Rethinking Reward Supervision: Rubric-Conditioned Self-Distillation

Siyi Gu, Jialin Chen, Sophia Zhou, Arman Cohan[†] & Rex Ying[†]
Yale University
{siyi.gu,arman.cohan,rex.ying}@yale.edu
[†]Equal advising

Abstract

Post-training of reasoning language models is commonly driven by supervised distillation and reinforcement learning with verifiable rewards. Distillation often relies on chain-of-thought annotations that are expensive to obtain and may themselves be noisy, incomplete, or partially incorrect; even when the final solution is correct, an imperfect rationale can interfere with learning. Reinforcement learning with verified rewards, on the other hand, typically compresses evaluative feedback into a scalar signal, obscuring which aspects of a response should be improved. We propose **Rubric-Conditioned Self-Distillation**, a framework that incorporates rubrics as structured, fine-grained feedback for on-policy self-distillation. Our method conditions the teacher model on criterion-level rubrics and uses it to provide token-level guidance on the student's own sampled trajectories. This design avoids treating a single reference rationale as the sole supervision target. Instead, rubrics specify what a strong response should satisfy, enabling more fine-grained credit assignment over the reasoning process than scalar reward optimization. We instantiate this framework with a two-stage pipeline that first learns to generate task-specific rubrics and then trains a rubric-guided reasoner. We evaluate on a diverse suite of science reasoning benchmarks and results show that rubric-conditioned self-distillation effectively converts rubric-level criteria into token-level guidance over the reasoning process, surpassing GRPO by 1.0 points and OPSD by 0.9 points on average. Code available: https://github.com/carriegu0818/RCSD.

1 Introduction

Recent advances in large language models (LLMs) have led to substantial progress in reasoning, problem-solving, and instruction following. Reinforcement learning has been particularly effective in domains such as mathematics and code generation, where final outcomes can be automatically verified. However, Group Relative Policy Optimization (GRPO) objective typically optimizes sparse outcome-level rewards: the model is rewarded only after generating a complete response, for example, based on whether the final answer is correct or whether execution succeeds (Shao et al., 2024). While effective in verifiable settings, such supervision provides little information about *why* a trajectory succeeds or fails, creating a persistent credit-assignment bottleneck in online learning (Hübotter et al., 2026; Zhao et al., 2026).

One natural way to enrich this supervision is through *rubrics*. Rather than scoring a response holistically, rubrics decompose quality into explicit criteria, yielding a more structured and interpretable representation of what makes an answer strong (Gunjal et al., 2025; Zhang et al., 2025a). Recent work has shown that rubric-based evaluation can extend post-training beyond strictly verifiable tasks by supplying richer judgments than binary correctness alone (Gunjal et al., 2025). Yet in most existing works, rubric information enters training only through the *reward*: criterion-level judgments are aggregated into a single scalar score

and then optimized with RL-style updates on the entire trajectory. As a result, the rich information inside such textual rubric feedback is largely discarded during optimization.

A recent line of work addresses sparse outcome rewards by replacing them with dense teacher supervision. In on-policy distillation (OPD) and on-policy self-distillation (OPSD), the student learns from its own sampled trajectories while a teacher provides token-level guidance along those rollouts (Agarwal et al., 2024; Xu et al., 2024; Zhao et al., 2026; Hübotter et al., 2026; Ye et al., 2026). These methods alleviate the mismatch of off-policy imitation and provide a denser learning signal than final-answer rewards. However, existing approaches typically construct the teacher from stronger-model outputs. Such supervision is therefore tied to particular privileged trajectories, which may not cleanly expose the underlying dimensions along which a response should be evaluated. These trajectories represent only one valid reasoning trace rather than the underlying dimensions that define a strong response. In this sense, rationale-based supervision can over-specify *how* an answer should be produced, without cleanly identifying *what properties* the answer should satisfy.

In this work, we introduce ***R**ubric-**C**onditioned **S**elf-**D**istillation* (RCSD), a post-training framework that uses rubrics as privileged teacher-side supervision for on-policy self-distillation. Our key idea is that rubrics should not only score responses after generation; they should shape token-level learning during optimization. Instead of collapsing rubric feedback into a scalar reward, we condition the teacher on criterion-level rubric information and distill its token-level guidance on the student's own sampled trajectories. The resulting training signal is simultaneously *criterion-aware*, *on-policy*, and *token-level*: it preserves distinctions across evaluation dimensions, operates on student-generated rollouts rather than fixed off-policy traces, and provides dense guidance without reducing feedback to a single number. Figure 1 illustrates the optimization signal difference on the same incorrect trajectory: RL assigns one reward to the full sequence, OPSD provides dense supervision toward a reference trajectory, and RCSD provides dense rubric-conditioned feedback that preserves correct steps while penalizing the specific local error.

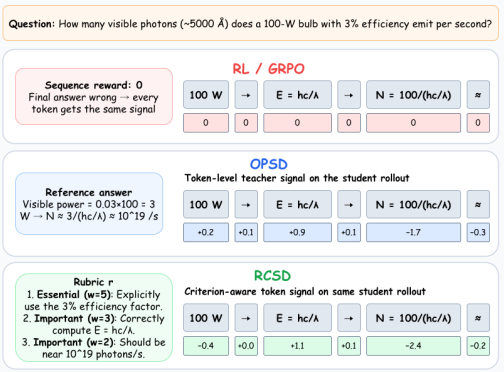

Figure 1: Illustration of how optimization signals differ between RL/OPSD/RCSD with incorrect student trajectory.

We operationalize our idea with a two-stage pipeline. We first train a rubric generator to amortize instance-specific evaluation criteria from privileged supervision, and then train a reasoner with rubric-conditioned teacher guidance. More broadly, we reframe rubrics as a structured supervision interface for model self-improvement, especially in hard-to-verify and open-ended tasks where high-quality responses are not fully captured by automatic verification or scalar outcome rewards. Across diverse reasoning benchmarks, RCSD achieves the best overall average (70.6), surpassing GRPO by 1.4 points and OPSD by 0.9 points. Notably, the gains are pronounced on scientific and rubric-based reasoning tasks, where response quality is poorly captured by scalar outcome-level rewards alone.

2 Method

We propose to preserve the fine-grained, structured feedback during optimization by using learned rubrics as privileged teacher-side supervision for on-policy self-distillation. Rather than compressing rubric feedback into a single number, we expose it to a privileged teacher, which then provides dense token-level guidance on the student's own sampled trajectories. Figure 2 situates our method relative to two standard alternatives. Reinforcement learning applies outcome-level supervision through a sparse scalar reward. On-policy self-distillation replaces this with token-level teacher guidance, but typically conditions the teacher on a privileged reference answer. In contrast, our method redefines this supervision interface:

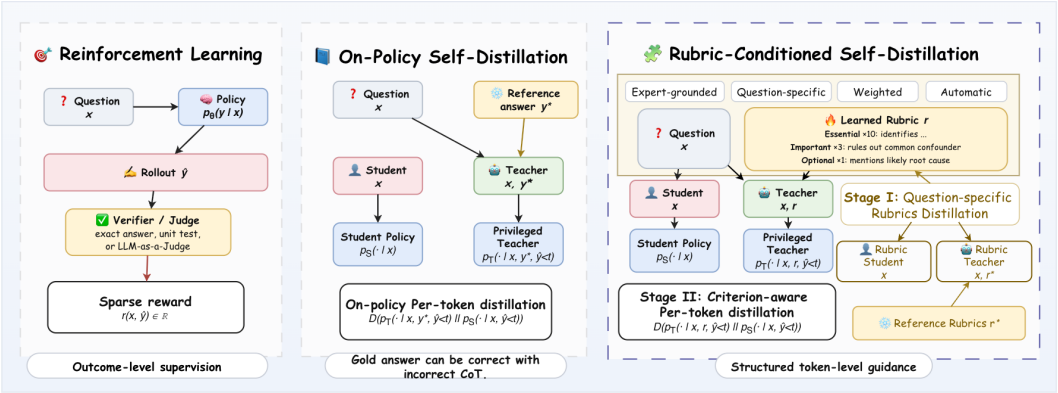

Figure 2: RCSD uses rubrics as privileged teacher-side supervision for on-policy self-distillation. In contrast to RL which compresses feedback into a scalar reward, and OPSD which conditions the teacher on a reference answer, RCSD learns question-specific rubrics in Stage I and reuses them in Stage II to induce structured token-level guidance on the student's own reasoning trajectory.

instead of conditioning the privileged teacher on a single reference trajectory, we condition it on a rubric that specifies criterion-level properties of a strong response.

