BEYOND STATIC CRITERIA: MODEL–RUBRIC CO-EVOLUTION FOR REINFORCEMENT LEARNING WITH SELF-DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Recently, on-policy self-distillation (OPSD) and its reinforcement-learning extensions have offered a promising route to improving large language models. By conditioning the teacher on additional privileged information, these methods provide token-level guidance along the student's own sampled trajectories. However, the effectiveness of these methods depends on predefined privileged teacher information, giving rise to two key challenges. First, teacher contexts are often constructed independently of each sampled rollout, making them unable to adapt to its specific errors or unmet behaviors. Second, the teacher information is typically fixed before training and cannot evolve with the improving capabilities of the policy. To address these limitations, we propose **M**odel–**R**ubric Co-Evolution for **R**einforcement **L**earning with **S**elf-**D**istillation (MR.RLSD), an integrated reinforcement-learning and self-distillation framework centered on co-evolving rubrics. MR.RLSD uses rubrics as a shared interface between reward feedback and teacher guidance. Rubric scores determine policy-update directions, while criteria unmet by each response are transformed online into teacher contexts that provide contrastive feedback for modulating token-level update magnitudes, without requiring pre-constructed privileged contexts. During training, a shared policy serves as both a response generator and a rubric generator, proposing new criteria based on current rollouts and historical rubrics. The updated criteria jointly refine reward computation and teacher guidance, allowing both the optimization targets and self-distillation feedback to adapt to the evolving policy. Experiments across health, writing, and science show that MR.RLSD outperforms the evaluated reinforcement learning and self-distillation baselines, demonstrating the effectiveness of co-evolving the policy and its guiding criteria for open-ended generation.

1 INTRODUCTION

Large language models (LLMs) have advanced rapidly in reasoning and open-ended generation, with post-training methods such as supervised fine-tuning (SFT) and reinforcement learning with verifiable rewards (RLVR) playing a central role in further eliciting and improving their capabilities (Ouyang et al., 2022; Guo et al., 2025). Recently, on-policy self-distillation (OPSD) has emerged as an active direction for LLM post-training. OPSD instantiates the student and teacher using the same model: the student samples trajectories from the original query, while the teacher conditions on additional privileged information and provides token-level supervision along those same trajectories, allowing the model to learn from its own predictions under different information conditions (Zhao et al., 2026). Meanwhile, OPSD-based methods and insights can also be integrated with reinforcement learning, using environmental rewards to determine policy-update directions and privileged teacher signals to modulate token-level update magnitudes, thereby grounding self-distillation in reward feedback (Yang et al., 2026; Pan et al., 2026).

Nevertheless, the effectiveness of these methods remains closely tied to the privileged information available to the teacher. Existing self-distillation methods largely rely on predefined privileged information, such as reference answers or correct reasoning traces. Some works also sample incorrect rollouts as contrasting references. However, because such information is constructed in advance, it cannot provide targeted guidance: For a given rollout, the predefined teacher information cannot

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

adapt to its actual content and provide guidance tailored to its specific deficiencies. More importantly, preconstructed privileged information remains static rather than evolving alongside the policy. Recent work has also observed that vanilla OPSD can reach its best performance early in training and subsequently degrade in certain settings, suggesting that its initial gains do not necessarily translate into sustained improvement (Yang et al., 2026). Beyond effectively using existing teacher feedback, we therefore ask: *can the guidance carried by teacher feedback evolve together with the policy?*

Rubrics provide a natural interface for addressing this question. For open-ended tasks without deterministic answers, rubrics describe desirable response properties through explicit, fine-grained natural-language criteria and can be aggregated into reward signals for reinforcement learning (Gunjal et al., 2026). Meanwhile, rubrics not only evaluate responses but also naturally encode actionable guidance for producing better ones. In particular, a criterion that the current response fails to satisfy identifies a concrete deficiency and a corresponding direction for improvement, and can therefore be transformed into conditioning information for the teacher (Xia et al., 2026). Moreover, rubrics can self-evolve throughout training (Guan et al., 2026; Ding et al., 2026). This motivates us to dynamically evolve rubrics, and thereby explicitly evolve the guidance available to the teacher, so that self-distillation can continually address requirements that the current policy does not satisfy.

Motivated by this idea, we propose **M**odel–**R**ubric Co-Evolution for **R**einforcement **L**earning with **S**elf-**D**istillation (MR.RLSD). The central idea is to use a shared set of evolving rubrics to jointly organize response-level reinforcement learning and token-level self-distillation, translating these criteria into teacher guidance online without requiring pre-constructed privileged contexts. Specifically, the current rubrics first evaluate a group of on-policy rollouts to produce response-level rewards. The criteria that each response fails to satisfy are then transformed into corrective and inverted contexts, under which the teacher contrastively evaluates the student-sampled trajectory to derive token-level self-distillation signals. The response-level reward determines the direction of each token update, while the rubric-conditioned teacher discrepancy modulates its magnitude. In parallel, the model proposes and validates new criteria based on current rollouts and historical rubric memory. The accepted rubrics are then used in the next iteration for both reward computation and teacher-context construction. Deficiencies exposed by the policy thus drive rubric updates, while the updated rubrics provide new guidance for both reward feedback and self-distillation, establishing a training loop in which the policy and its guiding knowledge evolve together.

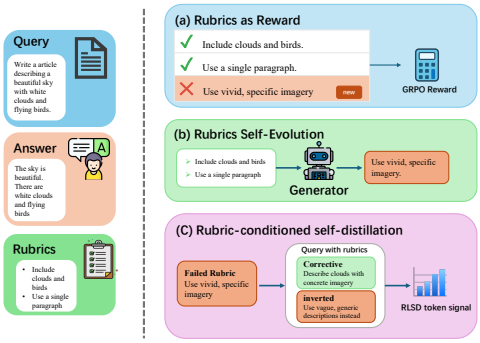


Figure 1: Rubrics serve as a bridge between reward construction, knowledge evolution, and on-policy self-distillation. (a) Translate open-ended quality criteria into GRPO rewards with positive and negative feedback. (b) Enable the rubricator to propose new criteria from current responses, allowing the rubricknowledge to evolve during training. (c) Failed criteria are converted into corrective and inverted contexts that provide rubric-conditioned token-level guidance for RLSD.

We conduct systematic training and evaluation across multiple domains, including Health, Writing, and Science. MR.RLSD achieves the best overall performance among the compared methods with both Qwen3-8B and Qwen3-4B-Instruct, outperforming the strongest baseline by an average of 1.6% and 1.2%, respectively. These results demonstrate the effectiveness of the proposed framework across diverse open-ended generation domains.

