Think-with-Rubrics: From External Evaluator to Internal Reasoning Guidance

Jiachen Yu[1,*] **Zhihao Xu**[2,*] **Junjie Wang**[1,†] **Yujiu Yang**[1,†]
[1]Tsinghua University [2]Renmin University of China

Abstract

Rubrics have been extensively utilized for evaluating unverifiable, open-ended tasks, with recent research incorporating them into reward systems for reinforcement learning. However, existing frameworks typically treat rubrics only as external evaluator disjointed from the policy's primary reasoning trace. Such design confines rubrics to post-hoc measurement, leaving them unable to actively guide the model's generation process. In this work, we introduce **THINK-WITH-RUBRICS**, a novel paradigm for instruction following tasks. THINK-WITH-RUBRICS integrates rubric generation into the reasoning context, transforming the rubric from an independent artifact into an internal guidance of LLM's generation. During training, LLM sequentially generates a rubric followed by a response, while a trained rubric verifier provides joint supervision by evaluating the consistency between the answer and the self-generated / golden rubrics. Experiments across multiple benchmarks demonstrate that THINK-WITH-RUBRICS consistently outperforms the Rubric-as-Reward baseline supervised by golden rubrics by an average of 3.87 points. We have also discussed the mechanism by which THINK-WITH-RUBRICS enhances model performance. Experimental results demonstrate that supervision from golden rubrics and self-generated rubrics enhances the performance of THINK-WITH-RUBRICS by improving the quality of self-generated rubrics and increasing the internal consistency of responses respectively.

1 Introduction

Reinforcement learning (RL) has driven substantial progress in LLMs on verifiable tasks [4, 14], where reliable reward signals can be derived from ground-truth labels. However, extending RL to open-ended, unverifiable domains remains challenging, as such tasks lack ground truth. In these settings, scalar rewards [7, 23] and monolithic generative reward models [25, 26] often provide supervision that is too coarse and underspecified to capture response quality.

To address this, recent research has increasingly focused on rubric-as-reward [2, 3, 8, 17, 20] paradigm: using structured sets of criteria to decompose each open-ended task into interpretable objectives. Rewards are then assigned based on how many rubric criteria a model's response satisfies, providing more stable and informative supervision during optimization. Yet, in existing reward systems, these rubrics are only used as external evaluators after the model has finished its response. As a result, these success standards serve only to *measure* the model's performance, not to *guide* its generation.

To operationalize this active guidance, we propose integrating rubric generation directly into the model's trace in Fig. 1. Our motivation for this integration is twofold. First, we draw inspiration from *human metacognition* [9, 16]: when faced with complex decisions, humans naturally begin by reflecting on the requirements of the task at hand and use those criteria to guide subsequent

*Equal contribution.
†Correspondence to Junjie Wang <wangjunjie@sz.tsinghua.edu.cn> Yujiu Yang <yang.yujiu@sz.tsinghua.edu.cn>

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

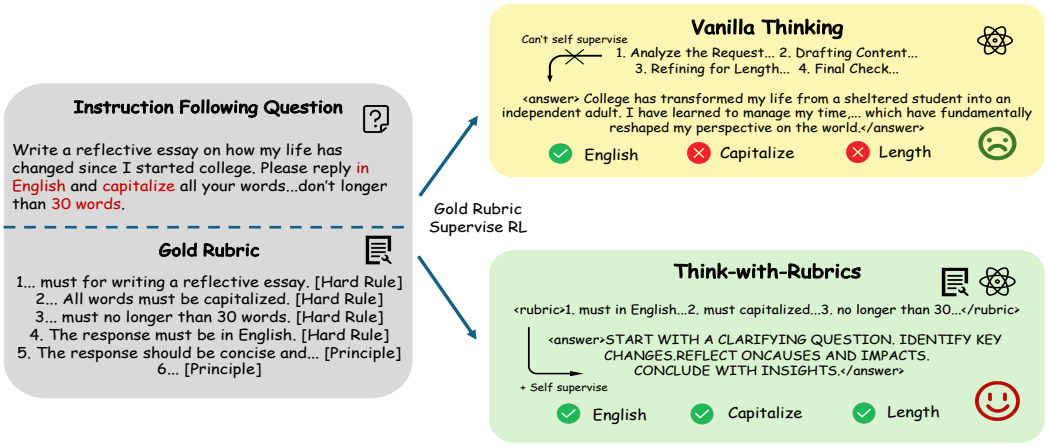

Figure 1: THINK-WITH-RUBRICS transforms rubrics from mere external evaluation metrics into an intrinsic part of the reasoning process. By dual supervision derived from both golden and self-generated rubrics, it effectively enhances the model's instruction-following capabilities.

reasoning and action. This motivates us to treat rubrics not as external evaluators, but as an internal mechanism to actively guide the model toward desired outcomes. Compared to planning-centric reasoning approaches [22, 31], our structured rubrics facilitate per-criterion supervision in RL stage, ensuring that self-generated outputs comply with the reasoning trajectory. Second, our work builds upon the emerging *judge-solve synergy in LLMs* [18, 26, 28]. These studies suggest a deep latent connection between a model's capacity to evaluate (as a reward provider) and its capacity to execute (as a problem solver). By internalizing rubrics as a cognitive step rather than an external evaluating task, we aim to bridge the gap between reward modeling and policy generation.

Based on the above motivation, we introduce THINK-WITH-RUBRICS that internalizes rubrics as essential elements of the model's thinking phase, as shown in Fig. 2. The THINK-WITH-RUBRICS paradigm necessitates that the model explicitly internalizes the construction of task-relevant rubrics within its reasoning trajectory prior to generating a final response. This approach mimics human cognitive processes, where success criteria are established before problem-solving, thereby guiding downstream deliberation and response. During the reinforcement learning phase, the model is optimized through a multi-faceted reward system which comprises three key components: 1) Golden rubric consistency reward provides external supervision by evaluating the response against ground-truth criteria; 2) Self-generated rubric consistency reward enforces internal alignment by verifying whether the response satisfies the model's self-generated rubric; 3) Format reward ensures structural integrity and prevents reward hacking by penalizing insufficient rubric sets. By jointly optimizing these objectives, THINK-WITH-RUBRICS factorizes the policy trajectory into a high quality self-generated rubric followed by a answer that is consistent with self-generated rubric.

Empirical results across multiple instruction-following benchmarks demonstrate that THINK-WITH-RUBRICS consistently outperforms Rubric-as-Reward baselines. On the 8B model, our method achieves improvements across IFEval, IFBench, and InfoBench, yielding a gain of 3.87 average points. This consistent improvement is also validated on the 4B model, where THINK-WITH-RUBRICS raises the average score by 4.16 points, confirming the scalability of the paradigm across model sizes.