2.1 Preliminaries

We use p_T and p_S to denote the teacher and student distributions, respectively.

Off-Policy Distillation (Hinton et al., 2015) trains a student to imitate trajectories generated by a teacher. In its most general form, the objective can be written as

$$\mathcal{L}_{\text{off}} = \mathbb{E}_{x \sim \mathcal{D}, y \sim p_T(\cdot|x)} \left[\sum_{t=1}^{|y|} D(p_T(\cdot \mid x, y_{<t}) \,\|\, p_S(\cdot \mid x, y_{<t})) \right], \tag{1}$$

where $D(\cdot \| \cdot)$ denotes a divergence between teacher and student sequence distributions. Off-policy distillation provides dense token-level supervision, but it suffers from a distribution mismatch: the student is trained on teacher-generated prefixes, whereas at inference time it must condition on its own generated prefixes, leading to compounding errors and degraded performance.

On-Policy Distillation (OPD) (Agarwal et al., 2024; Gu et al., 2023) addresses this mismatch by sampling trajectories from the student rather than the teacher. Given an input x, the student first generates an on-policy rollout $\hat{y} \sim p_S(\cdot \mid x)$. The teacher and student are then compared along the student's own trajectory, yielding the objective

$$\mathcal{L}_{\text{OPD}} = \mathbb{E}_{x \sim \mathcal{D},\, \hat{y} \sim p_S(\cdot|x)} \left[\frac{1}{|\hat{y}|} \sum_{t=1}^{|\hat{y}|} D(p_T(\cdot \mid x, \hat{y}_{<t}) \,\|\, p_S(\cdot \mid x, \hat{y}_{<t})) \right]. \tag{2}$$

However, the on-policy distillation method still heavily relies on token-level imitation of a teacher distribution, which often encourages the student to follow a single preferred response, ignoring the space of valid reasoning paths.

On-Policy Self-Distillation (OPSD) (Zhao et al., 2026; Hübotter et al., 2026) refers to the setting in which the teacher and student are derived from the same underlying model rather than from two separately trained models. In the on-policy self-distillation setting, a single model instantiates both a student policy and a privileged teacher policy. Given a reasoning dataset $S = \{(x, z)\}$, where z denotes privileged information such as a gold solution, a reference answer, or other side information, the student observes only the base input x and generates an on-policy response $\hat{y} \sim p_S(\cdot \mid x)$, while the teacher is conditioned on

privileged information z unavailable to the student at inference time. The OPSD objective is

$$\mathcal{L}_{\text{OPSD}} = \mathbb{E}_{(x,z)\sim\mathcal{S},\, \hat{y}\sim p_S(\cdot|x)} \left[\frac{1}{|\hat{y}|} \sum_{t=1}^{|\hat{y}|} D(p_T(\cdot \mid x, z, \hat{y}_{<t}) \,\|\, p_S(\cdot \mid x, \hat{y}_{<t})) \right]. \tag{3}$$

While OPSD further introduces privileged information to guide learning, it suffers from conditioning the teacher on a specific reference solution contained in z. This can be restrictive for reasoning tasks, where solution quality is better characterized by satisfying a set of criteria rather than a sole target. These limitations motivate a more flexible supervision interface that provides structured, multi-dimensional criterion-level guidance.

2.2 Motivation: Beyond Reward Optimization and Reference-Conditioned Distillation

We position RCSD against two common paradigms for improving reasoning models: reward-based optimization and reference-conditioned distillation.

(1) Reward optimization requires sparse external judgment. GRPO has been highly effective in verifiable domains, where correctness can be directly checked by exact-match answers or unit tests (Shao et al., 2024; Guo et al., 2025; Chollet et al., 2025; Jain et al., 2024). Recent work extends this paradigm to hard-to-verify or non-verifiable domains by using LLM-as-a-Judge rewards (Li et al., 2026; Gunjal et al., 2025). However, this extension still reduces supervision to sparse scalar reward signals, which provide limited information about which intermediate reasoning steps should be improved. It also introduces an additional external evaluator, increasing inference and training cost, and may further amplify reward bias from the judge model through its preferences, calibration errors, or inconsistent interpretation of rubrics. In contrast, RCSD does not require a separate reward model or judge during distillation. Instead, we provide rubrics directly to the teacher model and let the teacher generate rubric-conditioned reasoning on its own outputs. This turns rubrics into a structured supervision interface, allowing the student to learn from dense token-level teacher guidance rather than optimizing against sparse scalar reward signals.

(2) Reference-conditioned distillation is path-specific. OPSD conditions the teacher on a single reference trajectory, which can induce path-specific supervision. When the student deviates from this trajectory, even slightly, the teacher signal may encourage global revision rather than localized correction. Empirically, we observe that OPSD trajectories often recompute the same intermediate quantities or revise earlier steps without new information, leading to long and redundant reasoning chains. This suggests that OPSD provides token-level supervision, but lacks explicit criterion-level credit assignment. Our proposed method, RCSD, addresses this limitation by conditioning the teacher on rubric criteria instead of a single reference path, yielding supervision that is both on-policy and criterion-aware.

2.3 Problem Setup

Let x denote an input question. We consider two structured outputs associated with x: a rubric r and a answer y. A rubric is a structured set of question-specific evaluation criteria, $r = \{c_1, \ldots, c_K\}$, where each criterion c_k contains a title, a natural language description, and an importance weight $\in$ *Essential*, *Important*, *Optional*, *or Pitfall*. Conceptually, a rubric serves as an intermediate supervision interface that specifies what constitutes a good solution and provide richer and more interpretable information on how to guide the model optimization. A high-quality rubric encourages the model to focus on satisfying high-level criteria rather than imitating a specific reasoning path.

Our goal is to learn a model that produces high-quality reasoning trajectories under instance-specific evaluation criteria. Such criteria are useful during training, but are unavailable to the student at inference time. We therefore treat rubrics as *privileged supervision*: they are provided to the teacher during training and distilled into the student through on-policy token-level guidance. Since distilling high-quality rubrics could be costly, we automate the rubric generation process and further factorize learning into two sequential stages:

1. **Stage I: Learning a rubric generator.** We train a model to predict a rubric r conditioned on the question x, amortizing instance-specific evaluation criteria into a reusable form.
2. **Stage II: Rubric-conditioned reasoning.** We train a reasoner to generate trajectories that satisfy rubric criteria, using the rubric as structured guidance during optimization.

2.4 Stage I: Learning a Rubric Generator

A practical challenge in rubric-based training is that high-quality instance-specific rubrics are expensive to obtain. Our first stage therefore amortizes privileged supervision into a rubric generator. During training, the teacher is allowed to view both the question and a reference answer, while the student must learn to infer an appropriate rubric.

Student policy. The student rubric generator observes only the question: $p_S^R(r \mid x)$.

Teacher policy. The teacher rubric generator receives privileged access to the question and reference answer $y^\star$: $p_T^R(r \mid x, y^\star)$. The reference answer is not used as a supervision target to be copied directly; rather, it serves as a privileged context that helps the teacher infer how to generate a correct response trajectory within its own distribution.