Our contributions are as follows:

- We propose **MR.RLSD**, a framework for reinforcement learning with on-policy self-distillation through model–rubric co-evolution. For each rollout, MR.RLSD identifies the failed rubric criteria and rewrites them into corrective and inverted contexts, providing contrastive teacher guidance tailored to the specific deficiencies of that response without requiring pre-constructed privileged contexts. Meanwhile, the rubricator continuously proposes new criteria from current rollouts, and the evolved rubric pool jointly updates response-

level rewards and teacher conditioning, allowing the guidance to adapt as the policy improves.

- We validate MR.RLSD through training and evaluation across Health, Writing, and Science. Our method achieves leading performance under the evaluated settings, demonstrating its applicability across diverse open-ended generation domains.

2 Related Work

2.1 On-Policy Self-Distillation

Recent work has increasingly explored on-policy distillation (OPD) for language model post-training. Unlike SFT on fixed demonstrations, OPD (Agarwal et al., 2024) provides teacher supervision along student-sampled trajectories, bringing distillation closer to the student's own generation distribution. OPSD (Zhao et al., 2026) further instantiates the teacher and student using the same model, providing token-level guidance along student-generated trajectories by conditioning the teacher on additional privileged information. Moreover, recent work has begun exploring OPSD for rubric-based post-training in open-ended tasks (Gu et al., 2026; Rezaei et al., 2026). However, unlike MR.RLSD, the rubrics in these works do not co-evolve with the model during training.

Meanwhile, the OPSD can also be integrated with reinforcement learning. RLSD (Yang et al., 2026) uses environmental rewards to determine the direction of policy updates and exploits the probability difference between the privileged teacher and the student to modulate token-level update magnitudes. RLCSD (Pan et al., 2026) further contrasts teacher feedback under different hinting conditions to reduce privilege-induced style drift and to make the distillation signal more concentrated on task-relevant tokens.

2.2 Rubric-based Reinforcement Learning

Rubric-based reinforcement learning converts the multidimensional quality judgments required by open-ended tasks into structured signals for policy optimization. RaR(Gunjal et al., 2026) evaluates responses using instance-specific natural-language criteria and aggregates criterion-level scores into reinforcement learning rewards, extending RLVR to medical, scientific, and other domains without deterministic correctness signals. In addition, CriPO (Xia et al., 2026) transforms rubric criteria into self-distillation contexts to recover unexplored behaviors and behaviors that may be suppressed by scalar reward aggregation.

Recent work has also explored the self-evolution of rubrics. RLCER (Sheng et al., 2026) uses self-proposed and self-evolving rubrics to reward chain-of-thought reasoning, treating rubric satisfaction as a process-level supervision signal while jointly optimizing the reasoner and rubricator. EvoRubric (Guan et al., 2026) allows a single model to alternate between response generation and rubric generation, enabling it to discover and validate new evaluation criteria from responses produced by the current policy.

3 Method

3.1 Preliminaries

Given a query x, the rollout policy $\pi_{\theta_{\text{old}}}$ samples a group of G responses, denoted by $\mathcal{Y}_x = \{y_1, \ldots, y_G\}$. Group Relative Policy Optimization (GRPO) (Shao et al., 2024) computes the sequence-level advantage of each response based on its relative reward within the rollout group and optimizes the following clipped surrogate objective:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|y_i|}\sum_{t=1}^{|y_i|}\min\left(\rho_{i,t}A_i,\ \text{clip}\left(\rho_{i,t}, 1-\epsilon, 1+\epsilon\right)A_i\right)\right], \tag{1}$$

where$\rho_{i,t} = \frac{\pi_\theta(y_{i,t}|x, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t}|x, y_{i,<t})}$ is the token-level importance ratio between the current and rollout policies.

For open-ended generation tasks, response quality is typically multidimensional and context dependent. A single outcome verifier or an undifferentiated holistic score may therefore fail to capture all quality dimensions relevant to policy optimization. We represent these requirements using a rubric set $\mathcal{R}_x = \{r_j\}_{j=1}^{K}$, where $r_j = (d_j, w_j)$ consists of a natural-language criterion d_j and a signed weight w_j. Positive weights correspond to desired behaviors, whereas negative weights correspond to undesirable behaviors (Gunjal et al., 2026).

For a response y_i, a positive rubric contributes its weight when the desired behavior is satisfied, while a negative rubric contributes its negative weight when the undesirable behavior is triggered. Let $s_{i,j}$ denote the resulting criterion-level score. The rubric-based reward is defined as

$$r_i^{\text{rub}} = \frac{\sum_{j=1}^{K} s_{i,j}}{W_x^+}, \qquad W_x^+ = \max\left(\sum_{j=1}^{K} \max(w_j, 0), \epsilon_W\right), \tag{2}$$

where ϵ_W is a numerical stability constant. This normalization uses the maximum achievable positive score to maintain a comparable reward scale across rubric sets while retaining negative penalties. The resulting rewards are normalized within the rollout group to obtain the sequence-level advantages A_i, which are then used in Equation 1.

3.2 Rubric-Grounded Reinforcement Learning with Self-Distillation

Recent on-policy self-distillation (OPSD) methods have attracted increasing attention in the post-training of large language models (Zhao et al., 2026). These methods perform distillation along trajectories sampled from the current policy. The student and teacher can be instantiated using the same underlying model, while the teacher additionally observes privileged information, such as a reference answer, a reasoning trace, environmental feedback, or other information unavailable during standard inference. RLSD (Yang et al., 2026) incorporates this self-distillation mechanism into reinforcement learning. Rather than directly forcing the student to match the privileged teacher distribution, RLSD uses the teacher-student likelihood difference as a token-level weighting signal. The sequence-level advantage determines the update direction of the rollout, while the likelihood difference induced by privileged information modulates the update magnitude assigned to individual tokens. For a rollout y_i, RLSD defines

$$\begin{aligned}
\Delta_{i,t} &= \text{sg}\left[\log \pi_{\theta_T}\left(y_{i,t} \mid x, c_i, y_{i,<t}\right) - \log \pi_\theta\left(y_{i,t} \mid x, y_{i,<t}\right)\right], \\
w_{i,t} &= \exp\left(\text{sign}(A_i)\Delta_{i,t}\right), \\
\widehat{A}_{i,t} &= A_i\left[(1-\lambda) + \lambda \,\text{clip}\left(w_{i,t}, 1-\epsilon_w, 1+\epsilon_w\right)\right],
\end{aligned} \tag{3}$$

where c_i denotes the privileged context, π_{θ_T} is the teacher policy, $\text{sg}[\cdot]$ denotes the stop-gradient operator, λ controls the strength of token-level reweighting, and ϵ_w bounds the weight variation for an individual token. The teacher branch is evaluated with stop-gradient and therefore provides a token-level weighting signal rather than a separate distribution-matching objective.

Existing self-distillation methods often rely on privileged teacher contexts c_i that are constructed before, or independently of, the student rollout. Such contexts may encode reference answers or verified reasoning traces, but cannot directly adapt to the specific errors or unmet behaviors in the sampled response. This limitation is particularly pronounced in open-ended question answering, where multiple responses may be valid and no unique reference reasoning trace is available. Rubrics provide a natural alternative: as fine-grained evaluation criteria, they specify the behaviors that a response should satisfy or avoid. By identifying the criteria unmet by each rollout, rubric feedback can be transformed into rollout-specific teacher contexts that provide targeted guidance.

