a higher priority during training.
- k (Step Decay Coefficient): Set to 0.25. This parameter controls the reward decay when the predicted step number deviates from the expected one, preventing excessive rewards for incorrect steps.

8.5 Prompt

8.5.1 Prompt for Dataset Step Extraction

Single-word/phrase answer

Outline the reasoning in the fewest necessary atomic steps, omitting any redundancy. If a step involves obtaining information from image, add '[image]' at the start, like '# Step x: [image]...'.
Next, check my answer-if correct, output it; otherwise give the correct one.
Strictly format your response with the following sections, separated by '#':
Question:
Step 1:
Step 2:
...
Answer:
Brief answer:"

question:
Is the given caption "The cake is below the teddy bear." fitting for the image?

reasoning:
The image shows a cake with a teddy bear on top. The teddy bear is positioned higher up on the cake, and the cake itself is underneath it. Therefore, the caption "The cake is below the teddy bear" accurately describes the image."

→

question:
Is the given caption "The cake is below the teddy bear." fitting for the image?
reasoning:
#Step 1:[image] Observe that the teddy bear is placed on top of the cake.
#Step 2:[image] Determine spatial relationship: the cake is directly underneath the teddy bear.
#Step 3: Evaluate the caption: "The cake is below the teddy bear" correctly describes the vertical positioning.
#Answer: Yes
#Brief answer: Yes

Single-option answer

First, keep the original meaning of the question unchanged, and rewrite it as an open-ended question that can be answered with a single word/phrase. If the question includes a context section, keep it unchanged and place it before the question.
Next, check my answer-if correct, output it directly or rewrite it as a sentence; otherwise give the correct one.
Finally, summarize the correct answer in a single word or phrase.
Strictly format your response with the following sections, separated by '#':
Question:
Step 1:
Step 2:
...
Answer:
Brief answer:

question:
what is holding the umbrella?\nChoices:\nA. Hand.\nB. A man.\nC. A girl.\nD. A table.\nAnswer with the option's letter from the given choices directly.

reasoning:
The image shows a person's hand extending from a train window, holding a blue umbrella. The hand's bounding box overlaps with the umbrella's, indicating direct contact. Other options don't fit the context.\n\nAnswer: A. Hand

question:
what is holding the umbrella?
reasoning:
#Step 1: [image] Observe a hand extending from a train window, gripping a blue umbrella.
#Step 2: [image] Confirm direct contact between the hand and the umbrella via overlapping bounding boxes.
#Step 3: Eliminate incorrect options: no full man or girl visible, and the umbrella is not attached to a table.
Step 4: Identify that the hand is the direct object holding the umbrella.
Answer: A. Hand
Brief answer: Hand

For the mathematical category of questions, we selected three subsets—geo3k, geoqa_plus, and GEOS—from the math-mulberry dataset, which has already undergone step segmentation. These categories were chosen due to their relatively consistent question difficulty and more uniform step distribution compared to other categories, making them more suitable for reinforcement learning (RL) training

<image>You are given a question about the image and a step-by-step reasoning progress.
If the provided reasoning steps are already sufficient, output only the final answer.
If the reasoning steps are incomplete, continue generating the subsequent reasoning steps, and then provide the final answer. Ensure the steps are logical and concise.
Always format your response with the following sections, separated by #.
Step (next number):
(New reasoning step here)
Step (next number+1):
(New reasoning step here)
...
The final answer is:
(Your final answer here)

The the instruction following prompt for training is designed to guide the model through an incremental reasoning process. The key goals are:

Promoting Incremental Reasoning: The model is instructed to generate reasoning steps in sequence, ensuring that each step logically leads to the next. This encourages the model to build on previous steps and engage in systematic problem-solving, which is crucial for handling complex tasks.

Ensuring Completeness in Reasoning: If the initial reasoning steps are insufficient, the model is prompted to continue generating additional steps. This ensures that the model can handle incomplete or ambiguous reasoning processes, thereby improving its ability to deal with real-world situations that often require iterative problem-solving.

Structured Response Format: By enforcing a structured response format with clearly delineated steps and a final answer, the prompt fosters consistency in how the model organizes and presents its reasoning. This structure aids in improving the interpretability of the model's outputs and makes it easier for humans to follow the model's thought process.

Adaptability for Training: This design helps the model adapt to a wide range of problem complexities by learning to evaluate when reasoning is sufficient and when further elaboration is required. This encourages the model to handle a variety of reasoning tasks effectively during training.

8.5.2 Prompt for Training

8.5.3 Prompt for Validating

<image>You are given a question about the image.
Generate a step-by-step reasoning progress to solve the problem. Ensure the steps are logical and concise.
Finally, provide the final answer in the following format: \'The final answer is: xxx\'.
Always format your response with the following sections, separated by #."
Step 1:
(Reasoning step 1 here)
Step 2:
(Reasoning step 2 here)
...
The final answer is:
(Your final answer here)

The instruction following prompt for validation focuses on the generation of reasoning without prior steps. The design objectives are:

Encouraging Reasoning from Scratch: The model is required to generate a step-by-step reasoning process from the given question, fostering the model's ability to approach new problems without pre-existing steps. This is crucial for evaluating the model's ability to generalize to unseen tasks and situations.