On-policy rubric distillation. Given a sampled rubric rollout $\hat{r} \sim p_S^R(\cdot \mid x)$, we train the student to match the teacher's next-token distribution along the student's own rubric trajectory:

$$\mathcal{L}_{\text{rubric}} = \mathbb{E}_{\hat{r} \sim p_S^R(\cdot \mid x)} \left[\sum_{t=1}^{|\hat{r}|} D_{\text{KL}} \Big(p_T^R(\cdot \mid x, y^\star, \hat{r}_{<t}) \parallel p_S^R(\cdot \mid x, \hat{r}_{<t}) \Big) \right]. \tag{4}$$

This objective preserves the key advantage of on-policy self-distillation: the student is trained on its own sampled prefixes rather than teacher-generated ones. At the same time, the teacher can inject privileged information from the reference answer to shape the rubric-generation process. As a result, Stage I distills evaluation structure into a rubric generator that can produce question-specific criteria without requiring privileged inputs at test time.

2.5 Stage II: Rubric-Conditioned Reasoning

Given a rubric, the second stage trains a reasoner to generate answers that better satisfy instance-specific criteria. The key design choice is that the rubric is not merely appended as auxiliary prompt text for the student. Instead, it is provided as a privileged context to the teacher, which uses it to deliver criterion-aware token-level guidance on the student's own rollout.

Student policy. The student reasoner observes only the question: $p_S^Y(y \mid x)$.

Teacher policy. The teacher reasoner receives the question together with rubric feedback: $p_T^Y(y \mid x, r)$. Here r denotes the learned rubric from the Stage I rubric generator. Conditioning the teacher on r allows the training signal to reflect multiple dimensions of response quality, rather than a single scalar reward or a single reference trajectory.

On-policy rubric-conditioned distillation. Given an on-policy student rollout $\hat{y} \sim p_S^Y(\cdot \mid x)$, we minimize

$$\mathcal{L}_{\text{reason}} = \mathbb{E}_{\hat{y} \sim p_S^Y(\cdot \mid x)} \left[\sum_{t=1}^{|\hat{y}|} D_{\text{KL}} \Big(p_T^Y(\cdot \mid x, r, \hat{y}_{<t}) \parallel p_S^Y(\cdot \mid x, \hat{y}_{<t}) \Big) \right]. \tag{5}$$

This objective highlights the core advantage of on-policy distillation—dense supervision on student-generated trajectories. Rather than learning from a reference rationale or a

scalarized reward, the student is guided by a teacher conditioned on criterion-level rubric information. Its key importance is that criterion-level structure is retained in the teacher signal itself, so optimization can distinguish different dimensions of partial correctness rather than compressing them into a single undifferentiated score.

Reference-conditioned distillation supervises the student with one specific solution trajectory, which can introduce path-specific bias when multiple derivations are valid. Rubrics instead specify the criteria that a correct solution must satisfy, inducing a criterion-aware teacher distribution over many valid reasoning paths rather than a single reference path. This distinction naturally aligns with forward KL distillation. Minimizing $D_{\mathrm{KL}}(p_T(\cdot \mid x, r) \,\|\, p_S(\cdot \mid x))$ encourages the student to cover the support of the rubric-conditioned teacher distribution, preserving probability mass on alternative solutions that satisfy the same criteria. In contrast, more mode-seeking objectives may concentrate on dominant teacher modes and underrepresent valid but less likely derivations. Thus, forward KL provides a principled objective for transferring dense, criterion-level guidance while maintaining diversity across valid reasoning paths.

RCSD introduces a new supervision interface for post-training, by using rubrics as privileged structured guidance for on-policy self-distillation. Overall, Stage I learns *what* a strong response should satisfy, while Stage II learns *how* to realize those criteria along the student's own trajectory. Prior rubric-based RL typically uses rubrics only for outcome-level scoring, while standard OPSD preserves token-level optimization but ties supervision to a single privileged answer. RCSD differs from both by letting fine-grained rubric feedback directly guide on-policy learning.

3 Experiments

Our experiments are designed to answer the following questions: (1) whether rubric-guided token-level feedback improves both verifiable and open-ended reasoning performance compared with scalar-reward and reference-conditioned distillation baselines; (2) whether the resulting gains persist across model scales and remain robust on out-of-domain benchmarks; and (3) whether learned rubrics can approach the effectiveness of reference rubrics, and how sensitive RCSD is to rubric quality.

Data Construction. We follow the two-stage design described in Section A to construct rubric generation data and reasoning data. For rubric learning, we construct our training set based on RaR-Science (Gunjal et al., 2025) and RubricHub (Li et al., 2026) for reference rubric supervision. For reasoner training, we additionally take `natural_reasoning` (Yuan et al., 2025) and filter out entries with empty reference answers. The final dataset contains approximately 10k samples for rubric generation and 30k samples for reasoning generation.

Evaluation. We evaluate on a diverse set of science reasoning benchmarks: 1) Verifiabl: GPQA-Diamon (Rein et al., 2024), SciBench (Wang et al., 2023a), PIQA (Bisk et al., 2020), ResearchQA(Yifei et al., 2026), 2) non-verifiable: RaR-Science (Gunjal et al., 2025) and RubricHub (Li et al., 2026). For open-ended scientific tasks, we use gpt-4.1-mini as LLM-as-a-Judge to evaluate following their paper setup and take a 500 subset from the test set. To assess out-of-domain generalization, we further report results on medical question answering benchmarks, including MedMCQA (Pal et al., 2022) and PubMedQA (Jin et al., 2019). We follow the Language Model Open Science Evaluation framework for science reasoning.[1].

Baselines. We compare the following methods: *(1) supervised fine-tuning*, which trains the student on the distilled CoT trajectories, *(2) GRPO* (Shao et al., 2024), where the reward is a scalar reward produced by an LLM judge, (3) *Rubric-GRPO*, where the reward is aggregated by prompting LLM-as-a-Judge to assign specific rubric rewards, (4) *On-Policy Self-Distillation (Zhao et al., 2026)*, where teacher conditions on reference answers and provides dense token-level supervision on student trajectories.

[1] https://github.com/GAIR-NLP/lm-open-science-evaluation

Method	GPQA-D	SciBench	PIQA	RaR	ResearchQA	RubricHub	Avg
Qwen3-8B	60.6	69.1	90.2	59.7	64.9	50.8	65.9
+SFT	62.0	65.4	89.2	61.5	63.2	51.2	65.4
+GRPO	63.5	69.7	90.3	68.2	71.5	51.9	69.2
+GRPO-Rubrics	62.1	70.2	90.3	**69.9**	72.1	52.9	69.6
+OPSD	63.6	68.7	90.1	68.5	72.8	54.5	69.7
+RCSD (Ours)	**64.5**	**70.8**	**90.8**	68.6	**73.1**	**55.7**	**70.6**

Table 1: Main results on diverse reasoning benchmarks. Qwen3-8B is used as the backbone model. GRPO is trained for 500 steps while self-distillation based methods are trained for 100 steps, following prior work's implementation. Higher is better for all metrics.

Loss Type	GPQA-D	SciBench	PIQA	RaR	ResearchQA	RubricHub	Avg
Forward KL	**64.5**	**70.6**	**90.8**	68.6	**73.1**	55.7	**70.6**
JSD	62.1	69.2	89.9	67.1	**73.1**	**56.2**	69.6
Reverse KL	63.4	69.1	90.5	**69.1**	72.6	54.8	69.9

Table 2: Ablation on the loss type.

Implementation details. We train with LoRA ($r = 64$, $\alpha = 128$) and AdamW with learning rate 5×10^{-6}, batch size 32, and FlashAttention 3. Following previous implementations on self-distillation(Zhao et al., 2026), we train RCSD, OPSD for 100 steps and GRPO for 500 steps. More specifically, The rubric generator is trained for 1000 steps with a maximum sequence length of 2,048. The reasoner is trained for 100 steps with a maximum completion length of 4096. The teacher is fixed during training, and supervision is applied through token-level distillation on student-generated trajectories. At evaluation time, we use temperature 1.0, top-$p = 0.95$, top-$k = -1$, min-$p = 0.0$, presence penalty 0.0, and results are averaged over 4 independent generations. For LLM-as-a-Judge evaluation, we report results from a single generation. Details are in Appendix C.