Beyond the performance gains, we further provide rich insights into *why* and *how* THINK-WITH-RUBRICS works. Through dedicated mechanism analyses, we reveal two complementary perspectives: (1) Self-generated rubric supervision substantially improves the internal consistency between the model's own rubrics and its generated answers, enabling the model to strictly adhere to self-imposed criteria; and (2) Golden rubric supervision implicitly improves the quality of self-generated rubrics without any direct rubric-level comparison. Remarkably, using only self-generated rubrics for supervision matches or even surpasses golden rubric only supervision, demonstrating the potential for self-evolution under this paradigm. Major contributions of this work are summarized as follows:

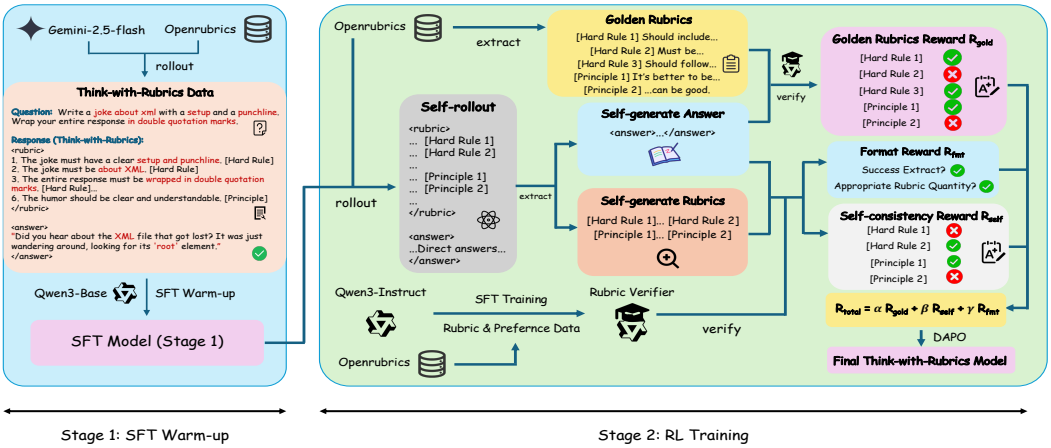

Figure 2: Overview of THINK-WITH-RUBRICS model training. Training is conducted in two phases: SFT warm-up and RL. SFT involves distilling Gemini-2.5-Flash for format adaptation, while RL utilizes a combination of golden and self-generated rubric consistency rewards for joint supervision.

- We propose THINK-WITH-RUBRICS, a paradigm that transforms rubrics from independent evaluator into an internal part of the model's reasoning process. This allows models to dynamically adapt criteria to the specific problem context within the reasoning chain.
- We develop a unified training workflow. By combining supervision from both self-generated and golden rubric, THINK-WITH-RUBRICS consistently outperforms vanilla Rubric-as-Reward training.
- We reveal how self-supervision and external golden supervision enhance the performance of THINK-WITH-RUBRICS model, and demonstrated that THINK-WITH-RUBRICS remains consistently effective across various training hyperparameters and model scales.

2 Related Works

Rubric-based evaluation. Rubric-based evaluation has recently emerged as a prominent methodology for assessing open-ended tasks. By decomposing task quality into a structured set of interpretable criteria, this approach provides finer-grained supervisory signals and enhanced interpretability compared to standard, holistic LLM-as-a-judge methods. Benchmarks such as JudgeBench [19] and FireBench [29] inherently utilize rubrics to assess model performance. Concurrently, works including *Openrubrics* [8] and RubricHub [5] concentrate on designing automated rubric generation frameworks, facilitating the construction of corresponding prompt-rubric paired datasets. Our research systematically builds upon the rubric paradigms and datasets established by these preceding efforts to further advance this domain.

Rubric-based reward. Building upon rubric-based evaluation, an emerging line of work transforms rubrics into training signals under the rubric-as-reward [3] paradigm. Instead of only output a single scalar score, these methods compute rewards based on rubric-level satisfaction. For instance, Rubric-RM [8], trained on the *Openrubrics* dataset, functions as an effective supervisory model that significantly enhances the performance of downstream policies. RLCF [20] extracts task-specific checklists from instructions and synthesizes them into fine-grained RL rewards utilizing both LLM-as-a-judge mechanisms and specialized verifier programs. SciIF [17] constructs rubric-style constraint catalogs for individual scientific tasks to facilitate RL optimization specifically within scientific domains. However, existing methods largely treat rubrics as external supervisory artifacts where rubrics are typically introduced only after a candidate response has been generated. The rubric influences optimization only through delayed supervision. Instead, we incorporate rubric generation directly into the model's reasoning trajectory, allowing the model to first formulate success criteria and then reason under those criteria. This enables a tighter coupling between thinking and evaluation.

3 Methods

3.1 Problem Formulation

Setting. In this paper, we consider the task of open-ended instruction following, which is crucial for aligning LLMs with human intentions. In this setting, an input query x typically contains multiple constraints, ranging from explicit requirements that must be followed (e.g., "output in JSON") to implicit subjective preferences (e.g., "be concise and professional").

Rubrics. To operationalize the evaluation of such tasks, we define a rubric $r = \{c_1, c_2, \ldots, c_n\}$ as a structured collection of n discrete evaluation criteria. In our training setting, each instruction x is associated with a *golden rubric* r^*, representing the ground-truth set of constraints provided by *Openrubrics* dataset. In this dataset, criteria $c_k^* \in r^*$ are decomposed into two categories c_{hard}^* and $c_{\text{principle}}^*$, where c_{hard}^* contains explicit task constraints (e.g., format or instruction requirements), and $c_{\text{principle}}^*$ captures higher-level quality principles such as completeness and helpfulness.

Verifier. To assess whether a candidate response y fulfills a criterion, we employ a *rubric verifier* $\mathcal{V}$. Specifically, for each individual criterion $c_k \in r$, the verifier produces a binary judgment $v_k = \mathcal{V}(y, c_k) \in \{0, 1\}$, where $v_k = 1$ indicates that the response y successfully satisfies the criterion c_k, and $v_k = 0$ otherwise. Criterion-level judgments will be aggregated into a rubric-level consistency score in Section 3.3. Compared to scalar reward models, this rubric-as-reward paradigm provides a more fine-grained and interpretable assessment of response quality.

Our Paradigm (THINK-WITH-RUBRICS). Unlike previous works on rubrics that only regard rubrics as the reward, THINK-WITH-RUBRICS internalizes rubric generation as part of the policy trajectory, so that the answer is conditioned on explicitly generated task criteria. We formally define the input and output as follows.

The policy π_θ in THINK-WITH-RUBRICS first gives a *self-generated* rubric $\hat{r} \sim \pi_\theta(\hat{r} \mid x)$ to establish an internal cognitive assessment of the given task x. Then, the model produces the final response $y \sim \pi_\theta(y \mid x, \hat{r})$ conditioned on both x and the *self-generated* rubric $\hat{r}$. The joint probability of the trajectory is factorized as:

$$\pi_\theta(\tau \mid x) = \pi_\theta(\hat{r} \mid x) \cdot \pi_\theta(y \mid x, \hat{r}) \tag{1}$$

where θ denotes the model parameters. In this way, we transform the model from a passive executor into an active deliberator that first defines the rules of success before finalizing the answer.

3.2 SFT Warm Up

During the training and evaluation processes, we instruct the model to adhere to a specific output format through prompt guiding, in order to facilitate the parsing and extraction of answers and rubrics. The structured output format is `<rubric>` $\hat{r}$ `</rubric>` followed by `<answer>` y `</answer>`, where $\hat{r}$ denotes the generated rubric and y denotes the final response.

However, we observed that approximately 20% of the samples failed to generate responses strictly following the format when directly using the base model for evaluation. To prevent reinforcement learning from wasting too many steps on format compliance and to fully utilize the data, we first performed supervised fine-tuning (SFT) to adapt the model to the THINK-WITH-RUBRICS generation format. The objective of this stage is mainly to initialize the policy with a stable trajectory structure that explicitly separates rubric construction from answer generation.