For each response y_i, we identify its failed rubrics. A positive rubric is considered failed when its desired requirement is not satisfied, whereas a negative rubric is considered failed when the undesirable behavior it describes is triggered. We then rewrite these failed rubrics into corrective instructions for the current response. The prompt template is shown as follows:

Failed rubrics rewrite prompt

Question: `[Original query]`

Guidance dimensions:

```
- Ensure the response satisfies this requirement:  [Unmet positive rubric]
- Avoid this undesirable behavior:  [Triggered negative rubric]
- ...
Use the above guidance when reasoning and composing the final answer.
```

For an unmet positive rubric, the corrective instruction asks the response to satisfy the corresponding desired behavior. For a triggered negative rubric, it asks the response to avoid the identified undesirable behavior. The resulting instructions form the rollout-specific corrective context c_i^{fix}.

Moreover, additional context may nevertheless alter teacher token probabilities for reasons unrelated to response quality, including changes in formatting, phrasing, or general instruction style. To reduce these task-irrelevant effects, we follow the contrastive construction of RLCSD (Pan et al., 2026) and introduce an inverted context for the same target rollout. The corrective and inverted contexts are derived from the same set of failed rubrics; the inverted context only reverses the required behavioral directions. For an unmet positive rubric, the inverted instruction asks the response to omit or violate the originally desired behavior. For a triggered negative rubric, it asks the response to retain or exhibit the originally undesirable behavior.

We evaluate the same student-sampled rollout under the corrective and inverted teacher contexts through teacher forcing. Neither branch generates a new response. The rubric-conditioned likelihood difference is defined as

$$\Delta_{i,t}^{\text{rub}} = \text{sg}\left[\log \pi_{\theta_T}\left(y_{i,t} \mid x, c_i^{\text{fix}}, y_{i,<t}\right) - \log \pi_{\theta_T}\left(y_{i,t} \mid x, c_i^{\text{inv}}, y_{i,<t}\right)\right]. \tag{4}$$

Because the two contexts are derived from the same failed rubrics and use matched structures, generic formatting, style, and phrasing effects shared by the two branches are partially cancelled through subtraction. The resulting difference is therefore more closely associated with the behavioral directions specified by the failed rubrics.

We replace $\Delta_{i,t}$ in Equation 3 with $\Delta_{i,t}^{\text{rub}}$ and compute

$$w_{i,t}^{\text{rub}} = \exp\left(\text{sign}(A_i)\Delta_{i,t}^{\text{rub}}\right), \tag{5}$$

$$\widehat{A}_{i,t}^{\text{rub}} = A_i\left[(1-\lambda) + \lambda\,\text{clip}\left(w_{i,t}^{\text{rub}}, 1-\epsilon_w, 1+\epsilon_w\right)\right]. \tag{6}$$

Thus, the aggregated rubric scores determine the sequence-level reward and update direction, while the corresponding failed rubrics are transformed into rollout-specific teacher contexts that refine how the reward is allocated across token positions.

3.3 Co-Evolving Rubrics for RLSD

Rubric-derived contexts provide targeted guidance for individual rollouts. However, when they are constructed from a fixed rubric pool, the knowledge available to the teacher remains bounded by the evaluation dimensions encoded in that pool. As the policy adapts to the existing criteria, those criteria may reveal fewer new errors and quality differences, causing both the reward and the teacher feedback to become less discriminative.

Therefore, we allow the rubric set to evolve together with the policy. Following EvoRubric, we use the policy model as a rubric generator, enabling it to propose new rubrics. The rubricator receives the query x, the current rollout group $\mathcal{Y}_x$, and historical rubrics $\mathcal{R}_{x,\text{hist}}$ retrieved from the existing rubric pool. It then generates M candidate rubric sets:

$$R_m \sim \pi_\theta\left(\cdot \mid x, \mathcal{Y}_x, \mathcal{R}_{x,\text{hist}}, p_{\text{rub}}\right), \qquad m = 1, \ldots, M, \tag{7}$$

where p_{rub} is the role prompt that activates the rubricator. The candidate rubrics aim to describe quality dimensions that are insufficiently covered by the current pool or to distinguish quality differences within the current rollout group.

After generating new rubrics, we further filter them to ensure that only valid and informative rubrics are used in the RLSD stage. The filtering results also provide reward signals for training the rubricator. Specifically, we apply the following two-stage filtering procedure:

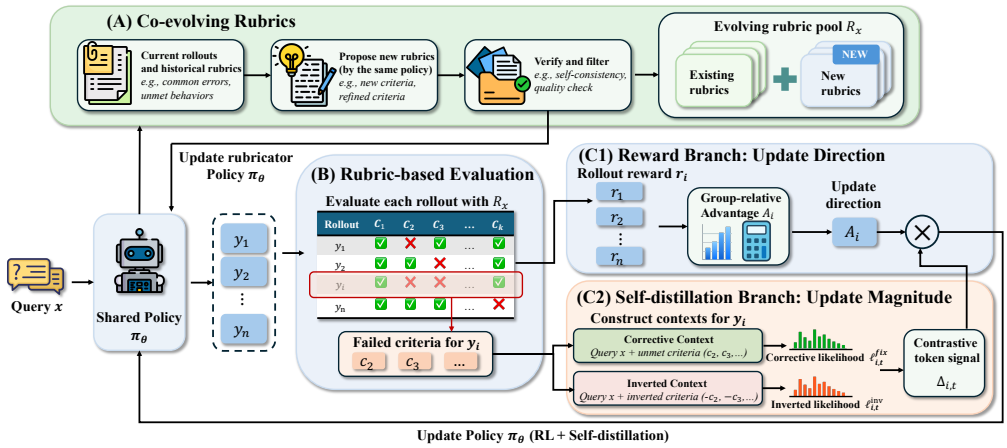


Figure 2: **Overview of MR.RLSD framework.** The shared policy generates a group of rollouts and, under the rubricator prompt, proposes new rubric criteria based on the current rollouts and historical rubrics. After verification and discriminativeness filtering, the updated rubric pool is used to compute rollout rewards and identify failed criteria. These failed rubrics are rewritten into corrective and inverted privileged contexts to obtain the contrastive signal $\Delta_{i,t}^{\mathrm{rub}}$. Rubric rewards determine the update direction, while the contrastive signal modulates the token-level update magnitude; the reasoner and rubricator are then jointly optimized.