3.1 Main Results

Main results are reported in Table 1. RCSD achieves the best overall average, improving over the base Qwen3-8B model by 4.7 points and outperforming the strongest baseline, OPSD, by 0.9 point. The gains are especially pronounced on rubric-based reasoning benchmarks, where RCSD improves over the base model by 8.2 points on RESEARCHQA and 4.9 points on RUBRICHUB. RCSD also obtains the best performance on SCIBENCH, reaching 70.8, suggesting that rubric-guided self-distillation remains effective for scientific reasoning tasks that require preserving multiple criterion-level reasoning signals. Compared with GRPO and GRPO-Rubrics, RCSD benefits from dense token-level supervision rather than relying on scalar rewards. Compared with OPSD, RCSD avoids distilling from a single reference-style answer and instead uses rubric-conditioned guidance, which appears to provide more flexible and task-relevant supervision. We also observe improvements on GPQA-D, PIQA, RESEARCHQA, and RUBRICHUB, showing that the method improves not only scientific reasoning but also broader rubric-guided reasoning performance.

Ablation on Loss Type. Table 2 studies the effect of different distillation losses. Forward KL performs best overall, achieving the highest average of 70.6. Reverse KL is the second-best variant, with an average of 69.9, and performs best on RAR. JSD achieves the best result on RUBRICHUB and matches Forward KL on RESEARCHQA, but shows a noticeable drop on GPQA-D, leading to a lower overall average of 69.6. These results suggest that Forward KL is the most effective objective for RCSD in this setting. One possible explanation is that Forward KL encourages the student to cover the teacher's rubric-conditioned distribution more faithfully, preserving diverse criterion-aware reasoning signals. In contrast, Reverse KL may be more mode-seeking, while JSD provides a more conservative update that is stable but less effective in transferring the full teacher signal.

Figure 3 further explains the loss-type ablation in Table 2. Forward KL achieves the best benchmark performance and shows the most favorable training behavior: student entropy

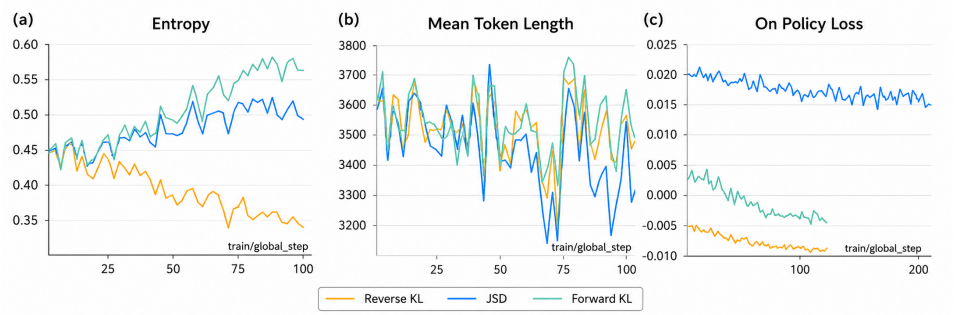

Figure 3: Training dynamics under different distillation losses: reverse KL (yellow), JSD (blue), and forward KL (green). We show student entropy, mean token length, and on-policy loss over training.

Method	GPQA-D	SciBench	PIQA	Criteria			Token Len.
				Avg	Min	Max	Avg
GT Rubrics	**65.2**	**71.0**	**91.0**	7.5	7	12	248.7
Generated Rubrics	64.5	70.6	90.8	8.4	6	20	236.6

Table 3: Ablation on rubric source. GT (Ground Truth) Rubrics use reference rubrics during reasoner training, while Generated Rubrics correspond to the learned rubric generator.

increases steadily, suggesting that the model preserves a broader rubric-conditioned output distribution rather than collapsing to narrow modes. In contrast, reverse KL consistently reduces entropy, reflecting its more mode-seeking tendency, while JSD produces a milder entropy increase. Mean token length is noisy across all objectives, with no clear systematic advantage, although JSD tends to generate slightly shorter responses later in training. The on-policy loss curves are also stable overall: forward KL decreases smoothly, reverse KL becomes increasingly negative, and JSD remains positive with a gradual decline. Together, these trends suggest that forward KL most effectively transfers the teacher's criterion-aware signal, whereas reverse KL is more restrictive and JSD is stable but less effective.

Scaling Model Size. To study whether RCSD remains effective across scales, we evaluate Qwen3-1.7B, Qwen3-4B, and Qwen3-8B on RAR, RESEARCHQA, and RUBRICHUB. Figure 4 shows that RCSD consistently improves over the corresponding base model across all three benchmarks and model sizes. At 1.7B, RCSD improves performance by 2.7 points on RAR, 4.4 points on RESEARCHQA, and 6.6 points on RUBRICHUB. At 4B, the gains are 4.9, 5.6, and 3.2 points, respectively, and at 8B, the gains are 8.9, 8.2, and 4.8 points. These results suggest that RCSD provides robust gains across model scales, while larger models achieve stronger absolute performance and continue to benefit from criterion-aware teacher signals on rubric-guided reasoning tasks.

Ablation on Rubric Source. Another central question is whether learned rubrics can approach the effectiveness of reference rubrics. Table 3 compares **GT Rubrics**, where the teacher conditions on reference rubrics, with **Generated Rubrics**, where the teacher conditions on rubrics produced by the learned rubric generator. Overall, generated rubrics remain competitive across the benchmark suite, with only small gaps to GT rubrics on GPQA-D (64.5 vs. 65.2), SCIBENCH (70.6 vs. 71.0), and PIQA (90.8 vs. 91.0). This indicates that the learned rubric generator preserves most of the benefit of reference rubrics. Generated rubrics contain slightly more criteria on average than reference rubrics (8.4 vs. 7.5), with a broader range of criterion counts (6–20 vs. 7–12), while having a similar average token length (236.6 vs. 248.7). This suggests that learned rubrics remain comparably informative for training despite not relying on manually written reference rubrics. This is important in practice: it means we do not rely on handcrafted or manually curated rubric annotations and that the two-stage pipeline is a viable instantiation of RCSD. Qualitative failure analysis is provided in Appendix E.2. Our results further show that the model trained with GT rubrics outperforms all the baselines in Table 1, demonstrating the effectiveness of token-level distillation on rubric feedback.

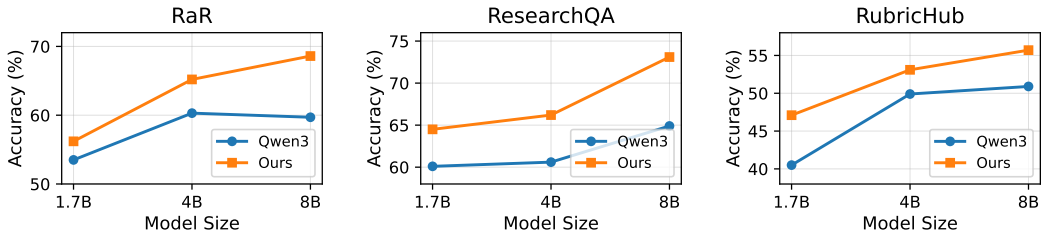

Figure 4: Model size ablation on rubric-based reasoning benchmarks at 1.7B, 4B, and 8B scales. RCSD consistently improves performance over the corresponding base model across RaR, ResearchQA, and RubricHub.

Method	GPQA-D	SciBench	PIQA	RaR	ResearchQA	RubricHub	Avg
Qwen3-8B	60.6	69.1	90.2	59.7	64.9	50.8	65.9
+8b Direct	62.0	70.2	90.4	68.5	72.9	55.4	69.9
+14b Direct	**64.9**	70.6	90.1	**69.2**	72.8	**56.5**	**70.7**
+RCSD (Ours)	64.5	**70.8**	**90.8**	68.6	**73.1**	55.7	70.6

Table 4: Ablation on the necessity of the Stage-I rubric generator. +14b Direct and +8b Direct prompt Qwen3-14B and Qwen3-8B teachers to provide rubrics directly, while +RCSD uses rubrics produced by the learned Stage-I rubric generator.