Specifically, for each instruction x, we distill responses from Gemini-2.5-Flash on *Openrubrics* dataset. Following the format mentioned in Section 3.1, the *self-generated* criteria $\hat{c}_k \in \hat{r}$ are also decomposed into $\hat{c}_{\text{hard}}$ and $\hat{c}_{\text{principle}}$ which represent different importance levels.

The SFT objective is standard next-token maximum likelihood over the entire trajectory:

$$\mathcal{L}_{\text{SFT}} = -\mathbb{E}_{(x,\tau)\sim\mathcal{D}} \left[\sum_{t=1}^{|\tau|} \log \pi_\theta(\tau_t \mid x, \tau_{<t}) \right], \tag{2}$$

where $\tau = (\hat{r}, y)$ denotes the distilled teacher trajectory.

3.3 RL Training

3.3.1 Rubric Verifier

In RL stage, assigning rewards based on fine-grained rubrics necessitates a robust and reliable rubric verifier. Rather than directly employing off-the-shelf LLMs as the verifier, we train a specialized rubric verifier $\mathcal{V}$ on the *OpenRubrics* dataset. The aim to train the verifier is to achieve superior alignment with reference labels in *OpenRubrics*, ensuring more accurate and stable reward signals during the RL process.

To boost efficiency and improve accuracy of rubric verification during the large batchsize and rollout size training process, we distill a lightweight Qwen3-8B rubric verifier from Qwen3-32B-Instruct using SFT. Given a rubric-response pair (r, y), verifier $\mathcal{V}$ outputs a judgment $v_k = \mathcal{V}(y, c_k) \in \{\texttt{met}, \texttt{not met}\}$ for each criterion $c_k \in r$.

We adopt an implicitly aggregated verification scheme, where the verifier jointly evaluates all rubric items in a single pass and produces a weighted compliance score

$$S(y, r) = \frac{w_h \sum_{c_i \in c_{\text{hard}}} \mathbb{I}[\mathcal{V}(y, c_i) = \texttt{met}] + w_p \sum_{c_j \in c_{\text{principle}}} \mathbb{I}[\mathcal{V}(y, c_j) = \texttt{met}]}{w_h |c_{\text{hard}}| + w_p |c_{\text{principle}}|}, \tag{3}$$

where c_{hard} and $c_{\text{principle}}$ denote hard rules and principles criterion subset in rubric r respectively. w_h and w_p are hyperparameters representing their corresponding weights. This allows for flexible weighting of hard rules and principles based on their relative importance.

3.3.2 Reward System

The reward system in THINK-WITH-RUBRIC consists of three components with different weights:

$$R(\tau) = \alpha R_{\text{gold}} + \beta R_{\text{self}} + \gamma R_{\text{fmt}}. \tag{4}$$

Golden rubric consistency reward. The golden rubric consistency reward measures the compliance of the final answer with the golden rubric $r^\star$ from *Openrubrics*:

$$R_{\text{gold}} = S(y, r^\star). \tag{5}$$

When $\alpha = 0$, our approach reduces to a fully self-supervised self-evolution paradigm.

Self-generated rubric consistency reward. The self-generated rubric consistency reward evaluates the compliance of the final answer with the self-generated rubric:

$$R_{\text{self}} = S(y, \hat{r}). \tag{6}$$

This reward measures self-alignment within the self-generated rubrics. Importantly, we do not directly supervise the quality of the generated rubric itself. Instead, the improvement of self-generated rubric is achieved indirectly through the joint optimization of R_{gold} and R_{self}. We assume that if an answer that satisfies the self-generated rubric also receives a high score under the golden rubric, the generated rubric is implicitly encouraged to align with the golden standard:

$$S(y, \hat{r}) \uparrow \text{ and } S(y, r^\star) \uparrow \Rightarrow \hat{r} \approx r^\star. \tag{7}$$

We will verify this assumption in Section 4.3 by examining whether the rubric generated by the model which supervised by both self-generated and golden rubric consistency reward is truly of high quality.

Format reward. The use of self-generated rubric consistency reward inherently introduces the risks of *reward hacking*, where it could generate a minimal set of trivial and easily satisfied criteria to artificially get higher reward. To prevent this, we introduce a format reward R_{fmt}. We directly extract the model's self-generated rurbic $\hat{r}$ between special tokens `<rubric>` and `</rubric>`, and extract the final answer y between `<answer>` and `</answer>`. The model gets format reward only when both the rubric and answer are successfully parsed from model output.

According to the distribution of criteria for golden rubrics in *Openrubrics*, we constrain the number of criterion $n(\hat{r}) = |\hat{c} \in \hat{r}|$ in a reasonable range. In *Openrubrics*, 99.61% of the samples have a criterion count ranging from 5 to 15, with an average of 8.66 and a mode of 10. So we define

$$R_{\text{fmt}} = \mathbb{I}[\texttt{parseable}] \cdot \max\left(0, 1 - \frac{|n(\hat{r}) - 10|}{5}\right), \tag{8}$$

which reaches its maximum at $n(\hat{r}) = 10$ and linearly decreases to 0 at $n(\hat{r}) = 5$ and 15. This discourages the model from generating only a few trivial rubrics or excessively redundant criteria.

The policy is then optimized using DAPO without KL loss [27]:

$$\mathcal{L}_{\text{RL}} = -\mathbb{E}_{\tau \sim \pi_\theta} \left[A(\tau) \log \pi_\theta(\tau \mid x) \right] \tag{9}$$

where $A(\tau)$ is the DAPO advantage computed from the trajectory reward in Eq. 4.

4 Experiments

4.1 Experimental Settings

Models. Our policy model is initialized from Qwen3-8B-Base [24]. For RL stage, the rubric verifier is built upon Qwen3-8B-Instruct and is distilled from teacher model Qwen3-32B-Instruct. To ensure consistency with our framework, all training stages are conducted under the non-thinking mode, so that our policy strictly follows the `<rubric>` $\rightarrow$ `<answer>` generation paradigm.

Data construction. All training data are derived from the *Openrubrics* [8] dataset. For the SFT warm-up stage, we use Gemini-2.5-Flash [1] as the teacher model to sample answers that follow the THINK-WITH-RUBRICS generation format. The prompt we use can be founded at Appendix A.4. Specifically, we perform rollouts on 5k samples from *Openrubrics* to construct the SFT training set.

For verifier training, we use a 2k split of OpenRubrics. On this split, Qwen3-32B-Instruct performs rollouts over the original chosen and rejected responses provided by the dataset. For each response pair, the teacher model evaluates whether each rubric item is satisfied and produces criterion-level verification traces. These traces are then filtered using the ground-truth preference labels provided in *Openrubrics*, and only label-consistent verification trajectories are retained for verifier training. After training, the resulting rubric verifier achieves 93.2% rubric-level judgment accuracy on 1.5k response pairs, outperform than original Qwen3-8B (86.3%), demonstrating strong reliability as a evaluator.

For RL training, we use a 27k split of *Openrubrics*. The policy is optimized on this split using DAPO under the reward formulation described in Section 4. All data splits used across the SFT warm-up, verifier training / eval, and RL training stages are strictly non-overlapping, ensuring no data leakage between any training or evaluation phase.