1. **Verifier filtering.** We first employ a verifier to assess the validity of each generated rubric according to a set of predefined criteria. Specifically, a valid rubric should (i) follow the required format, (ii) be relevant to the current query and response, (iii) contain no logical contradictions or unsupported assumptions, and (iv) specify a clear and operational criterion that can be consistently evaluated. Rubrics that satisfy all criteria are retained for the subsequent stage, while those that fail verification are discarded. We denote the penalty associated with verification failure as $p_{\mathrm{verify}}(R_m)$, which is used as a negative reward signal for the rubricator.
2. **Discriminativeness filtering.** We then apply each rubric that passes verification to all responses in the current rollout group. A rubric is considered informative only if it induces meaningful score variation across the group. We discard a rubric if it assigns the same score to all responses or if its score differs from the common score for only a single response. Such rubrics provide either no group-relative information or an overly sparse signal, and therefore offer limited supervision for relative policy optimization. Only rubrics that distinguish multiple responses within the rollout group are retained for the RLSD stage.

Let $\widehat{\mathcal{R}}_x$ denote the candidates that pass all filtering stages. The rubric pool is updated as

$$\widehat{\mathcal{R}}_x = \mathrm{Filter}\left(\{R_m\}_{m=1}^{M}, \mathcal{Y}_x\right), \tag{8}$$

$$\mathcal{R}_{x,new} = \mathrm{Update}\left(\mathcal{R}_x, \widehat{\mathcal{R}}_x\right), \tag{9}$$

where Update adds validated criteria to the existing pool while removing duplicate or discarded criteria.

The updated rubric pool affects both levels of supervision. First, the validated rubrics participate in the reward calculation in Equation 2, thereby updating the quality dimensions optimized by GRPO. Second, when a rollout fails a newly evolved rubric, the criterion is incorporated into both its corrective and inverted contexts. It consequently contributes to the discrepancy in Equation 4 and the token-level weighting in Equation 6. Meanwhile, in the next epoch, the newly added rubrics are also used as base rubrics for the rubricator, enabling it to generate more diverse rubric criteria.

Moreover, to enable the rubricator to co-evolve with the policy, we also update its parameters. For each generated rubric set R_m, we compute a rubricator reward $r_{\mathrm{rub}}(R_m)$.

$$R_{\mathrm{rub}}(R_m) = \frac{\mathrm{num}(\widehat{\mathcal{R}}_x)}{\mathrm{num}(\mathcal{R}_m)} - p_{\mathrm{verify}}(R_m) - p_{\mathrm{size}}(R_m) - p_{\mathrm{len}}(R_m), \tag{10}$$

where num denotes the number of generated rubrics and the number of rubrics that pass the filtering process, $p_{\mathrm{size}}(R_m)$ penalizes excessively large candidate sets and $p_{\mathrm{len}}(R_m)$ penalizes overly long or redundant rubric descriptions. We normalize this reward within the candidate rubric group and update the rubricator using GRPO.

3.4 Overall Objective

Putting these components together, we obtain **M**odel–**R**ubric Co-Evolution for **R**einforcement **L**earning with **S**elf-**D**istillation (**MR.RLSD**), as illustrated in Figure 2. The reasoner and rubricator share a parameterized policy π_θ under different role prompts.

At each iteration, the reasoner first generates a group of rollouts. Conditioned on these rollouts and historical rubrics, the rubricator proposes new criteria, which are validated and filtered before being incorporated into the rubric pool. The updated pool is then used to score the rollouts and identify their failed criteria. These failed rubrics are rewritten into corrective and inverted teacher contexts, under which the same rollouts are evaluated to obtain contrastive token-level self-distillation signals. Rubric scores determine the response-level update directions, while the contrastive signals modulate token-level update magnitudes. Meanwhile, rubricator rewards guide the policy toward generating more effective criteria.

Let $\mathcal{J}_{\mathrm{rea}}(\theta)$ denote the reasoner's GRPO objective with rubric-conditioned token-level advantages, and let $\mathcal{J}_{\mathrm{rub}}(\theta)$ denote the rubricator's GRPO objective with group-normalized rubricator rewards. We optimize the shared policy using

$$\boxed{\mathcal{J}_{\mathrm{overall}}(\theta) = \mathcal{J}_{\mathrm{rea}}(\theta) + \eta\, \mathcal{J}_{\mathrm{rub}}(\theta)} \tag{11}$$

where η controls the contribution of rubricator optimization. This joint training process couples policy improvement with rubric evolution, continually updating both reward criteria and teacher guidance. Detailed formulations of the two objectives are provided in Appendix B.

4 Experiments

4.1 Experimental Setup

Datasets and Benchmarks To ensure comparability with prior work, our experimental protocol largely follows the setup in EvoRubric. We conduct our experiments across three domains: Medical, Writing, and Science. For training, we randomly sample 3,000 queries each from **HealthBench**(Medical) (Arora et al., 2025), **Long-writer-zero**(Writing) (Wu et al., 2026a), and **ResearchQA**(Science) (Yifei et al., 2026). We use mutually disjoint training, validation, and test queries, and exclude validation/test queries and their associated rubrics from policy optimization and rubric evolution. For evaluation, we evaluate two benchmarks in each domain: (1) **Medical**: **HealthBench** (Arora et al., 2025) and **LLMMed-Eval** (Zhang et al., 2025) benchmark; (2) **Writing**: **WritingBench** (Wu et al., 2026b) and **Creative Writing** (Paech, 2025) benchmark; (3) **Science**: **ResearchQA** (Yifei et al., 2026) and **GPQA-Diamond** (Rein et al., 2023) benchmark. We evaluate each benchmark twice and report the average score.

Models and Implementation Details We choose two base models Qwen3-8B(non-thinking) (Yang et al., 2025), Qwen3-4B-Instruct-2507 (Yang et al., 2025). We implement and train all methods using the verl (Sheng et al., 2024) framework. The full hyperparameter settings are provided in Appendix D.

Baselines We compare MR.RLSD against the following methods. **Vanilla**: the original base model without RL fine-tuning; **GRPO** (Shao et al., 2024): the standard group relative policy optimization algorithm; **RLCSD** (Pan et al., 2026): Reinforcement Learning with Contrastive On-Policy Self-Distillation. We use rubrics to construct contrastive privileged contexts for open-ended tasks; the remaining implementation follows ours.**Evorubric** (Guan et al., 2026): A co-evolutionary rubric-based RL method in which the model generates new rubrics and uses them as additional reward criteria. All the aforementioned methods are reproduced under identical settings on the same dataset, ensuring a consistent experimental environment.