Necessity of Stage-I Rubric Generator. We further study whether the training benefits from the learned rubric generator over directly prompting the base model to generate rubrics for teacher model. As shown in Table 4, RCSD achieves a competitive average of 70.6, nearly matching +14b Direct at 70.7 and outperforming +8b Direct at 69.9. Notably, RCSD obtains the best results on SciBench, PIQA, and ResearchQA, suggesting that the learned generator can amortize rubric construction while preserving strong downstream supervision. Although +14b Direct slightly leads on the overall average, it requires prompting a larger teacher to produce rubrics directly; in contrast, the Stage-I generator provides a scalable way to generate task-specific rubrics for RCSD.

Analysis of Rubric Quality Degradation. We evaluate the sensitivity of RCSD to rubric quality by providing the teacher with generic, random, noisy, or reduced rubrics. Generic uses a shared rubric template for all examples; Random samples a rubric from another question; Noisy combines half of the original rubric with half of a random rubric; and Reduced removes half of the rubric items. As shown in Table 5, all variants improve over the base model, suggesting criterion-style supervision remains useful even with imperfect rubrics and that the teacher can partially correct unreasonable rubric items. Nevertheless, +Learned achieves the best overall results on GPQA-D, PIQA, and ResearchQA, indicating that instance-specific learned rubrics provide the most reliable supervision. The competitiveness of degraded variants suggests that the teacher can compensate for imperfect rubric information, likely by using the reference answer to reconcile rubric errors. However, the advantage of +Learned shows that rubric relevance and coherence still matter. The strong performance of +Reduced further suggests that RCSD does not depend on verbose rubrics; concise but relevant criteria preserve most of the benefit.

Method	GPQA-D	SciBench	PIQA	RaR	ResearchQA	RubricHub	Avg
Qwen3-8B	60.6	69.1	90.2	59.7	64.9	50.8	65.9
+Generic	62.1	70.2	90.2	68.7	72.6	55.6	69.9
+Noisy	61.6	**70.9**	90.4	**68.8**	72.4	55.3	69.9
+Random	63.3	70.3	90.5	68.2	72.6	54.8	70.0
+Reduced	63.0	70.6	90.6	**68.8**	72.3	**55.7**	70.2
+Learned	**64.5**	70.8	**90.8**	68.6	**73.1**	**55.7**	**70.6**

Table 5: Ablation on rubric quality. +Generic, +Noisy, +Random, and +Reduced use degraded rubric variants to test the sensitivity of distillation to rubric quality, while +Learned uses the full learned rubric supervision. Bold indicates the best result in each column.

4 Related Work

LLM Post-training Reinforcement learning (RL) has become a critical post-training tool for improving multi-step reasoning in LLMs (Zhang et al., 2025b), particularly in the form of *reinforcement learning with verifiable rewards* (RLVR) (Shao et al., 2024; Guo et al., 2025; Chollet et al., 2025; Jain et al., 2024). Despite these advances, most existing RL methods for reasoning remain fundamentally *outcome-based*. This creates a key limitation for reasoning tasks: the single-score reward is assigned to the entire sequence, and information about *where* and *how* the model failed is lost. Two responses may receive the same reward despite making very different mistake. Recent work moves beyond scalar-only rewards by introducing finer-grained supervision, either through critique-augmented learning that provides natural-language feedback on sampled reasoning traces or through process reward models that assign credit to intermediate steps in a reasoning chain (Zhang et al., 2025c; Bi et al., 2025; Lightman et al., 2023; Setlur et al., 2024; Yao et al., 2026). Yet these methods are mainly developed for domains such as math, where intermediate steps are easier to evaluate. A complementary line on on-policy distillation replaces scalar outcome rewards with dense teacher guidance. Classical knowledge distillation and sequence-level distillation train a student to imitate a teacher's outputs, but typically operate off-policy on teacher-generated trajectories (Hinton et al., 2015; Kim & Rush, 2016). Recent work on *on-policy distillation* and *on-policy self-distillation* addresses this mismatch by training the student on trajectories sampled from its own policy while using a teacher to provide token-level supervision (Agarwal et al., 2024; Xu et al., 2024; Zhao et al., 2026; Hübotter et al., 2026; Ye et al., 2026). However, existing on-policy self-distillation methods typically construct the teacher from privileged reference solutions or textual feedback. Our work builds on this line but reshapes the supervision interface: instead of conditioning the teacher on a single reference trajectory, we condition it on structured rubric feedback.

Reinforcement Learning with Rubrics The *LLM-as-a-Judge* paradigm enables scalable evaluation when human labeling is expensive or ambiguous, but coarse holistic scores are often noisy and sensitive to prompting or formatting (Zheng et al., 2023; Wang et al., 2023b). Rubric-based evaluation addresses this limitation by decomposing quality into explicit and interpretable criteria, improving consistency and enabling more fine-grained diagnosis (Gunjal et al., 2025; Arora et al., 2025; Starace et al., 2025). Building on this idea, recent work has incorporated rubrics into reinforcement learning as structured reward decompositions, extending RL-style post-training beyond strictly verifiable domains while providing more interpretable supervision than scalar rewards alone (Huang et al., 2025; Gunjal et al., 2025; Zhang et al., 2025a; Bi et al., 2025; Shao et al., 2025; Fang et al., 2026). In these approaches, rubric judgments are aggregated into scalar rewards and applied to completed responses. As a result, rubric structure helps determine *what score* a response receives, but not *how* token-level learning is carried out on the model's own trajectory. Many approaches also rely on predefined rubrics, often produced by frontier LLMs, which are costly to obtain (Shao et al., 2025). Our work differs from prior rubric-based RL in the role assigned to rubrics: we use them as privileged teacher-side supervision for on-policy self-distillation, allowing criterion-level structure to directly shape token-level updates during optimization.

5 Conclusion

We introduced *Rubric-Conditioned Self-Distillation*, a post-training framework that uses rubrics as privileged teacher supervision for on-policy self-distillation. Instead of collapsing rubric feedback into scalar rewards, RCSD preserves criterion-level structure during optimization by converting rubrics into dense token-level guidance on student-generated rollouts. Empirically, RCSD shows that preserving criterion-level feedback during on-policy distillation leads to stronger and more fine-grained reasoning. More broadly, RCSD offers a verifier-free approach to post-training open-ended reasoning models, where high-quality responses cannot always be judged by exact-match answers, executable tests, or other automatic outcome verifiers.

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 *The twelfth international conference on learning representations*, 2024.

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.

Baolong Bi, Shenghua Liu, Yiwei Wang, Siqian Tong, Lingrui Mei, Yuyao Ge, Yilong Xu, Jiafeng Guo, and Xueqi Cheng. Reward and guidance through rubrics: Promoting exploration to improve multi-domain reasoning. *arXiv preprint arXiv:2511.12344*, 2025.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Francois Chollet, Mike Knoop, Gregory Kamradt, Bryan Landers, and Henry Pinkard. Arc-agi-2: A new challenge for frontier ai reasoning systems. *arXiv preprint arXiv:2505.11831*, 2025.

Junfeng Fang, Zhepei Hong, Mao Zheng, Mingyang Song, Gengsheng Li, Houcheng Jiang, Dan Zhang, Haiyun Guo, Xiang Wang, and Tat-Seng Chua. Rubric-based on-policy distillation. *arXiv preprint arXiv:2605.07396*, 2026.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. *arXiv preprint arXiv:2306.08543*, 2023.

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.

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.

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

Zenan Huang, Yihong Zhuang, Guoshan Lu, Zeyu Qin, Haokai Xu, Tianyu Zhao, Ru Peng, Jiaqi Hu, Zhanming Shen, Xiaomeng Hu, et al. Reinforcement learning with rubric anchors. *arXiv preprint arXiv:2508.12790*, 2025.

Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, et al. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*, 2026.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. Pubmedqa: A dataset for biomedical research question answering. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 2567–2577, 2019.

Yoon Kim and Alexander M Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 1317–1327, 2016.

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.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The twelfth international conference on learning representations*, 2023.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Conference on health, inference, and learning*, pp. 248–260. PMLR, 2022.

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. In *First conference on language modeling*, 2024.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for llm reasoning. *arXiv preprint arXiv:2410.08146*, 2024.