Training parameters. In the SFT stage, we train 200 steps with a learning rate of 1e-5 and a batch size of 32 to obtain the optimal SFT model. The SFT of the rubric verifier uses the same parameters. In the RL stage, we use a batch size of 32, roll out 8 replies per prompt, with a learning rate of 1e-6, and achieve optimal results in around 90 steps. For the reward system, the value of α, β, γ is 0.3, 0.5 and 0.2, with $w_h = 2$ and $w_p = 1$. Detailed parameter sensitivity ablation can be found in 4.4, and more detailed parameter settings can be found in the Appendix A.3.

Baseline. For the baseline, we adopt the Rubrics-as-Reward paradigm [3] with two different prompt settings. One directly prompts the model to generate answers without any intermediate reasoning. And the other instructs the model to first perform step-by-step reasoning and then produce the final response within `<answer>` and `</answer>` tags, representing a general thinking process. This setting serves to validate whether regarding rubric generation as a specialized form of thinking is better than undifferentiated free-form reasoning. Both settings undergo the same two-stage training pipeline of SFT followed by RL. Only golden rubric consistency reward defined in Eq. 5 is directly employed to realize the Rubrics-as-Reward objective during the RL stage.

Evaluation benchmarks. We evaluate our method on three representative instruction-following benchmarks: *IFEval* [32], *IFBench* [11], and *InfoBench* [13]. We follow the standard settings of each benchmark for evaluation. These benchmarks jointly cover strict instruction constraints, multi-rule compliance, and open-ended instruction-following quality, providing a comprehensive evaluation of the proposed THINK-WITH-RUBRICS paradigm. To test whether the gains transfer from explicit-constraint instruction following to fully open-ended settings, we additionally evaluate on three general open-ended benchmarks: *WildBench* [6] with real-world user–LLM conversations; *EvalAgent* [21] with expert-level evaluation criteria from the web for each open-ended task; and *AgentIF* [12] built from 50 real-world agentic applications.

Method	IFEval		IFBench		InfoBench	Average
	Prompt	Instr.	Prompt	Instr.		
Base Model						
Qwen3-8B-Base	50.28	59.83	16.32	19.40	72.09	46.23
Rubric-as-Reward						
SFT Warm-up (Direct answer)	57.30	67.51	18.03	18.51	78.27	51.20
+ DAPO (Golden rubric reward)	58.96	68.82	20.41	20.60	<u>82.44</u>	53.94
SFT Warm-up (With think)	52.68	62.35	<u>24.83</u>	**28.91**	78.40	51.97
+ DAPO (Golden rubric reward)	59.70	68.35	22.11	23.88	80.22	54.01
Think-with-Rubrics (Ours)						
SFT Warm-up	61.00	70.02	19.05	19.40	79.60	53.22
+ DAPO (Golden rubrics reward only)	<u>64.87</u>	**73.14**	23.13	26.87	<u>82.44</u>	56.81
+ DAPO (Self-generated reward only)	64.69	72.30	**25.85**	<u>27.76</u>	80.98	<u>57.17</u>
+ DAPO (Mixed reward)	**65.61**	<u>73.02</u>	<u>24.83</u>	26.27	**83.20**	**57.88**

Table 1: Main results on three instruction-following benchmarks. For IFEval and IFBench, we report both prompt-level and instruction-level scores under strict evaluation. The **Average** column is computed using the prompt-level scores of IFEval and IFBench together with the InfoBench score. Best results are in **bold** and second-best results are <u>underlined</u> (the same in other tables).

Method	WildBench	EvalAgent	AgentIF
Base Model			
Qwen3-8B-Base	2.35	28.84	12.59
Rubric-as-Reward			
+ DAPO (Golden rubric reward)	3.57	39.95	15.70
Think-with-Rubrics (Ours)			
+ DAPO (Golden rubrics reward only)	<u>4.69</u>	<u>47.77</u>	17.40
+ DAPO (Self-generated reward only)	4.33	43.02	**17.96**
+ DAPO (Mixed reward)	**4.76**	**50.12**	17.26

Table 2: Results on open-ended benchmarks. For WildBench, we report the reference-free WB-Score (1–10) on the v2-hard subset. For EvalAgent, we report the satisfaction rate (%) over expert-mined criteria . For AgentIF, we report the Instruction Success Rate (ISR, %), i.e., the fraction of tasks where all constraints are satisfied.

4.2 Main Results Analysis

Based on the main experimental results in Table 1, we draw the following conclusions:

THINK-WITH-RUBRICS consistently improves over origin Rubric-as-Reward methods. THINK-WITH-RUBRICS consistently outperforms the Rubric-as-Reward baselines across all benchmarks. Compared with the conventional setting of only optimize answers under golden rubric supervision, our method achieves clear gains in *both* SFT and RL stages, demonstrating the effectiveness of integrating rubric into the trajectory. Using rubrics as the thinking process outperforms natural language thinking, as rubrics provide structured, step-by-step guidance that can be directly leveraged for self-supervision. These results directly support our central motivation: rubrics can serve not only as post-hoc evaluation criteria, but also as internal guidance for reasoning and answer. Moreover, under the optimal mixed reward, the average thinking length of THINK-WITH-RUBRICS (231 tokens) is less than vanilla thinking (401 tokens). This confirms that rubrics effectively condense vital problem-solving requirements into a streamlined reasoning process.

Mixed consistency rewards provide the most effective supervision. Among all RL settings, the mixed reward using both self-generated and golden rubric consistency achieves the best overall performance. This validates our indirect supervision hypothesis: explicit supervision on rubric quality is not necessary, as rubric learning can be achieved through the joint optimization of answer consistency under both self-generated and golden rubrics. Empirically, the two rewards address complementary failure modes of THINK-WITH-RUBRICS: golden rubric consistency mainly improves

Rubric Generator Model	Fact.	Focus	Safety	Math	Precise IF	Avg.
Qwen3-8B-Base	65.26	87.47	59.33	79.23	65.00	71.26
SFT Warm-up	72.42	90.51	**66.22**	72.68	56.88	71.74
+ DAPO (Golden rubrics reward only)	**75.37**	**90.71**	59.33	79.23	**68.12**	**74.55**
+ DAPO (Self-generated reward only)	65.05	82.63	58.44	77.05	57.50	68.13
+ DAPO (Mixed reward)	74.32	89.90	61.11	**80.33**	62.50	73.63

Table 3: Rubric quality evaluation on RewardBench-2. We use rubrics generated by different models to guide Qwen3-8B judging and report macro-average accuracy across all subsets. The RL-trained THINK-WITH-RUBRICS model with golden rubrics rewards achieves better overall macro score.

rubric quality, while self-generated rubric consistency encourages the model to generate answers that faithfully follow its own rubric. Their combination therefore yields the strongest performance.

Self-evolution is feasible under the THINK-WITH-RUBRICS paradigm. Notably, using only self-consistency reward already achieves highly competitive performance (57.17% average point), even outperforming the golden-only setting (56.81% average point). This strongly suggests that self-evolution is feasible once rubrics are internalized into the reasoning process. We discuss the reasons behind this phenomenon in Section 4.3, namely that supervision via self-generated rubrics substantially improves the self-consistency of the model's outputs. Since the SFT-warmed model is already capable of generating preliminary usable rubrics, there exists the potential for self-evolution driven solely by self-supervision. The model can use self-generated rubrics as an effective reasoning scaffold and progressively improve through self-consistent optimization, providing strong evidence that rubric-guided reasoning is an effective path toward scalable self-evolving alignment.