Model	Method	Health		Writing		Science		Avg.
		HealthBench	LLMMED	WritingBench	CreativeWriting	ResearchQA	GPQA-D	
Qwen3-8B	Vanilla	38.51	61.92	67.25	39.86	69.06	52.02	54.77
	GRPO	48.28	65.07	72.88	40.30	76.44	53.54	59.42
	RLCSD	49.54	64.82	74.38	41.05	78.63	53.54	60.33
	Evorubric	51.02	64.47	**75.36**	41.84	80.14	53.54	61.06
	MR.RLSD	**51.76**	**68.07**	75.02	**42.99**	**81.16**	**57.07**	**62.68**
Qwen3-4B-Ins	Vanilla	45.51	65.91	75.71	36.23	70.45	43.43	56.21
	GRPO	51.06	68.20	78.81	37.91	75.90	50.00	60.31
	RLCSD	51.49	67.51	78.89	38.23	78.84	**51.01**	61.00
	EvoRubric	51.44	68.97	79.63	39.44	80.46	48.99	61.49
	MR.RLSD	**53.06**	**71.51**	**79.86**	**40.27**	**80.99**	50.51	**62.70**

Table 1: **Main results of our methods in health, writing and science benchmarks.** The best results are highlighted in **bold** and the second best results are highlighted in underline. The MR.RLSD method achieves state-of-the-art performance on Qwen3-8B, Qwen3-4B-Instruct, outperforming the baseline by an average of 1.62% and 1.21%, respectively.

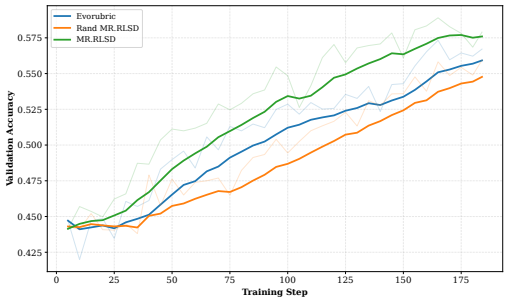


Figure 3: Random sampled rubrics for MR.RLSD may hurt performance.

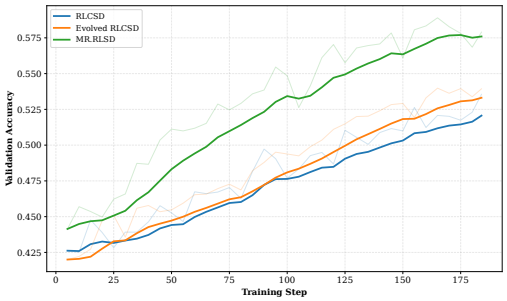


Figure 4: Evolving rubrics solely for privileged information can also improve performance.

4.2 Experiment Results

Our main experimental results are presented in Table 1. Our method, MR.RLSD, achieves the highest average score across the six benchmarks for both models. On Qwen3-8B, MR.RLSD achieves the best performance on five of the six benchmarks, with an average score of 62.68, exceeding GRPO, RLCSD, and EvoRubric by 3.26, 2.35, and 1.62 points, respectively. On Qwen3-4B-Instruct, MR.RLSD also leads on five of the six benchmarks, with an average score of 62.70, exceeding GRPO, RLCSD, and EvoRubric by 2.39, 1.70, and 1.21 points, respectively. These results support the effectiveness of combining rollout-specific teacher guidance with model-rubric co-evolution for improving performance across diverse domains.

4.3 Effectiveness of Failed-Rubric Guidance

As discussed in Section 3.2, using the rubrics failed by each rollout as teacher guidance enables more targeted feedback for addressing the specific deficiencies of the response. To verify this effect, we replace the failed-rubric selection in MR.RLSD with random rubric selection and compare it with EvoRubric and MR.RLSD. The validation performance throughout training is shown in Figure 3.

The random-selection variant of MR.RLSD performs even worse than EvoRubric, whereas our method achieves sustained improvements on the validation set. This result shows that constructing targeted teacher guidance from failed rubrics is important: randomly selected rubrics may provide irrelevant guidance and consequently hurt performance.

4.4 Effectiveness of Evolving Rubrics for Teacher Guidance

In Section 3.3, we use evolved rubrics for both reward computation and privileged context construction. To examine whether their benefits extend beyond reward expansion, we conduct an ablation in which newly generated rubrics are used only to construct teacher contexts and are excluded from reward computation. We compare this variant with the original RLCSD, with results shown in Figure 4.

Using evolved rubrics only for teacher-context construction still improves RLCSD, indicating that rubric evolution provides additional teacher knowledge beyond reward expansion and strengthens RLSD training.

4.5 WHY DO WE NEED CONTRASTIVE CONTEXT?

In Section 3.2, we point out that a single-sided rubric context may introduce task-irrelevant token differences into the distillation signal, thereby affecting policy optimization. To verify this observation, we remove the contrastive mechanism and train the model using only a single-sided rubric context, while recording the resulting average Δ and its standard deviation.

As shown in Table 2, removing the contrastive mechanism results in a negative mean Δ and a relatively large standard deviation, indicating that the distillation signal is substantially affected by task-irrelevant token differences. After introducing contrastive contexts, the mean Δ increases while its standard deviation decreases accordingly. This suggests that contrastive construction suppresses shared stylistic and contextual effects, yielding a more stable signal that focuses on task-relevant behaviors. The improved performance on HealthBench further supports this observation.

Method	**Avg. Δ**	**Std. Δ**	**HealthBench**
w/o contrast	-0.2137	0.1406	49.18
MR.RLSD	$+0.7070$	0.0789	51.76

Table 2: Effect of contrastive privileged contexts on the distillation signal and HealthBench performance.

4.6 EFFECTIVENESS OF RUBRICS AS PRIVILEGED INFORMATION

The effectiveness of MR.RLSD relies on converting rubrics into privileged information that can elicit higher-quality teacher outputs. To verify this effect, we conduct an inference-time study that examines whether adding rubric prompts leads to better responses for the same query. We consider three input configurations: (i) Q, which contains only the query;(ii) $Q + R$, which contains the query and the original rubrics; and (iii) $Q + R + G$, which contains the query, the original rubrics, and additional rubrics generated by the rubricator. The construction of these inputs follows the same procedure as the training process described in Section 3.2. For each query, we compare the outputs generated under these configurations in a pairwise manner and use GPT-OSS-120B (Agarwal et al., 2025) as the evaluator.



(a) Healthbench (b) Researchqa

Figure 5: Pairwise win-tie-loss comparisons. **Q**: only query. **Q+R:** query+base rubrics. **Q+R+G:** query + base rubrics + rubricator generated rubrics.

As shown in the Figure 5, adding rubric prompts substantially improves the output quality compared with using the query alone. Moreover, further incorporating the rubrics generated by the rubricator yields additional improvements. These results demonstrate not only that rubrics can effectively serve as privileged hints, but also that the rubrics generated by the rubricator provide useful and sufficiently reliable guidance.

5 CONCLUSION

In this work, we presented **M**odel-**R**ubric Co-Evolution for**R**einforcement **L**earning with **S**elf-**D**istillation (**MR.RLSD**), a framework that couples rubric-based reinforcement learning with on-policy self-distillation. MR.RLSD identifies the failed criteria of each rollout and transforms them into corrective and inverted contexts, providing contrastive teacher guidance tailored to the sampled response. Meanwhile, a rubricator continuously proposes and validates new criteria, which jointly update the reward signal and teacher conditioning. This allows the policy and its guiding knowledge to evolve together without requiring pre-constructed privileged contexts. Experiments across Health, Writing, and Science show that MR.RLSD achieves the best overall performance among the compared methods with both Qwen3-8B and Qwen3-4B-Instruct. Our work offers a new perspective on self-evolving self-distillation methods.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, volume 2024, pp. 21246–21263, 2024.