Rulin Shao, Akari Asai, Shannon Zejiang Shen, Hamish Ivison, Varsha Kishore, Jingming Zhuo, Xinran Zhao, Molly Park, Samuel G Finlayson, David Sontag, et al. Dr tulu: Reinforcement learning with evolving rubrics for deep research. *arXiv preprint arXiv:2511.19399*, 2025.

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.

Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, et al. Paperbench: Evaluating ai's ability to replicate ai research. *arXiv preprint arXiv:2504.01848*, 2025.

Xiaoxuan Wang, Ziniu Hu, Pan Lu, Yanqiao Zhu, Jieyu Zhang, Satyen Subramaniam, Arjun R Loomba, Shichang Zhang, Yizhou Sun, and Wei Wang. Scibench: Evaluating college-level scientific problem-solving abilities of large language models. *arXiv preprint arXiv:2307.10635*, 2023a.

Yidong Wang, Zhuohao Yu, Zhengran Zeng, Linyi Yang, Cunxiang Wang, Hao Chen, Chaoya Jiang, Rui Xie, Jindong Wang, Xing Xie, et al. Pandalm: An automatic evaluation benchmark for llm instruction tuning optimization. *arXiv preprint arXiv:2306.05087*, 2023b.

Wenda Xu, Rujun Han, Zifeng Wang, Long T Le, Dhruv Madeka, Lei Li, William Yang Wang, Rishabh Agarwal, Chen-Yu Lee, and Tomas Pfister. Speculative knowledge distillation: Bridging the teacher-student gap through interleaved sampling. *arXiv preprint arXiv:2410.11325*, 2024.

Jiarui Yao, Ruida Wang, and Tong Zhang. Prl: Process reward learning improves llms' reasoning ability and broadens the reasoning boundary. *arXiv preprint arXiv:2601.10201*, 2026.

Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. On-policy context distillation for language models. *arXiv preprint arXiv:2602.12275*, 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*, 2026. To appear.

Weizhe Yuan, Jane Yu, Song Jiang, Karthik Padthe, Yang Li, Ilia Kulikov, Kyunghyun Cho, Dong Wang, Yuandong Tian, Jason E Weston, et al. Naturalreasoning: Reasoning in the wild with 2.8 m challenging questions. *arXiv preprint arXiv:2502.13124*, 2025.

Junkai Zhang, Zihao Wang, Lin Gui, Swarnashree Mysore Sathyendra, Jaehwan Jeong, Victor Veitch, Wei Wang, Yunzhong He, Bing Liu, and Lifeng Jin. Chasing the tail: Effective rubric-based reward modeling for large language model post-training. *arXiv preprint arXiv:2509.21500*, 2025a.

Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, Che Jiang, Yuchen Fan, Kai Tian, Guoli Jia, Pengfei Li, et al. A survey of reinforcement learning for large reasoning models. *arXiv preprint arXiv:2509.08827*, 2025b.

Xiaoying Zhang, Yipeng Zhang, Hao Sun, Kaituo Feng, Chaochao Lu, Chao Yang, and Helen Meng. Critique-grpo: Advancing llm reasoning with natural language and numerical feedback, 2025c. URL https://arxiv.org/abs/2506.03106.

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.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

A Training Procedure

Algorithm 1 summarizes the full training procedure. Stage I amortizes expensive instance-specific criteria into a rubric generator, and Stage II uses those criteria to shape teacher-side token-level correction on student rollouts. The resulting framework preserves criterion-level structure, remains on-policy, and better utilize this rich structured information in textual rubrics rather than scalar rewards.

Algorithm 1 RCSD: Rubric-Conditioned On-Policy Self-Distillation

Require: Training set $\mathcal{D} = \{(x, r^\star, y^\star)\}$; rubric generator p_S^R; reasoner p_S^Y
Stage I: Learn rubric generator
for each training example x with reference answer $y^\star$ **do**
  Sample rubric trajectory $\hat{r} \sim p_S^R(\cdot \mid x)$
  Compute teacher and student token distributions along $\hat{r}$
  Update rubric generator by minimizing $\mathcal{L}_{\text{rubric}}$
end for
Stage II: Train rubric-conditioned reasoner
for each training example x **do**
  Obtain rubric $\hat{r}$ from $\hat{r} \sim p_S^R(\cdot \mid x)$
  Sample answer rollout $\hat{y} \sim p_S^Y(\cdot \mid x)$
  Compute teacher and student token distributions along $\hat{y}$
  Update reasoner by minimizing $\mathcal{L}_{\text{reason}}$
end for

B Experiments

Generalization to other Domains. We also evaluate generalization ability on medicine benchmarks in Table 6. Although our method is trained primarily on scientific reasoning tasks rather than medical-domain data, it maintains competitive performance on both MEDMCQA and PUBMEDQA. In particular, RCSD improves over the base Qwen3-8B model on both benchmarks, from 64.5 to 65.8 on MEDMCQA and from 74.2 to 75.1 on PUBMEDQA. These results suggest that rubric-guided teacher supervision does not lead to catastrophic forgetting on adjacent knowledge-intensive domains, while preserving strong general reasoning ability beyond the training distribution.

Training Steps Analysis. Figure 5 shows the performance trajectory across different training checkpoints. On GPQA-D, performance improves steadily from the base model and peaks at 120 steps, suggesting that RCSD is most effective with moderate training. On SCIBENCH, performance improves early and remains relatively stable across later checkpoints, indicating that the method quickly adapts to scientific reasoning tasks without substantial degradation from continued training.

Method	MedMCQA	PubMedQA
Qwen3-8B	64.5	74.2
+GRPO	65.1	**76.2**
+GRPO-Rubrics	65.6	74.2
+OPSD	**66.0**	74.4
+RCSD (ours)	65.8	75.1

Table 6: Generalization to medicine benchmarks. Best performance is bolded.

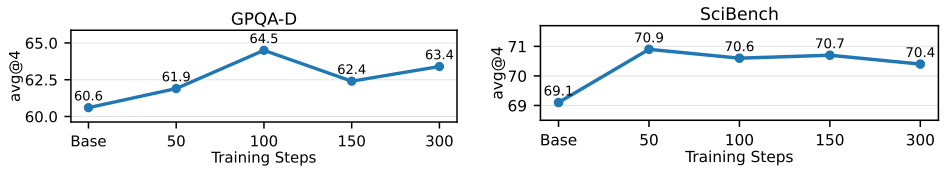

Figure 5: Performance across training checkpoints on GPQA-D and SciBench.

C Implementation Details

In **standard GRPO** with an LLM judge, each rollout is scored by a fixed judge model (Qwen3-14B), which is conditioned on the user prompt, the model completion, and a reference answer. The judge outputs a discrete quality score on a 1–10 scale, which we normalize to ($[0, 1]$) to serve as the reward signal. In **GRPO-Rubrics** method, we first perform an offline rubric-generation step, where Qwen3-14B, prompted as a rubric writer, maps each training example's question and reference answer into a structured rubric. During training, the reward judge is conditioned on the prompt, the model completion, and the pre-generated rubric, and produces a holistic score. As a result, rewards reflect rubric satisfaction rather than direct comparison to the gold text. Training configurations are reported in Table 7.

Table 7: Training Configurations

Parameter	**GRPO**	**RCSD/OPSD**	**SFT**
Learning Rate	5×10^{-6}	5×10^{-6}	5×10^{-6}
Max Completion Length	4096	4096	4096
Batch Size	32	32	32
Sampling Temperature	1.2	1.2	1.2
Training Steps	500	100	500
Number of Generations per Prompt	4	1	1

D Prompt Templates

We provide the prompt templates used in our rubric generation and LLM-as-a-Judge evaluation, where we adopt the prompt format from RaR (Gunjal et al., 2025). For rubric generation, we instruct the model to generate self-contained, criterion-level rubrics in a structured JSON format. For LLM-as-a-Judge evaluation, we ask the evaluator to holistically score a generated response according to the provided rubrics and return a JSON-formatted rating.