Generalization to open-ended and agentic scenarios. As shown in Table 2, THINK-WITH-RUBRICS also surpasses the Rubric-as-Reward baseline with thinking across general open-ended benchmarks, achieving gains of $+1.19$ on WildBench, $+10.17\%$ on EvalAgent, and $+1.56$ on AgentIF. The mixed reward setting performs best on the two open-ended benchmarks. These results demonstrate that the benefits of internalizing rubrics as reasoning guidance extend well beyond verifiable instruction following to open-ended, implicit-constraint, and agentic scenarios.

4.3 Mechanism analysis

Supervise answer with golden rubrics can implicitly help improve self-generated rubrics. To verify the assumption proposed in Eq 7, which leverages golden rubrics to implicitly supervise the quality of self-generated rubrics, and to further investigate the underlying reasons for the enhanced quality of self-generated rubrics under the THINK-WITH-RUBRICS paradigm, we evaluate rubrics produced under the THINK-WITH-RUBRICS paradigm on RewardBench-2 [10]. Specifically, we use the rubrics generated from Qwen3-8B-Base, the SFT warm-up model, and the final RL-trained model under several reward settings to guide Qwen3-8B-Instruct judge answer on RewardBench-2, and report macro-average accuracy across all subsets.

As shown in Table 3, RL-trained model with only golden rubrics supervision performs the best, followed by mixed reward. Both settings incorporate supervision from golden rubrics, with full golden rubric supervision achieving superior performance. This validates that even without a direct comparison between the golden and self-generated rubrics, indirectly supervising the model based on how well the answer generated conditioned on the self-generated rubrics complies with the golden rubrics can implicitly drive the self-generated rubrics closer to the golden standard.

Model	Num	Prec.	Rec.	F1
Qwen3-8B-Base	4.2	**74.2**	31.5	41.5
Golden-only	8.8	70.9	**44.2**	**52.1**
Self-only	8.5	70.4	40.5	48.9
Mixed	8.3	71.4	43.6	51.9

Table 4: Direct criterion-level alignment between self-generated and golden rubrics on EvalAgent (%).

Beyond the evidence above, we directly measure the criterion-level alignment between self-generated rubrics and golden rubrics on EvalAgent, judged criterion-by-criterion by Gemini-2.5-Flash, and report Precision, Recall, and F1 in Table 4. Consistent with RewardBench-2, the Golden-only and Mixed settings achieve higher F1, verifying that golden rubric supervision indirectly improves self-generated rubric quality as assumed in Eq. 7. Compared with the base model, whose Precision is

Model	IFEval	IFBench
SFT Warm-up	58.91	60.15
+ DAPO (Golden rubrics reward only)	68.70	75.12
+ DAPO (Self-generated reward only)	69.00	75.45
+ DAPO (Mixed reward)	**69.38**	**77.82**

Table 5: Consistency score of self-generated answer and rubric on IFEval and IFBench.

$\alpha : \beta$	IFEval	IFBench	InfoBench	Avg.
5 : 3	65.06	23.13	83.11	57.10
1 : 1	64.69	23.13	**83.56**	57.13
3 : 5	**65.61**	**24.83**	83.20	**57.88**

Table 6: Ablation on reward mixing ratio. IFEval and IFBench are strict prompt-level accuracy (%). The same in Table 7 and 9.

inflated by generating far fewer criteria (4.2), THINK-WITH-RUBRICS generates markedly richer rubrics (≈8.5 criteria) while largely preserving rubric accuracy.

However, rubric quality is not directly coupled with performance on instruction-following benchmarks. Rubrics produced under self-only supervision setting, despite demonstrating strong performance on instruction-following benchmarks, incur a performance drop when utilized to guide LLM-as-a-judge tasks. This reveals that self-supervision does not produce better general-purpose evaluation criteria, but rather internal control cues optimized for the model's own generation process. The value of self-generated rubrics under self-supervised setting lies not in serving as external evaluators, but in maintaining constraint consistency during the model's own answer generation.

Self-generated rubrics supervision improves performance by enhancing responses' internal consistency. We used the rubric verifier to evaluate the average score of consistency between the answers and rubrics generated by the model itself on IFEval and IFBench. The consistency score calculation formula is based on Eq. 3. The experimental results are shown in Table 5.

From the results, it can be seen that adding self-generated reward in RL training improves the consistency between the model's self-generated answers and rubrics by 14.07 points on average, while SFT model exhibits significant self-inconsistency. This reveals a possible mechanism for using only self-generated rubrics for self-evolve, where untrained models perform poorly when THINK-WITH-RUBRICS not only because they cannot generate high-quality rubrics, but also because they cannot provide consistent responses based on their own rubrics. Golden and self-generated rubrics supervision have each alleviated this problem from two aspects.

4.4 Sensitivity Analysis and Ablation Study

Sensitivity analysis of reward weight. We further study the sensitivity of THINK-WITH-RUBRICS to the choice of reward weights by varying the ratio between the golden rubric consistency reward weight α and the self-generated rubric consistency reward weight β across three settings: 5:3, 1:1, and 3:5. As shown in Table 6, all three settings achieve strong performance across three benchmarks, with 3:5 obtaining the best average score. The performance differences across different ratios remain relatively small (within 1.3%), suggesting that THINK-WITH-RUBRICS is robust to reward weights.

Ablation on model size. To verify whether THINK-WITH-RUBRICS remains effective across varying model scales, we conduct the same experiment on Qwen3-4B-Base, with the results in Table 7. The result shows that THINK-WITH-RUBRICS consistently outperforms the vanilla rubric-as-reward baseline on the 4B model. Notably, the two experimental configurations incorporating self-supervision like self-only and mixed exhibit significant improvements. These two settings substantially mitigate the performance degradation typically induced by reinforcement learning on the more challenging IFBench. This might because smaller models, constrained by their foundational capabilities, lack self-consistency and are hard to adhere to self-generated rubrics during answer generation. Consequently, the primary performance gains are derived directly from the self-generated rubric supervision.

Method	IFEval	IFBench	Avg.
Base Model			
Qwen3-4B-Base	36.97	21.43	29.20
Rubric-as-Reward			
SFT Warm-up (Direct answer)	48.80	21.43	35.12
+ DAPO (Golden rubric reward)	56.38	19.05	37.72
SFT Warm-up (With think)	48.98	**23.88**	36.34
+ DAPO (Golden rubric reward)	58.04	20.07	39.05
Think-with-Rubrics (Ours)			
SFT Warm-up	55.45	<u>22.79</u>	39.12
+ DAPO (Golden rubric only)	<u>61.74</u>	17.35	39.55
+ DAPO (Self-gen rubric only)	**63.96**	22.45	**43.21**
+ DAPO (Mixed rubric reward)	61.18	21.77	<u>41.48</u>

Table 7: Results of 4B model on IFEval and IFBench.

Method	IFEval	IFBench	EvalAgent
Performance			
With count reward	65.61	24.83	50.12
Without count reward	64.70	25.85	45.62
Avg. number of criteria			
With count reward	6.7	6.5	8.3
Without count reward	4.0	3.7	4.5

Table 8: Ablation on rubric-count term reward.