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Rahul K Arora, Jason Wei, Rebecca Soskin Hicks, Preston Bowman, Joaquin Quiñonero-Candela, Foivos Tsimpourlas, Michael Sharman, Meghan Shah, Andrea Vallone, Alex Beutel, et al. Healthbench: Evaluating large language models towards improved human health. *arXiv preprint arXiv:2505.08775*, 2025.

Hongxin Ding, Baixiang Huang, Yue Fang, Weibin Liao, Zheng Li, Jinyang Zhang, Zhijing Wu, Junfeng Zhao, and Yasha Wang. Evorubrics: Dynamic rubrics as rewards via adversarial co-evolution for llm reinforcement learning. *arXiv preprint arXiv:2606.23038*, 2026.

Siyi Gu, Jialin Chen, Sophia Zhou, Arman Cohan, and Rex Ying. Rethinking reward supervision: Rubric-conditioned self-distillation. *arXiv preprint arXiv:2606.19327*, 2026.

Xin Guan, Xiaomeng Hu, Shen Huang, Zhenyi Wang, Bo Zhang, Zijian Li, Pengjun Xie, Bo Liu, and Jiuxin Cao. Evorubric: Self-evolving rubric-driven rl for open-ended generation. *arXiv preprint arXiv:2605.29847*, 2026.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. In *International Conference on Learning Representations*, volume 2026, pp. 127924–127945, 2026.

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.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

Samuel J Paech. Eq-bench creative writing benchmark v3. `https://github.com/EQ-bench/creative-writing-bench`, 2025.

Leyi Pan, Shuchang Tao, Yunpeng Zhai, Lingzhe Zhang, Zhaoyang Liu, Bolin Ding, Aiwei Liu, and Lijie Wen. Rlcsd: Reinforcement learning with contrastive on-policy self-distillation. *arXiv preprint arXiv:2606.11709*, 2026.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

MohammadHossein Rezaei, Anas Mahmoud, Zihao Wang, Utkarsh Tyagi, Advait Gosai, Razvan-Gabriel Dumitru, Aakash Sabharwal, Bing Liu, and Yunzhong He. Rubric-guided self-distillation: Post-training without rubric verifiers. *arXiv preprint arXiv:2606.12507*, 2026.

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.

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. *arXiv preprint arXiv: 2409.19256*, 2024.

Leheng Sheng, Wenchang Ma, Ruixin Hong, Xiang Wang, An Zhang, and Tat-Seng Chua. Reinforcing chain-of-thought reasoning with self-evolving rubrics. *arXiv preprint arXiv:2602.10885*, 2026.

Yuhao Wu, Yushi Bai, Zhiqiang Hu, Roy Ka-Wei Lee, and Juanzi Li. Longwriter-zero: Mastering ultra-long text generation via reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 144867–144892, 2026a.

Yuning Wu, Jiahao Mei, Ming Yan, Chenliang Li, Shaopeng Lai, Yuran Ren, Zijia Wang, Ji Zhang, Mengyue Wu, Qin Jin, et al. Writingbench: A comprehensive benchmark for generative writing. *Advances in Neural Information Processing Systems*, 38, 2026b.

Mingxuan Xia, Yuhang Yang, Chao Ye, Shuai Zhu, Shenzhi Yang, Guangcheng Zhu, Yuhang Zhang, Cheng Peng, Haobo Wang, and Siqing Wang. Enhancing rubric-based rl via self-distillation. *arXiv preprint arXiv:2607.18082*, 2026.

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.

Chenxu Yang, Chuanyu Qin, Qingyi Si, Minghui Chen, Naibin Gu, Dingyu Yao, Zheng Lin, Weiping Wang, Jiaqi Wang, and Nan Duan. Self-distilled rlvr. *arXiv preprint arXiv:2604.03128*, 2026.

Li S Yifei, Allen Chang, Chaitanya Malaviya, and Mark Yatskar. Researchqa: Evaluating scholarly question answering at scale across 75 fields with survey-mined questions and rubrics. *Transactions of the Association for Computational Linguistics*, 14:1344–1368, 2026.

Ming Zhang, Yujiong Shen, Zelin Li, Huayu Sha, Binze Hu, Yuhui Wang, Chenhao Huang, Shichun Liu, Jingqi Tong, Changhao Jiang, et al. Llmeval-med: A real-world clinical benchmark for medical llms with physician validation. *arXiv preprint arXiv:2506.04078*, 2025.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

A LLM USAGE STATEMENT

This paper used large language models to assist with literature search and the identification of relevant prior work, and to polish the language and improve the clarity of the manuscript. The role of LLMs is limited to enhancing the clarity and readability of the manuscript, including optimizing sentence structure and correcting errors in linguistic expression. No part of the core research methodology, data analysis, or theoretical contributions was generated or influenced by LLMs. The authors verified the cited sources and reviewed all AI-assisted edits, and take full responsibility for the accuracy and integrity of the final manuscript.

B OPTIMIZATION OBJECTIVES

As described in Section 3.4, MR.RLSD couples reasoner optimization with rubricator evolution through a shared parameterized policy. Following the clipped GRPO formulation, we optimize the reasoner and rubricator with separate role-specific objectives, whose detailed definitions are given below. For the reasoner, the rubric-conditioned token-level advantage $\widehat{A}^{\mathrm{rub}}_{i,t}$ is obtained by replacing the standard RLSD discrepancy with the contrastive rubric discrepancy $\Delta^{\mathrm{rub}}_{i,t}$ defined in Equation 4. The reasoner objective is

$$\mathcal{J}_{\mathrm{rea}}(\theta) = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|y_i|}\sum_{t=1}^{|y_i|}\min\left(\rho^{\mathrm{rea}}_{i,t}\widehat{A}^{\mathrm{rub}}_{i,t}, \mathrm{clip}\left(\rho^{\mathrm{rea}}_{i,t}, 1-\epsilon, 1+\epsilon\right)\widehat{A}^{\mathrm{rub}}_{i,t}\right)\right], \tag{12}$$

where

$$\rho^{\mathrm{rea}}_{i,t} = \frac{\pi_\theta\left(y_{i,t} \mid x, y_{i,<t}\right)}{\pi_{\theta_{\mathrm{old}}}\left(y_{i,t} \mid x, y_{i,<t}\right)}. \tag{13}$$

Algorithm 1 Model-Rubric Co-Evolution for RL with Self-Distillation(MR.RLSD)