D.1 Rubric Generator Prompt

Rubric Generator Prompt

```
You are an expert rubric writer. Your job is to generate a self-contained set of
    evaluation criteria ("rubrics") for judging how good a response is to a
    given question. Rubrics can cover aspects of a response such as, but not
    limited to, factual correctness, ideal-response characteristics, style,
    completeness, helpfulness, harmlessness, patient-centeredness, depth of
    reasoning, contextual relevance, and empathy.

Each item must be self-contained: non-expert readers should not need to infer
    anything or consult external information. Begin each description with its
    category: "Essential Criteria: ...", "Important Criteria: ...", "Optional
    Criteria: ...", or "Pitfall Criteria: Does not mention ...".

Inputs:

question: The full question text.
reference_answer: The ideal answer, including any specific facts, explanations,
    or advice.

Total items:

Choose 5--15 rubric items based on the complexity of the question.
```

```
Each rubric item:

title: 2--4 words.
description: One sentence starting with its category prefix that explicitly
     states exactly what to look for.
weight: For Essential/Important/Optional, use 1--5; for Pitfall, use -1 or -2.

Category guidance:

Essential: Critical facts or safety checks; if missing, the response is invalid.
     Use weight 5.
Important: Key reasoning, completeness, or clarity; strongly affects quality. Use
      weight 3--4.
Optional: Helpful style or extra depth; nice to have but not deal-breaking. Use
     weight 1--2.
Pitfall: Common mistakes or omissions specific to this prompt. Use weight -1 or
     -2.

To ensure self-contained guidance:

When referring to answer choices, explicitly say "Identifies (A)", "Identifies (B
     )", etc., rather than vague phrasing.
If the format requires a conclusion like "The final answer is (B)", include a
     rubric item such as: "Essential Criteria: Includes a clear statement 'The
     final answer is (B)'."
If reasoning should precede the answer, include a rubric item such as: "Important
      Criteria: Presents the explanation before stating the final answer."
If brevity is valued, include a rubric item such as: "Optional Criteria: Remains
     concise and avoids unnecessary detail."
If the question context demands mention of specific findings, include that
     explicitly.

Output:
Provide a JSON array of rubric objects. Each object must contain exactly three
     keys: title, description, and weight. Do not copy large blocks of the
     question or reference_answer into the text. Each description must begin with
      its category prefix, and no extra keys are allowed.

Now, given the question and reference_answer, generate the rubric as described.
     The reference answer is an ideal response but not necessarily exhaustive;
     use it only as guidance.
```

D.2 LLM-as-a-Judge Prompt

LLM-as-a-Judge Prompt

```
System Prompt:
You are an expert evaluator. Given a user prompt, a generated response, and a
     list of quality rubrics, please rate the overall quality of the response on
     a scale of 1 to 10 based on how well it satisfies the rubrics.

Consider all rubrics holistically when determining your score. A response that
     violates multiple rubrics should receive a lower score, while a response
     that satisfies all rubrics should receive a higher score.

Start your response with a valid JSON object that starts with json and ends with .
      The JSON object should contain a single key "rating" and the value should
     be an integer between 1 and 10.
```

```
Example response:

{
  "rating": 7
}

User Prompt Template:
Given the following prompt, response, and rubrics, please rate the overall
     quality of the response on a scale of 1 to 10 based on how well it satisfies
      the rubrics.

Your JSON Evaluation:
```

E Qualitative Analysis

E.1 Case Study on Reasoning Quality

We compare model responses on a simple ideal-gas-law problem. Although all responses identify the correct governing equation $P = \frac{nRT}{V}$, they differ substantially in verbosity, consistency, and numerical discipline.

Prompt. *Suppose that* $10.0\,\mathrm{mol}\ \mathrm{C_2H_6(g)}$ *is confined to* $4.860\,\mathrm{dm}^3$ *at* $27°\mathrm{C}$. *Predict the pressure exerted by the ethane from the perfect gas. The unit of the answer is* atm. *Let's think step by step.*

RCSD

Token count: 2896
Selected quotes:

> "First, I need to convert the temperature from Celsius to Kelvin. The given temperature is 27°C. To convert, I add 273.15. So $27 + 273.15 = 300.15\,\mathrm{K}$."
> "The volume is given in dm^3, which is the same as liters ... $V = 4.860\,\mathrm{L}$."
> "Since the answer should be in atm, I need to use the value of R that has units $\mathrm{L \cdot atm/(mol \cdot K)}$ ... $R = 0.08206$."
> "Now plug these into $PV = nRT$. Solving for P: $P = \frac{nRT}{V}$."
> "So total $P \approx 50.679647$. Rounding to three decimal places gives $50.680\,\mathrm{atm}$."
> "The answer is therefore $\boxed{50.680}$."

Response excerpt. The response first converts temperature to Kelvin, maps $4.860\,\mathrm{dm}^3$ to $4.860\,\mathrm{L}$, selects $R = 0.08206\,\mathrm{L \cdot atm/(mol \cdot K)}$, and applies

$$P = \frac{nRT}{V}.$$

It then computes

$$P = \frac{10.0 \times 0.08206 \times 300.15}{4.860} = \frac{246.30309}{4.860} \approx 50.680.$$

The response ends with a short, clean answer:

$$\boxed{50.680}.$$

OPSD

Token count: 4936
Selected quotes:

"The ideal gas law is $PV = nRT$, right?"
"Since the answer needs to be in atmospheres, I should use the R value that has units of $\mathrm{L \cdot atm/(mol \cdot K)}$ ... $R = 0.082057$."
"The formula is $P = nRT/V$."
"So the numerator is approximately $246.2940855\,\mathrm{L \cdot atm}$."
"So total is approximately $50.677 + 0.000795 \approx 50.6778$. So approximately $50.6778\,\mathrm{atm}$. Rounding to three decimal places, that would be $50.678\,\mathrm{atm}$."
"Therefore, the answer is $\boxed{50.678}$."

Response excerpt. The response follows the same overall structure, but uses a more precise constant $R = 0.082057\,\mathrm{L \cdot atm/(mol \cdot K)}$, leading to

$$P = \frac{10.0 \times 0.082057 \times 300.15}{4.860} = \frac{246.2940855}{4.860} \approx 50.677795.$$

After repeated recalculation and self-verification, it rounds to

$$\boxed{50.678}.$$

The response is correct, but noticeably more repetitive than RCSD.

Qwen3-8B

Token count: 9294
Selected quotes:

"The gas constant R is $0.0821\,\mathrm{L \cdot atm/(mol \cdot K)}$, right?"
"Wait, but sometimes R is taken as 0.08206 ..."
"The exact value of R is 0.082057 ..."
"So with $R = 0.0821$, the answer is approximately $50.704\,\mathrm{atm}$."
"But if I use $R = 0.08206$, it's approximately $50.680\,\mathrm{atm}$."
"But since the problem might expect using $R = 0.0821$, I think the answer is expected to be around $50.704\,\mathrm{atm}$."

Response excerpt. The response begins correctly by identifying $PV = nRT$, converting 27°C to 300.15 K, and noting that $4.860\,\mathrm{dm}^3 = 4.860\,\mathrm{L}$. However, it repeatedly switches between different values of the gas constant:

$$R = 0.0821, \quad 0.08206, \quad 0.082057.$$

This causes the intermediate calculations to drift between

$$50.704, \quad 50.680, \quad 50.678,$$

and the response becomes very long, circular, and self-contradictory. In particular, the model explicitly states that it will "proceed with $R = 0.0821$," which yields approximately

$$\boxed{50.704}.$$

Even though the underlying formula is correct, the response demonstrates weaker numerical stability and substantially poorer reasoning efficiency.

This example reveals a clear difference in *reasoning quality*, not just final-answer correctness. All three models know the ideal gas law and the required unit conversions, but they differ in how efficiently and consistently they execute the solution. RCSD produces the strongest trajectory. Its response is the shortest, commits early to a coherent numerical setup, and reaches a correct answer without revisiting earlier choices. OPSD also arrives at a correct answer, but its reasoning is more verbose and repetitive: it repeatedly re-checks arithmetic that has already been established, which increases token usage without improving solution quality. In contrast, the base QWEN3-8B response is substantially longer and exhibits a more serious failure mode: it repeatedly changes core numerical assumptions, especially the value of R, and consequently oscillates between different final answers. Overall, this case study suggests that RCSD improves *reasoning efficiency* and *trajectory stability*: its responses

are shorter, less repetitive, and more internally consistent than those produced by OPSD and the base model.