$w_h : w_p$	IFEval	IFBench	Avg.
$2:1$	**65.61**	**24.83**	**45.22**
$1:1$	64.70	24.49	44.60

Table 9: Ablation on reward weight ratio of hard rules and principles.

Ablation on the rubric-count term in format reward. To examine this design of the count term in the format reward (Eq. 8), we remove it and retrain the model. As shown in Table 8, removing this term barely changes performance on the explicit instruction-following bench, but causes a large drop of 4.5 points on EvalAgent, whose open-ended tasks require far more comprehensive criteria. The reason can be seen from the average number of generated criteria: without the count reward, the model generates fewer criteria, and the gap widens as tasks become more complex (4.5 vs. 8.3 on EvalAgent). Qualitatively, the rubric collapses into a few coarse-grained criteria that merge multiple atomic constraints (e.g., 2 criteria with 27% satisfaction for an artwork-description task, vs 13 criteria with 81% under the count reward), making the reward signal sparser and weaker. The count term can prevent such rubric degeneration and keeps the reward signal dense enough for complex tasks.

Sensitivity analysis of rubric weights in reward. The parameters w_h and w_p in the original reward formulation denote the weights assigned to the mandatory hard rules and the recommended principles, respectively. We adjust both parameters to 1 to investigate whether the model-generated rubrics under the THINK-WITH-RUBRICS paradigm exhibit distinction in importance. The experimental results, presented in Table 9, demonstrate that the original weight configuration yields a marginal improvement over the equal-weight setting. This finding validates that categorizing the generated rubrics into two distinct classes successfully captures their relative importance, but it isn't the dominant factors influencing the overall training.

4.5 Case study

Fig. 1 shows a case in IFEval that highlights the difference between vanilla thinking and THINK-WITH-RUBRICS. When employing vanilla thinking, due to the lack of self-consistency, the final answer fails to satisfy the constraints the model itself has identified. In contrast, THINK-WITH-RUBRICS explicitly materializes the task constraints into a structured rubric first. And self-consistency reward help the model follow its own rubric. The full view of the case can be seen in Appendix A.2.

5 Conclusion

We propose THINK-WITH-RUBRICS, a novel paradigm that internalizes rubric generation as a component of model's reasoning process. Rather than treating rubrics as post-hoc evaluation tools, our framework conditions answer generation on self-constructed rubrics to guide subsequent answer generation within the same trajectory. We develop a unified RL training workflow that jointly supervises the model with both self-generated and golden rubric consistency rewards, eliminating the inefficiencies inherent in decoupled rubric-based optimization. Extensive experiments on several benchmarks demonstrate that THINK-WITH-RUBRICS consistently outperforms the Rubrics-as-Reward baseline, with the mixed reward setting achieving the best overall performance. Mechanistic analyses further reveal that golden rubric supervision implicitly elevates self-generated rubric quality, while self-consistency supervision mitigates internal inconsistency between the model's own rubrics and its generated answers. Notably, the competitive performance of the self-only setting suggests that THINK-WITH-RUBRICS provides a viable path toward scalable self-evolution, where models can progressively improve without reliance on external annotation. We hope this work inspires further exploration of rubric-integrated reasoning as a general paradigm for open-ended task alignment.

6 Acknowledgements

This work was partly supported by the National Natural Science Foundation of China (Grant No. 62576191), the Shenzhen Science and Technology Program (ZDCY20250901103533010).

References

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

[2] Jacob Dineen, Aswin Rrv, Qin Liu, Zhikun Xu, Xiao Ye, Ming Shen, Zhaonan Li, Shijie Lu, Chitta Baral, Muhao Chen, and Ben Zhou. QA-LIGN: Aligning LLMs through constitutionally decomposed QA. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 20619–20642, Suzhou, China, 2025. Association for Computational Linguistics.

[3] Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. *arXiv preprint arXiv:2507.17746*, 2025.

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

[5] Sunzhu Li, Jiale Zhao, Miteto Wei, Huimin Ren, Yang Zhou, Jingwen Yang, Shunyu Liu, Kaike Zhang, and Wei Chen. Rubrichub: A comprehensive and highly discriminative rubric dataset via automated coarse-to-fine generation. *arXiv preprint arXiv:2601.08430*, 2026.

[6] Bill Yuchen Lin, Yuntian Deng, Khyathi Chandu, Abhilasha Ravichander, Valentina Pyatkin, Nouha Dziri, Ronan Le Bras, and Yejin Choi. Wildbench: Benchmarking llms with challenging tasks from real users in the wild. In *International Conference on Learning Representations*, pages 47852–47870, 2025.

[7] Chris Yuhao Liu, Liang Zeng, Yuzhen Xiao, Jujie He, Jiacai Liu, Chaojie Wang, Rui Yan, Wei Shen, Fuxiang Zhang, Jiacheng Xu, et al. Skywork-reward-v2: Scaling preference data curation via human-ai synergy. *arXiv preprint arXiv:2507.01352*, 2025.

[8] Tianci Liu, Ran Xu, Tony Yu, Ilgee Hong, Carl Yang, Tuo Zhao, and Haoyu Wang. Openrubrics: Towards scalable synthetic rubric generation for reward modeling and llm alignment. *arXiv preprint arXiv:2510.07743*, 2025.

[9] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594, 2023.

[10] Saumya Malik, Valentina Pyatkin, Sander Land, Jacob Morrison, Noah A Smith, Hannaneh Hajishirzi, and Nathan Lambert. Rewardbench 2: Advancing reward model evaluation. *arXiv preprint arXiv:2506.01937*, 2025.

[11] Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hannaneh Hajishirzi. Generalizing verifiable instruction following. *arXiv preprint arXiv:2507.02833*, 2025.

[12] Yunjia Qi, Hao Peng, Xiaozhi Wang, Amy Xin, Youfeng Liu, Bin Xu, Lei Hou, and Juanzi Li. Agentif: Benchmarking instruction following of large language models in agentic scenarios. *arXiv preprint arXiv:2505.16944*, 2025.

[13] Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. Infobench: Evaluating instruction following ability in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13025–13048, 2024.

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

[15] 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*, pages 1279–1297, 2025.

[16] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

[17] Encheng Su, Jianyu Wu, Chen Tang, Lintao Wang, Pengze Li, Aoran Wang, Jinouwen Zhang, Yizhou Wang, Yuan Meng, Xinzhu Ma, et al. Sciif: Benchmarking scientific instruction following towards rigorous scientific intelligence. *arXiv preprint arXiv:2601.04770*, 2026.

[18] Shaoning Sun, Jiachen Yu, Zongqi Wang, Xuewei Yang, Tianle Gu, and Yujiu Yang. S2j: Bridging the gap between solving and judging ability in generative reward models. *arXiv preprint arXiv:2509.22099*, 2025.

[19] Sijun Tan, Siyuan Zhuang, Kyle Montgomery, William Y Tang, Alejandro Cuadron, Chenguang Wang, Raluca Ada Popa, and Ion Stoica. Judgebench: A benchmark for evaluating llm-based judges. *arXiv preprint arXiv:2410.12784*, 2024.

[20] Vijay Viswanathan, Yanchao Sun, Shuang Ma, Xiang Kong, Meng Cao, Graham Neubig, and Tongshuang Wu. Checklists are better than reward models for aligning language models. *arXiv preprint arXiv:2507.18624*, 2025.