Input: Training queries x, shared policy π_θ, initial rubric pools $\{\mathcal{R}_x^{(0)}\}$, group sizes G and M, hyperparmeters ϵ, ϵ_w, η
Output: Trained policy π_θ.

for each training iteration k **do**
  Set $\theta_{\text{old}} \leftarrow \theta$ and sample a query x.
  Generate G reasoner rollouts$\mathcal{Y}_x^{(k)} = \{y_i\}_{i=1}^{G}$using $\pi_{\theta_{\text{old}}}$.
  Generate M candidate rubric sets via query x, rollouts $\mathcal{Y}_x^{(k)}$, and current rubrics $\mathcal{R}_x^{(k)}$: $R_m^{(k)} \sim \pi_\theta\left(\cdot \mid x, \mathcal{Y}_x, \mathcal{R}_{x,\text{hist}}, p_{\text{rub}}\right)$
  Filter candidate rubrics $R_m^{(k)}$ and update the pool to $\mathcal{R}_x^{(k+1)}$.
  Compute rubricator rewards $R_{\text{rub}}(R_m^{(k)})$ and group-normalized advantages A_m^{gen}.
  Evaluate the rollouts using $\mathcal{R}_x^{(k+1)}$ to compute reasoner rewards r_i^{rub} and advantages A_i.
  Rewrite the failed rubrics of each $y_i$into corrective and inverted contexts c_i^{fix} and c_i^{inv}.
  Compute $\Delta_{i,t}^{\text{rub}} = \text{sg}\left[\log \pi_{\theta_T}\left(y_{i,t} \mid x, c_i^{\text{fix}}, y_{i,<t}\right) - \log \pi_{\theta_T}\left(y_{i,t} \mid x, c_i^{\text{inv}}, y_{i,<t}\right)\right]$.
  Compute token weights $w_{i,t}^{\text{rub}} = \exp\left(\text{sign}(A_i)\Delta_{i,t}^{\text{rub}}\right)$
  Modulate advantages via $w_{i,t}$: $\widehat{A}_{i,t}^{\text{rub}} = A_i\left[(1-\lambda) + \lambda\,\text{clip}\left(w_{i,t}^{\text{rub}}, 1-\epsilon_w, 1+\epsilon_w\right)\right]$.
  $\mathcal{J}_{\text{rea}}(\theta) = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|y_i|}\sum_{t=1}^{|y_i|}\min\left(\rho_{i,t}^{\text{rea}}\widehat{A}_{i,t}^{\text{rub}},\ \text{clip}\left(\rho_{i,t}^{\text{rea}}, 1-\epsilon, 1+\epsilon\right)\widehat{A}_{i,t}^{\text{rub}}\right)\right]$
  $\mathcal{J}_{\text{rub}}(\theta) = \mathbb{E}\left[\frac{1}{M}\sum_{m=1}^{M}\frac{1}{|R_m|}\sum_{t=1}^{|R_m|}\min\left(\rho_{m,t}^{\text{rub}}A_m^{\text{gen}},\ \text{clip}\left(\rho_{m,t}^{\text{rub}}, 1-\epsilon, 1+\epsilon\right)A_m^{\text{gen}}\right)\right]$
  Update the shared parameters θ using $\mathcal{J}_{\text{overall}} = \mathcal{J}_{\text{rea}} + \eta\,\mathcal{J}_{\text{rub}}$.
end for
return Trained policy π_θ and updated rubric pools.

For the rubricator, let A_m^{gen} denote the normalized advantage obtained from the rubricator reward $R_{\text{rub}}(R_m)$. Its objective is

$$\mathcal{J}_{\text{rub}}(\theta) = \mathbb{E}\left[\frac{1}{M}\sum_{m=1}^{M}\frac{1}{|R_m|}\sum_{t=1}^{|R_m|}\min\left(\rho_{m,t}^{\text{rub}}A_m^{\text{gen}}, \text{clip}\left(\rho_{m,t}^{\text{rub}}, 1-\epsilon, 1+\epsilon\right)A_m^{\text{gen}}\right)\right], \tag{14}$$

where

$$\rho_{m,t}^{\text{rub}} = \frac{\pi_\theta\left(R_{m,t} \mid x, \mathcal{Y}_x, \mathcal{R}_{x,\text{hist}}, p_{\text{rub}}, R_{m,<t}\right)}{\pi_{\theta_{\text{old}}}\left(R_{m,t} \mid x, \mathcal{Y}_x, \mathcal{R}_{x,\text{hist}}, p_{\text{rub}}, R_{m,<t}\right)}. \tag{15}$$

Here, θ_{old} denotes the rollout policy, $|y_i|$ is the length of response y_i, and $|R_m|$ is the token length of the serialized rubric set. The shared policy is optimized with the combined objective

$$\mathcal{J}_{\text{overall}}(\theta) = \mathcal{J}_{\text{rea}}(\theta) + \eta\,\mathcal{J}_{\text{rub}}(\theta), \tag{16}$$

where η controls the contribution of rubricator optimization. Thus, the reasoner is updated using rubric-based rewards and contrastive teacher signals, while the rubricator is updated according to the quality of the rubrics it generates.

C ALGORITHM

The overall procedure of our algorithm is presented in Algorithm 1.

D EXPERIMENTAL SETUP

The detailed hyperparameter settings are shown in Table 3. Our training is conducted on 8 Nvidia H20 GPUs, with full parameter optimization and gradient checkpointing.

Parameter	Value
LLM Judger	`GPT-OSS-120B`
Rubricator Verifier	`GPT-OSS-120B`
Reasoner Rollout Samples per Prompt (G)	8
GRPO Clip Bounds (ϵ)	0.2
KL Loss Coefficient	0.0
Max Prompt Length	16384
Max Response Length	8,192
Optimizer	AdamW
Learning Rate	1×10^{-6}
Learning Rate Schedule	Constant
Training Batch Size	64
Mini Batch Size	64
Micro Batch Size Per GPU	16
Total Epochs	4
Reasoner/rubricator coefficient η	1.0
RLCSD initial λ	0.3
RLCSD λ decay step	160
RLCSD λ min decay	0.0
RLCSD ϵ_w	0.1

Table 3: Experimental setup

Benchmark	HealthBench	llmmed-eval	WritingBench	Creative Writing	ResearchQA
Judge Model	`gpt-4o`	`gpt-4o`	`claude-sonnet-4.5`	`claude-sonnet-4.5`	`gpt-4.1`

Table 4: Judge models used for benchmark evaluation.

E Evaluation setup and Robustness to Different Judges

Most of our evaluation benchmarks are rubric-based and require an external judge model. The judge model used for each benchmark is listed in Table 4. We evaluate each benchmark twice and report the average score to reduce the effects of randomness.

Moreover, To examine whether our conclusions depend on a specific evaluator, on ResearchQA, we evaluate performance using alternative judge models, including `gpt-oss-120b` and Claude Opus 4.6. As shown in Table 6, MR.RLSD consistently achieves the best performance across these judges, supporting the robustness of its improvements to the choice of judge model.