E.2 Failure Analysis on Learned Rubric Quality

Learned Rubrics

Phonon / Bose–Einstein Example

1. **Bose-Einstein Derivation** (5): Essential Criteria: The response must clearly derive the Bose-Einstein distribution formula for phonon occupation number $n(\omega)$ using the given dispersion relation $\omega(k) = ck$.
2. **Temperature Dependence** (4): Important Criteria: The response should correctly express the temperature dependence of $n(\omega)$ as $n(\omega) = 1/(e^{\hbar\omega/kT} - 1)$, demonstrating the exponential dependence on temperature and frequency.
3. **Frequency-Dependent Behavior** (5): Essential Criteria: The response must explicitly explain how the occupation number $n(\omega)$ changes with temperature for different ω values, emphasizing the inverse relationship between $n(\omega)$ and temperature.
4. **Physical Interpretation** (4): Important Criteria: The response should include a clear explanation of the physical meaning of the Bose-Einstein distribution in the context of phonon statistics and thermal equilibrium.
5. **High/Low Temperature Behavior** (3): Optional Criteria: The response may include a brief discussion on the behavior of $n(\omega)$ at high and low temperatures, such as the classical limit or zero-point energy effects.
6. **Frequency Dependence Omission** (-1): Negative Criteria: The response must not omit the dependence of the occupation number on the frequency ω, as this is a critical aspect of the temperature dependence.
7. **Temperature Independence Error** (-1): Negative Criteria: The response must not incorrectly state that the occupation number is independent of temperature, which would contradict the Bose-Einstein distribution.
8. **Dispersion Relation Usage** (4): Important Criteria: The response should correctly use the dispersion relation $\omega(k) = ck$ to connect the phonon frequency with wavevector k in the derivation.
9. **Bose-Einstein Application** (4): Important Criteria: The response must demonstrate the correct application of Bose-Einstein statistics to phonons, including the proper handling of the occupation number formula.
10. **Frequency-Temperature Interaction** (3): Optional Criteria: The response may include a clear explanation of how the occupation number changes with frequency for different temperature values, enhancing the understanding of the system.

Soda-Lime Titration Example

1. **Mass Conversion** (5): Essential Criteria: The response must accurately convert the mass of soda lime to the masses of NaOH and CaO components using the given percentages.
2. **Mole Calculation** (5): Essential Criteria: The response must correctly compute the moles of NaOH and CaO from their respective masses using their molar masses.
3. **Reaction Equations** (4): Important Criteria: The response should accurately identify and write the balanced chemical equations for the neutralization reactions of NaOH and CaO with HCl.
4. **Stoichiometric Ratio** (4): Important Criteria: The response must correctly apply stoichiometric ratios from the balanced equations to relate moles of NaOH and CaO to moles of HCl required for neutralization.
5. **Total Moles Calculation** (5): Essential Criteria: The response should correctly calculate the total moles of HCl required by summing the moles from both NaOH and CaO neutralization steps.
6. **Volume Calculation** (5): Essential Criteria: The response must accurately determine the volume of 0.500M HCl needed using the total moles and the given molarity, converting to the correct units.

7. **Step-by-Step Explanation** (4): Important Criteria: The response should present a clear, step-by-step explanation of the calculation process for both NaOH and CaO neutralization, ensuring logical flow.
8. **Application Context** (3): Optional Criteria: The response may include a brief discussion of the significance of the neutralization reactions in real-world applications, though it is not required.
9. **Component Omission** (-1): Negative Criteria: The response must not omit the neutralization of either NaOH or CaO components, as both are required for accurate calculation.
10. **Molar Mass Accuracy** (-1): Negative Criteria: The response must not use incorrect molar masses for NaOH or CaO, as this would lead to wrong results.

Reference Rubrics (RaR-Science)

Phonon / Bose–Einstein Example

1. **Bose-Einstein Distribution** (5): Essential Criteria: The response must explicitly state and correctly use the Bose-Einstein distribution formula for $n(\omega)$, such as $n(\omega) = 1/(\exp(\hbar\omega/(k_B T)) - 1)$, linking ω and T in the derivation.
2. **Dispersion Relation Use** (4): Important Criteria: The answer should correctly incorporate the given dispersion relation $\omega(k) = ck$ to connect the frequency ω to the wave vector k in the context of the phonon system.
3. **Temperature Analysis** (5): Essential Criteria: The response must analyze how the phonon occupation number $n(\omega)$ varies with temperature for a fixed frequency and explain differences in behavior at various ω values.
4. **Mathematical Derivation** (4): Important Criteria: The answer should include clear and logically structured derivations that break down the mathematical steps required to arrive at the temperature dependence of $n(\omega)$.
5. **Frequency Trends** (3): Optional Criteria: The response may provide a concrete example or detailed explanation illustrating that at lower frequencies the occupation number is more sensitive to changes in temperature than at higher frequencies.
6. **Clarity and Conciseness** (3): Optional Criteria: The answer should be clear and concise, avoiding unnecessary elaboration while still covering all key elements of the derivation and conclusion.
7. **Exclusion of Loss Effects** (-1): Pitfall Criteria: The response should not include irrelevant factors such as frictional or damping losses, which are not part of the ideal derivation using Bose-Einstein statistics.

Soda-Lime Titration Example

1. **Separate Reactions** (5): Essential Criteria: The response must separately address the neutralization reactions for both NaOH and CaO components, calculating the moles of acid required for each reaction.
2. **Stoichiometry Accuracy** (5): Essential Criteria: The answer should correctly apply stoichiometric relationships to determine the moles of HCl needed for the complete neutralization of both compounds.
3. **Molarity Application** (4): Important Criteria: The response must demonstrate how the molarity of 0.500M HCl is used to convert the required moles of acid into the corresponding volume in cm^3 with proper unit conversions.
4. **Step-by-Step Work** (4): Important Criteria: The answer should provide a clear, logical sequence of calculations that lead to the final volume, ensuring transparency in each intermediary step.
5. **Final Volume Accuracy** (5): Essential Criteria: The response must explicitly state the correct final volume of 0.500M HCl required (133.04 cm^3) for complete neutralization.
6. **Unit Consistency** (2): Optional Criteria: The explanation should include correct unit conversions, especially showing how volumes are converted (e.g., L to cm^3), to enhance clarity and precision.

7. **Reaction Assumptions** (-2): Pitfall Criteria: The response should mention that the reactions are assumed to go to completion without interference, and neglecting to state such assumptions is a common oversight.

Failure analysis. These examples illustrate two recurring weaknesses of the learned rubrics relative to the reference rubrics. First, the learned rubrics are often *bloated and redundant*. In the phonon/Bose–Einstein example, the learned rubric expands to 10 criteria versus 7 in the reference rubric, and several titles repeat the same underlying evaluation axis: *Temperature Dependence*, *Frequency-Dependent Behavior*, *High/Low Temperature Behavior*, and *Frequency-Temperature Interaction* all partially restate the same requirement. By contrast, the reference rubric compresses this content into a smaller and sharper set of criteria, such as *Temperature Analysis*, *Mathematical Derivation*, and *Frequency Trends*. This suggests that the learned rubric generator tends to over-segment closely related concepts instead of consolidating them into a compact set of discriminative checks.

Second, the learned rubrics sometimes include *generic but weakly task-critical criteria*. In the soda-lime titration example, the learned rubric includes items such as *Step-by-Step Explanation* and *Application Context*, which are broadly reasonable but not central to the actual grading target. The reference rubric instead concentrates on the chemically essential checks: *Separate Reactions*, *Stoichiometry Accuracy*, *Final Volume Accuracy*, and *Unit Consistency*. Overall, these cases suggest that the learned rubrics usually identify the correct topic, but they are often more verbose, overlapping, and less precisely aligned with the true grading objective than the reference rubrics.