[21] Manya Wadhwa, Zayne Sprague, Chaitanya Malaviya, Philippe Laban, Junyi Jessy Li, and Greg Durrett. Evalagent: Discovering implicit evaluation criteria from the web. *arXiv preprint arXiv:2504.15219*, 2025.

[22] Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 2609–2634, 2023.

[23] Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J Zhang, Makesh N Sreedhar, and Oleksii Kuchaiev. Helpsteer 2: Open-source dataset for training top-performing reward models. *Advances in Neural Information Processing Systems*, 37:1474–1501, 2024.

[24] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[25] Ziyi Ye, Xiangsheng Li, Qiuchi Li, Qingyao Ai, Yujia Zhou, Wei Shen, Dong Yan, and Yiqun Liu. Learning llm-as-a-judge for preference alignment. In *The Thirteenth International Conference on Learning Representations*, 2025.

[26] Jiachen Yu, Shaoning Sun, Xiaohui Hu, Jiaxu Yan, Kaidong Yu, and Xuelong Li. Improve llm-as-a-judge ability as a general ability. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 14110–14126, 2025.

[27] 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.

[28] Taolin Zhang, Maosong Cao, Alexander Lam, Songyang Zhang, and Kai Chen. Compassjudger-2: Towards generalist judge model via verifiable rewards. *arXiv preprint arXiv:2507.09104*, 2025.

[29] Yunfan Zhang, Yijie Bei, Jetashree Ravi, and Pawel Garbacki. Firebench: Evaluating instruction following in enterprise and api-driven llm applications. *arXiv preprint arXiv:2603.04857*, 2026.

[30] Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 3: system demonstrations)*, pages 400–410, 2024.

[31] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*, 2022.

[32] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

A Appendix

A.1 Limitation

Despite the promising results achieved by THINK-WITH-RUBRICS, this work has two primary limitations. First, the scope of empirical validation is confined to instruction-following tasks, where success criteria are relatively well-defined and amenable to rubric decomposition. Whether the proposed paradigm generalizes to broader open-ended domains, such as creative writing, long-form summarization, or multi-turn dialogue, remains an open question. Second, internalizing rubric generation as an explicit reasoning step inevitably introduces additional inference overhead, as the model is required to produce a structured rubric prior to generating the final response. This leads to longer output sequences and increased decoding latency relative to direct-answer baselines, potentially limiting the practical applicability of the framework in latency-sensitive deployment scenarios. We leave the exploration of more efficient rubric generation strategies, such as compressed or implicit rubric representations, to future work.

A.2 Case Study

We present a representative failure case from IFEval to illustrate the behavioral difference between the Rubric-as-Reward baseline and THINK-WITH-RUBRICS, as shown in Figure 3. The instruction requires the model to provide recommendations for writing a reflective essay on college life, subject to three simultaneous hard constraints: the response must be written in English, all words must be fully capitalized, and the total length must not exceed 30 words. Under strict evaluation, failing to satisfy any single constraint results in a zero score for the entire sample.

The Rubric-as-Reward baseline produces a semantically reasonable response, yet entirely neglects the capitalization constraint, yielding a naturally cased sentence with no formatting applied. This reveals a fundamental limitation of the decoupled paradigm: since the golden rubric only enters the training pipeline as a post-hoc reward signal, the model has no internal mechanism to actively track pending constraints during decoding, making it susceptible to constraint omission when multiple orthogonal requirements must be simultaneously satisfied. Notably, when employing vanilla thinking, the model does identify the relevant constraints during its thinking process and even performs an explicit self-check, yet still produces a non-compliant answer—incorrectly concluding that all constraints have been satisfied. This highlights that unstructured thinking alone is insufficient to bridge the gap between constraint recognition and constraint adherence, owing to the lack of self-consistency between the reasoning trace and the final response.

In contrast, THINK-WITH-RUBRICS first materializes all task requirements into an explicit rubric, correctly identifying all four hard constraints and two quality principles before generation begins. The self-consistency reward incorporated in THINK-WITH-RUBRICS effectively resolves the inconsistency between the model's reasoning process and its final generated response. The subsequent answer strictly adheres to every rubric item: all words are capitalized, the response contains only 16 words, and actionable recommendations are provided. This case demonstrates that by converting implicit instruction constraints into an explicit self-imposed checklist, THINK-WITH-RUBRICS effectively reduces the risk of constraint omission that commonly afflicts reward-only training approaches.

A.3 Training Details and Compute Resources

Our training pipeline consists of two stages: an SFT warm-up stage followed by RL fine-tuning. Table 10 lists the hyperparameters for the SFT stage, and Table 11 lists the hyperparameters for the RL stage.

SFT Stage. We fine-tune Qwen3-8B-Base using LLaMA-Factory [30] with DeepSpeed ZeRO-3. The training data consists of up to 5,000 samples distilled from Gemini-2.5-Flash. Training runs for 3 epochs with a cosine learning rate schedule and a warmup ratio of 0.05. The effective batch size is 32 (per-device batch size of 1 with gradient accumulation over 8 steps across 4 H20 GPUs).

Compare of Rubrics-as-Reward and Think-with-Rubrics

Question: I want to write a reflective essay on how my life has changed since I started college. Do you have any recommendation? Please reply in English and capitalize all your words. Control the length of your reply. I don't want anything longer than 30 words.

Answer of Rubric-as-Reward training:
Reflect on academic challenges, personal growth, independence, and new perspectives gained since starting college.

Answer of Rubric-as-Reward training (With normal thinking):
1. Analyze the Request: Topic: Reflective essay on life changes since starting college.
Tone: Academic, reflective.
Length: 30 words or less.
Format: English, capitalized, brief final answer.
2. Drafting Content:
Focus on personal growth, independence, and academic challenges.
Keep it concise.
3. Refining for Length:
College has transformed my life from a sheltered student into an independent adult. I have learned to manage my time, navigate complex social dynamics, and develop a deeper sense of self-awareness, all of which have fundamentally reshaped my perspective on the world.
4. Final Check:
Is it 30 words or less? Yes.
Is it in English? Yes.
Is it capitalized? Yes.
Is it brief? Yes.
<answer>College has transformed my life from a sheltered student into an independent adult. I have learned to manage my time, navigate complex social dynamics, and develop a deeper sense of self-awareness, all of which have fundamentally reshaped my perspective on the world.</answer>

Answer of Think-with-Rubrics:
<rubric>
1. The response must be in English. [Hard Rule]
2. All words in the response must be capitalized. [Hard Rule]
3. The response must be no longer than 30 words. [Hard Rule]
4. The response must provide recommendations for writing a reflective essay. [Hard Rule]
5. The response should be concise and to the point. [Principle]
6. The response should directly address the user's request. [Principle]
</rubric>

<answer>
START WITH A CLARIFYING QUESTION. IDENTIFY KEY CHANGES. REFLECT ON CAUSES AND IMPACTS. CONCLUDE WITH INSIGHTS.
</answer>

Figure 3: A test case in IFEval that compares simple Rubric-as-Reward and THINK-WITH-RUBRICS. Under the THINK-WITH-RUBRICS setting, the model generated accurate rubric and solved the problem of difficulty in following instructions accurately with simple Rubric-as-Reward.