F Ablation Study for Fixed Rubricator

In MR.RLSD, the reasoner is optimized not only with rubric-based rewards, but also jointly with the rubricator to enable their co-evolution. To evaluate the effect of rubricator training, we freeze the rubricator parameters and use it only to generate rubrics, while keeping the remaining training procedure unchanged. The results are shown in Table 5.

Table 5: Ablation study for fixed rubricator

Method	Healthbench	LLMMed
Evorubric	51.02	64.47
MR.RLSD	51.76	68.07
fix rubricator	50.05	66.18

Freezing the rubricator leads to a substantial performance drop compared with jointly training both components. Continuously updating the rubricator helps maintain the quality of the generated rubrics, which in turn provides more effective reward and teacher guidance for training the reasoner.

Model	Judge Model	Method				
		Vanilla	GRPO	RLCSD	Evorubric	MR.RLSD
Qwen3-8B	GPT-4.1	69.06	76.44	78.63	80.14	81.16
	Opus-4.6	56.04	65.83	66.34	67.67	69.02
	GPT-OSS-120b	67.55	74.93	74.80	76.98	77.30
Qwen3-4B-Ins	GPT-4.1	70.45	75.90	78.84	80.46	80.99
	Opus-4.6	58.55	63.02	64.29	66.47	66.62
	GPT-OSS-120b	68.85	73.97	73.80	75.80	77.77

Table 6: Different judge model for researchqa.

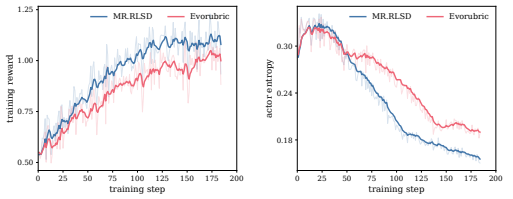


Figure 6: Training Dynamics in Healthbench for MR.RLSD and Evorubric

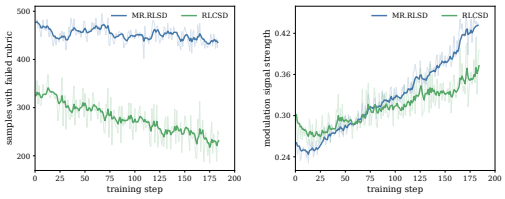


Figure 7: Training Dynamics in Healthbench for MR.RLSD and RLCSD

G Training Dynamics Analysis for MR.RLSD

We further analyze the training dynamics in the health domain, as shown in Figures 6 and 7. Figure 6 compares the reward and entropy trajectories of EvoRubric and MR.RLSD. MR.RLSD exhibits a faster increase in reward during the middle stage of training, supporting the advantage of jointly evolving rubric-guided rewards and teacher feedback over evolving only the rubric-based reward space.

Figure 7 compares the RLSD-related training statistics of MR.RLSD and RLCSD. Without rubric evolution, the number of rollouts containing failed rubrics gradually decreases. Once a rollout contains no failed rubrics, it cannot provide a contrastive rubric signal and effectively degenerates to standard GRPO. In contrast, MR.RLSD continuously introduces new evaluation directions through rubric evolution, maintaining rollouts with failed rubrics and supplying ongoing token-level modulation signals for the teacher. This sustained guidance continually supports policy optimization throughout training.

H Prompt template for our experiments

In this section, we provide detailed prompt templates used in various stages of our experiments. These templates ensure the consistency and reproducibility of our experimental setup. We present the prompt templates used for the ResearchQA dataset. For other datasets, the overall structure and logic remain consistent, with only minor differences in specific details.

Figure 8 presents the rubricator's system prompt. Given the query, existing rubrics, and model rollouts, the rubricator uses this prompt to generate new rubrics. Figure 9 presents the system prompt for the reward judge model. The external judge model uses this prompt to determine whether each response satisfies the corresponding rubric. Figure 10 presents the rubric verifier's prompt. The verifier uses it to assess whether each generated rubric meets the required standards and filters out invalid rubrics.

System prompt for rubricator

You are a specialist in designing reliable rubrics for ResearchQA scientific question answering.
Analyze the complete question, candidate answers, and Existing Rubrics. Generate only new, discriminative criteria for important answer qualities that are not already covered.
Prioritize: answering the exact question and all requested subparts; factual and logical correctness; distinctions between established evidence, plausible interpretation, and uncertainty; relevance and scope; appropriate use of quantitative details and sources when available; and clear, reproducible reasoning. Penalize fabricated studies, citations, numbers, or claims of certainty, as well as answers that evade the question or confuse correlation with causation. Do not require citations when the question does not ask for them or when no source can be verified. Do not reward length, jargon, or a larger list of entities by itself.
Each criterion must be observable in the answer, generalize beyond one sampled response, and not duplicate an Existing Rubric. Positive points describe a concrete desirable behavior; negative points describe an active error, not merely an absence of polish. Return [] when no new meaningful distinction remains.
Return ONLY a JSON list and no analysis or markdown:
[{ "criterion": "<observable ResearchQA quality or error>", "points": <integer from -10 to 10> }]

Figure 8: System prompt for rubricator

System prompt for reward judge model

You are a strict ResearchQA evaluator. Given the complete scientific question, a generated response, and quality rubrics, evaluate the response against EACH rubric using observable evidence.
For each rubric, mark "PRESENT" when its condition is satisfied or its described error is actually present, and "NOT_PRESENT" otherwise; point polarity does not reverse this meaning. Evaluate the exact question and all requested subparts, factual and logical correctness, relevance, scope, evidence support, quantitative details, and calibrated uncertainty when relevant. Do not infer quality from length, jargon, confidence, or citation count. Do not reward fabricated citations, studies, numbers, mechanisms, or unsupported certainty, and do not require citations unless requested or verifiable. A rubric that says "such as", "for example", or "including" is non-exhaustive.
<Prompt>
{complete research question}
</Prompt>
<Response>
{generated response}
</Response>
<Rubrics>
{rubric list with point values}
</Rubrics>

Figure 9: System prompt for reward judge model

System prompt for rubricator verifier

You are an expert ResearchQA rubric reviewer. Evaluate whether the Target Rubric is valid for the exact scientific question and non-redundant with Existing Rubrics.
Reject only clear: factual or logical conflict with the question; unmeasurable subjective claims; semantic duplication; generic advice; rewarding length or jargon alone; or scope creep unrelated to answering the question. A rubric may require factual correctness, relevant evidence, explicit reasoning, appropriate uncertainty, and source use when the question requests or supports it. Do not require unverifiable citations or invented studies, numbers, or mechanisms. A criterion detecting an answer that evades the question or presents speculation as established fact is valid. When uncertain, return VALID.
Return only one JSON object: {"verdict":"VALID"} or {"verdict":"INVALID: <Category>"}.

Figure 10: System prompt for rubricator verifier