EVORUBRIC: SELF-EVOLVING RUBRIC-DRIVEN RL FOR OPEN-ENDED GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement Learning (RL) has advanced Large Language Models (LLMs) in verifiable domains, while open-ended generation remains challenging due to the absence of definitive rewards. Rubric-based RL provides explicit evaluation criteria, but learning to construct these criteria remains challenging when final-answer correctness is not verifiable. We propose **EvoRubric**, a co-evolutionary RL framework that combines criterion-validity feedback, response discrimination, and peer agreement to learn rubrics for open-ended generation. A shared policy acts as both a Reasoner and a Rubric Generator, using its current responses and historical rubrics to discover new evaluation dimensions. To combine adaptive rubric discovery with a stable validity check, a frozen copy of the initial policy serves as the Meta-Verifier, while a frozen Grader scores responses against the retained criteria. Discriminative feedback, Leave-One-Out peer consensus, and a persistent memory pool transform this feedback into complementary rewards that jointly optimize both policy roles, closing the loop between response improvement and rubric discovery. EvoRubric improves over matched static and external evolving-rubric baselines across five benchmarks in Medical, Writing, and Science. Across three training seeds, it achieves five-benchmark averages of 56.28 at 8B and 61.13 at 14B, exceeding the strongest matched baselines by 3.09 and 2.06 points, respectively. Human audits assess criterion validity and response quality, and experiments with expert-initialized rubrics demonstrate compatibility with human priors. Code and data are available at https://anonymous.4open.science/r/EvoRubric-8FF9.

1 INTRODUCTION

Reinforcement Learning (RL) has been widely adopted to enhance the capabilities of Large Language Models (LLMs). In particular, Reinforcement Learning with Verifiable Rewards (RLVR) has significantly elicited advanced reasoning and deliberation capabilities in objectively verifiable domains, such as math and coding, as evidenced by standout models like DeepSeek-R1 (Guo et al., 2025) and the OpenAI o-series (OpenAI, 2024). However, beyond these verifiable tasks, a vast majority of real-world queries are inherently open-ended and lack definitive, unique answers, making direct policy optimization profoundly difficult. To address this, recent studies (Gunjal et al., 2026; Huang et al., 2025; Wei et al., 2026; Li et al., 2026; He et al., 2025) have proposed Rubric-based RL methods. By delineating a series of explicit, verifiable criteria to assess various dimensions of a response, these methods offer a promising direction for optimizing models in open-ended scenarios.

Rubric-based optimization involves two complementary functions: constructing criteria that specify response quality and applying those criteria to produce reward signals. A Grader executes the resulting rubric, while the rubric determines which response properties receive supervision. As the policy improves, its responses expose new strengths and failure modes that the evaluation criteria should capture. This suggests treating the ability to construct useful rubrics as a learning objective alongside the ability to produce high-quality responses.

Fixed rubrics can miss emerging response properties as the policy evolves (Figure 1a,b). DR-Tulu (Shao et al., 2025) addresses this through external rubric updates and a dynamic criterion pool (Figure 1c). RLCER (Sheng et al., 2026) further trains a shared Reasoner and Rubricator, using correlation with final-answer correctness to supervise rubric quality. Open-ended generation often lacks this correctness reference. The resulting challenge is therefore *how to learn useful evaluation*

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

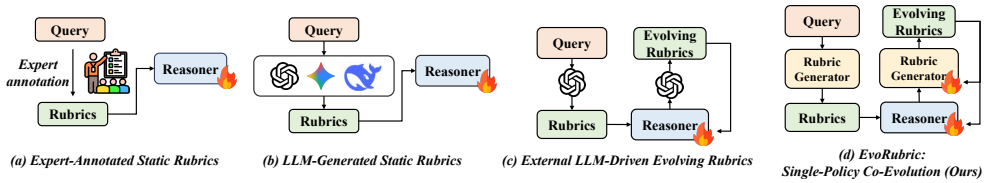


Figure 1: Rubric construction paradigms for open-ended generation: (a) expert-authored rubrics, (b) model-generated static rubrics, (c) external LLM-driven evolving rubrics, and (d) EvoRubric, which jointly optimizes response and rubric generation within a shared policy.

criteria from current responses when final-answer correctness cannot provide the rubric-learning target.

To address this challenge, we propose **EvoRubric**, which constructs rubric-learning feedback from criterion validity, response discrimination, and agreement among candidate rubric sets. A shared policy jointly learns response generation and rubric construction under these signals. The Reasoner samples candidate responses, whose differences provide concrete evidence for the Rubric Generator to discover new evaluation dimensions. Role-specific rewards train the Generator to produce valid, discriminative criteria and the Reasoner to improve its responses under these criteria. As the response distribution changes, the Generator learns to identify the evaluation dimensions that remain informative. We call this process *self-evolving*: the policy learns to construct supervision from its own responses, uses that supervision to improve, and carries selected criteria forward through rubric memory. Rubric discovery becomes an integral part of policy learning without a separately parameterized rubric generator.

The feedback signals address complementary aspects of rubric quality. A frozen copy of the initial policy serves as the Meta-Verifier, screening criterion validity using the same model family and scale. A frozen Grader applies the retained criteria to responses. Discriminative feedback rewards criteria that separate current responses, LOO consensus measures agreement across rubric batches, and persistent memory carries selected criteria across updates. Together, these mechanisms provide rubric-learning supervision without a verifiable final-answer target.

We evaluate EvoRubric on five benchmarks across Medical, Writing, and Science. Across three training seeds, EvoRubric achieves the highest mean scores among matched methods on all five benchmarks at both 8B and 14B, improving the macro-average by 3.09 and 2.06 points, respectively. A stronger GPT-4.1-generator baseline, longitudinal rubric audits, and blind human response comparisons provide complementary evidence. Experiments with expert-initialized rubrics further show compatibility with human priors.