RL Stage. We train the SFT-initialized model with DAPO (without KL loss) using the veRL framework [15]. The batch size is 32 prompts and we generate 8 rollouts per prompt, with a train mini-batch size of 16. Training runs for 120 steps with a cosine learning rate schedule and linear warmup over the first 10 steps. The clip ratios are set to $\epsilon_{\text{low}} = 0.2$ and $\epsilon_{\text{high}} = 0.28$, and the rollout temperature is 1.0.

Compute Resources. The SFT warm-up stage is conducted on 4 NVIDIA H20 GPUs. The RL fine-tuning stage is conducted on 8 NVIDIA H20 GPUs, with each complete RL training run finishing in approximately 16 hours.

A.4 Prompts

During inference, we employ two distinct prompt formats corresponding to the two paradigms evaluated in main result table. For the Rubric-as-Reward baseline with CoT reasoning (With think setting), we append a concise reasoning directive to the original instruction, asking the model to first think step by step and then enclose its final answer within `<answer>` tags. This format allows the

Table 10: Hyperparameters for the SFT warm-up stage.

Hyperparameter	Value
Framework	LLaMA-Factory
Parallelism	DeepSpeed ZeRO-3
Number of GPUs	$4 \times$ H20
Training epochs	3
Per-device batch size	1
Gradient accum. steps	8
Effective batch size	32
Learning rate	1×10^{-5}
LR scheduler	Cosine
Warmup ratio	0.05
Max sequence length	8,192
Max training samples	5,000
Precision	BF16

Table 11: Hyperparameters for the RL fine-tuning stage.

Hyperparameter	Value
Algorithm	DAPO (no KL)
Framework	veRL
Number of GPUs	$8 \times$ H20
Total training steps	120
Prompts per batch	32
Rollouts per prompt	8
Effective batch size	256
Mini-batch size	16
Learning rate	1×10^{-6}
LR scheduler	Cosine
Warmup steps	10
Clip ratio ϵ_{low}	0.2
Clip ratio ϵ_{high}	0.28
Rollout temperature	1.0

model to perform free-form reasoning prior to answering, while remaining consistent with the format used during its SFT warm-up stage. For THINK-WITH-RUBRICS, the prompt explicitly instructs the model to first construct a structured rubric comprising hard rules and quality principles, and then generate a response conditioned on that rubric, mirroring the two-stage generation paradigm enforced during training. Both formats are presented in Figures 4 and 5 respectively.

A.5 Broader Impacts

This work proposes THINK-WITH-RUBRICS, a training paradigm that improves the instruction-following capability and alignment of large language models. On the positive side, more reliably instruction-following models can benefit a wide range of downstream applications, including assistive writing tools, educational systems, and professional productivity software, by producing outputs that more faithfully respect user intent and constraints. Furthermore, the self-evolution potential demonstrated in this work suggests a path toward reducing dependence on costly human annotation, which could democratize the development of aligned AI systems for resource-constrained settings.

On the negative side, as with most work aimed at improving general instruction following, there is a possibility that enhanced compliance with user-specified constraints could be misused to generate content that more precisely satisfies undesirable requirements. We note, however, that this risk is not specific to our approach and is shared by the broader line of work on LLM alignment.

Rubric-as-Reward (With Think) Prompt Format

Instruction: {original instruction}

Please first think step by step and finally put your final answer (don't add any other output) between `<answer>` and `</answer>`.

Figure 4: The prompt format used for the Rubric-as-Reward baseline with CoT reasoning. The model is free to generate an unstructured reasoning trace before producing the final answer enclosed in `<answer>` tags.

Think-with-Rubrics Prompt Format

Here is the instruction:
<instruction>
{original instruction}
</instruction>

Please complete the task following these strict steps:

1. **Define Rubric**: Analyze the request and outline a rubric containing 'Hard Rules' (essential constraints) and 'Principles' (quality guidelines).
2. **Generate Answer**: Create a response to the request.

Format your output exactly as follows:

<rubric>
[Insert Rubric content here]
Format:
1. ..... [Hard Rule]
2. ..... [Principle]
</rubric>

<answer>
[Insert Answer here]
</answer>

Figure 5: The prompt format used for THINK-WITH-RUBRICS models. The model is explicitly required to first enumerate hard rules and quality principles as a structured rubric before generating the final answer, mirroring the <rubric> $\rightarrow$ <answer> generation paradigm enforced during training.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the three main contributions: the THINK-WITH-RUBRICS paradigm, the unified RL training workflow with joint supervision, and the mechanistic analyses. All claims are supported by experimental results reported in Section 4.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: A dedicated Limitations section is included in Appendix A.1, discussing the restricted scope of empirical validation to instruction-following tasks and the additional inference overhead introduced by explicit rubric generation.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: Section 4.1 provides full details of model initialization, data construction, data splits, training hyperparameters, verifier distillation procedure, reward weights, and evaluation protocols. Inference prompt templates are additionally provided in Appendix A.3.

Guidelines:

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

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: We will open-source our training data upon the conclusion of the review process; our code directly employs the LLaMA-Factory and veRL frameworks for training on the new dataset.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Section 4.1 specifies the base models, data splits (5k SFT / 4k verifier / 27k RL), optimizer (DAPO), learning rates, batch sizes, rollout counts, reward weights (α, β, γ, w_h, w_p), and evaluation protocols for all three benchmarks.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: Error bars are not reported due to the high computational cost of running multiple independent RL training runs with large language models. The consistent directional improvements across multiple benchmarks and model sizes (4B and 8B) provide qualitative evidence for the robustness of the results.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Appendix A.3 details the compute resources used in our experiments. The SFT warm-up stage is run on 4 NVIDIA H20 GPUs, and the RL fine-tuning stage is run on 8 NVIDIA H20 GPUs with each training run completing in approximately 16 hours.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research focuses on improving instruction-following alignment of language models and does not involve human subjects, sensitive data collection, or applications with foreseeable harmful use cases. The work conforms with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

Answer: [Yes]

Justification: A Broader Impacts section is included in the Appendix. Positive impacts include improved reliability of AI assistants and reduced dependence on human annotation. Potential negative impacts are also briefly discussed, noting that the associated risks are common to the broader field of LLM alignment research.

Guidelines:

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

11. **Safeguards**

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

Answer: [N/A]

Justification: The paper proposes a training methodology rather than releasing a new high-risk model or dataset. The training data are derived from the publicly available *Openrubrics* dataset, which does not pose elevated misuse risks.

Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: All datasets (Openrubrics, IFEval, IFBench, InfoBench, RewardBench-2) and models (Qwen3 series, Gemini-2.5-Flash) are properly cited in the paper. The *Openrubrics* dataset is publicly released for research use.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

Answer: [N/A]

Justification: The paper does not release new datasets or model checkpoints at submission time. The trained rubric verifier and policy models will be released upon acceptance.

Guidelines:

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

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

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

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects. All evaluations are conducted automatically using benchmark evaluation scripts and LLM-as-a-judge.

Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve human subjects research and therefore does not require IRB approval.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [Yes]

    Justification: LLM usage is central to the methodology and is thoroughly described. Specifically, Gemini-2.5-Flash is used for SFT data distillation and InfoBench evaluation; Qwen3-32B-Instruct serves as the teacher model for verifier distillation; and Qwen3-8B-Instruct is the base for the trained rubric verifier. All usages are detailed in Section 4.1.

    Guidelines:

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