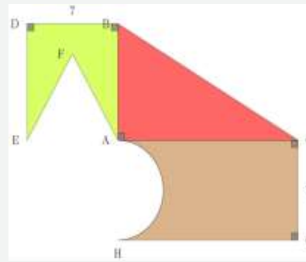
Ensuring Logical and Concise Steps: The emphasis on generating logical and concise reasoning steps encourages the model to produce clear and efficient explanations. This aligns with the goal of creating AI systems that are not only accurate but also able to explain their reasoning in a manner that is understandable and precise.

Providing a Final Answer: After reasoning, the model is instructed to provide the final answer in a clear format. This ensures that the model is able to synthesize its reasoning into a coherent conclusion, which is essential for answering complex questions in real-world applications.

Evaluation of Reasoning Ability: This prompt design is particularly useful for evaluating the model's capacity to generate reasoning and conclusions in a structured format. The clear format of steps and the final answer ensures consistency across multiple test cases, enabling a more accurate assessment of the model's performance.

8.6 CASE

Question 1:



If the ABDEF shape is a rectangle where an equilateral triangle has been removed from one side of it, the area of the ABDEF shape is 36, the ACGH shape is a rectangle where a semi-circle has been removed from one side of it and the area of the ACGH shape is 78, compute the area of the ABC right triangle. Assume π = 3.14. Round computations to 2 decimal places.

MiniCPM-o-2.6:

Model output:

Step 1: The area of the DEFG trapezium equals 70 and DB = $7 \cdot \frac{1}{2} \times (7 + 0) \times 7 = 3.5$.

Step 2: Subtracting the areas of the trapezium and the equilateral triangle gives: $70 - 3.5 - 3.37 = 6.93.$

Step 33 : Let AB = x. Then: $x \times 7 - \frac{\sqrt{3}}{4} \times x^2 \times 7 = 9.35.\ x \times 7 = 9.75$.

Step 33 : The length of the AB side is 1.35.

Step 3 3 : The lengths of the AC and BC sides are 1.25 and 7, respectively. Applying the Pythagorean theorem, we have $AC = \sqrt{BC^2 - AB^2} = \sqrt{7^2 - 1.25^2} = 66.65$.

Step 32: The lengths of the AC and BC sides are 6.65 and 7. Applying the formula for the area of the right triangle gives us. $Area = \frac{1}{2} \times AC \times BC = \frac{1}{2} \times 6.65 \times 7 = 28.4$. Round 8.4 to 2 decimal places resulting in 8.40.

Answer: 8.4

O^2-CRITICURL:

Model output:

#Step1: Let rectangle height = h, AB = x, BC = y.

Shape ABDEF: rectangle area xh minus equilateral triangle area $\frac{\sqrt{3}}{4}h^2$ equals 36:

$$xh - \frac{\sqrt{3}}{4}h^2 = 36.$$

Shape ACGH: rectangle area $(x + y)h$ minus semicircle area $\frac{\pi h^2}{8}$ equals 78:

$$(x + y)h - \frac{\pi h^2}{8} = 78.$$

#Step2: Assume AB vertical $\Rightarrow x = h$.

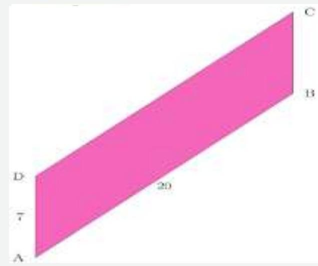
From first equation: $h^2 - 0.4330127h^2 = 36, 0.5669873h^2 = 36, h^2 \approx 63.51,\ h \approx 7.97$.

#Step3: From second equation: $(7.97 + y)(7.97) - 0.3927 \times 63.51 = 78, (7.97 + y)(7.97) - 24.92 = 78, (7.97 + y)(7.97) = 102.92, 7.97 + y \approx 12.91 \Rightarrow y \approx 4.94$

Step4:Area of triangle ABC = $\frac{1}{2}xy \approx \frac{1}{2}(7.97)(4.94) \approx 19.69$.

The final answer is 19.69

Question 2:



Compute the perimeter of the ABCD parallelogram. Round computations to 2 decimal places.

MiniCPM-o-2.6:

Model output:

Step 1: Identify the given parallelogram and its properties.
Step 2: Note that in is given a parallelogram, opposite sides are equal. Thus AD = bc and ab = cd.
Step 33: Given AB=20, CD=7, so perimeter P = 2(AB + CD)
Step 3: Plug in values into formula.
Step 3: Calculate using.
Step 4: Perimeter = 2(20+7)

Answer: 54.0

O^2-CRITICURL:

Model output:

#Step1: In a parallelogram, opposite sides are equal in length.
From the figure, we can identify the lengths of two adjacent sides:

AD = 7, DC = 20.

#Step2: Since ABCD is a parallelogram, the side opposite to AD is BC, and the side opposite to DC is AB.

Therefore, BC = AD = 7, AB = DC = 20.

#Step3: The perimeter of a polygon is the sum of the lengths of all its sides. Perimeter of parallelogram ABCD = AD + DC + CB + BA.
#Step4: Substitute the values of the side lengths into the formula:

Perimeter = 7 + 20 + 7 + 20=54.

Alternatively, the formula for the perimeter of a parallelogram is P = 2 * (a + b), where 'a' and 'b' are the lengths of adjacent sides. P = 2 * (7 + 20), P = 2 * (27), P = 54.
The result is an integer, so no rounding is needed.

The final answer is 54.