In summary, our main contributions are as follows:

- We introduce **EvoRubric**, a rubric-learning framework for open-ended generation that constructs training feedback without a verifiable final-answer correctness reference.
- We integrate frozen initial-policy verification, discriminative feedback, and peer consensus with persistent rubric memory to support joint response and rubric learning.
- Independent-judge evaluation across three training seeds at both 8B and 14B demonstrates gains over matched baselines across five benchmarks. Human audits characterize residual errors and rubric diversity, and expert-initialization experiments show compatibility with human priors.

2 Related Work

Open-Ended QA. Driven by rapid advancements in Large Language Model (LLM) architectures and capabilities, alongside their significantly expanded context window capacities (Peng et al., 2024; Gao et al., 2025; Guan et al., 2026), LLM-generated responses are shifting from short-text tasks with deterministic answers (e.g., extractive QA (Trivedi et al., 2022) or multiple-choice questions (Rein et al., 2024)) to increasingly complex, open-ended long-text domains (e.g., medical question-answering (Arora et al., 2025) and creative writing (Wu et al., 2025b)). Consequently, assessing the

"correctness" and "verifiability" of these responses has become substantially more challenging. To address this, a growing body of research has pivoted towards the "LLM-as-a-Judge" paradigm (Song et al., 2024; Du et al., 2025), leveraging LLMs to automatically evaluate response quality. More recently, to achieve more precise verification of response content and quality, emerging benchmarks have tended to design a series of fine-grained evaluation criteria to determine whether a response satisfies specific requirements. For instance, HealthBench (Arora et al., 2025) incorporates expert-curated, medical-grade rubrics as checklists to thoroughly examine response quality. However, these annotation methodologies still heavily rely on human domain expertise or state-of-the-art closed-source models.

Reinforcement Learning with Rubrics. Reinforcement Learning with Verifiable Rewards (RLVR) has driven significant progress in deterministic domains like math and coding, empowering models such as DeepSeek-R1 (Guo et al., 2025) and the OpenAI o-series (OpenAI, 2024). However, extending this success to open-domain QA is challenging due to the difficulty of defining objective rewards for open-ended answers. Early RLHF methods measured human preferences via scalar scores (Ouyang et al., 2022; Meng et al., 2024; Dai et al., 2024), but often lack interpretability and suffer from reward hacking and sparse feedback (Fu et al., 2025; Chaudhari et al., 2026). Recently, Rubric-based RL has emerged, using high-quality rubrics as structured optimization signals (He et al., 2025; Huang et al., 2025; Xu et al., 2026; Li et al., 2026; Wei et al., 2026; Gunjal et al., 2026). Rubric construction determines the supervision available to a policy. DR-Tulu (Shao et al., 2025) updates externally generated criteria using current responses and a maintained pool. RLCER (Sheng et al., 2026) jointly optimizes a shared Reasoner and Rubricator with correctness-based rubric supervision. Rubric-ARM (Xu et al., 2026) instead alternates optimization of a rubric generator and a judge against preference labels. EvoRubric jointly trains the response policy and rubric generator while keeping the Grader fixed, constructing rubric-learning feedback from validity checks and peer agreement when a verifiable correctness reference is unavailable. Appendix A compares these supervision mechanisms.

3 EvoRubric

Problem formulation. We consider rubric-based RL for open-ended generation, where a rubric specifies evaluation criteria and a Grader applies them to candidate responses. Our objective is to learn high-quality responses and informative evaluation criteria without a verifiable final-answer correctness target. A shared policy π_θ performs response generation and rubric generation under role-specific prompts. Its initial checkpoint π_{θ_0} serves as a frozen Meta-Verifier, while a frozen Grader produces criterion-level response scores. Complementary rewards update θ for both generation roles.

Reinforcement Learning with GRPO. We employ Group Relative Policy Optimization (GRPO) (Shao et al., 2024). For each query and role, advantages are computed by standardizing rewards within the corresponding rollout group: $\hat{A}_i = (r_i - \mu)/(\sigma + \varepsilon)$, where μ and σ are the group reward mean and standard deviation, and ε ensures numerical stability. Reasoner and Generator rollouts form separate advantage groups and are then combined for a shared policy update using the clipped GRPO objective (Appendix B).

Rubric-based scoring. A rubric $\mathcal{C} = \{c_1, \ldots, c_K\}$ contains positive requirements and negative constraints with signed weights w_k. The Grader determines whether each criterion is met, yielding $s_{i,k} = w_k \mathbb{I}[c_k \text{ is met by } o_i]$, where $\mathbb{I}$ is the indicator function. Base, historical, and current rubric sets use the same signed-score normalization, yielding scores in $[-1, 1]$. These scores provide fixed base-rubric and evolving rubric-pool feedback (Appendix C).

3.1 Multi-Role Rollout under a Single Policy

EvoRubric couples response generation, criterion evaluation, and shared-policy optimization (Figure 2). Let p_{rea} and p_{gen} denote the role-specific prompts for the Reasoner and Rubric Generator, respectively. Given a query q, the Reasoner first samples N responses $\mathcal{O} = \{o_1, \ldots, o_N\}$:

$$o_i \sim \pi_\theta(\cdot \mid q, p_{rea}). \tag{1}$$

Within its input budget, the Generator then examines the sampled response text together with fixed base rubrics $\mathcal{C}_{base}$ and historical criteria $\mathcal{C}_{hist}$ retrieved from the Memory Pool, sampling M candidate

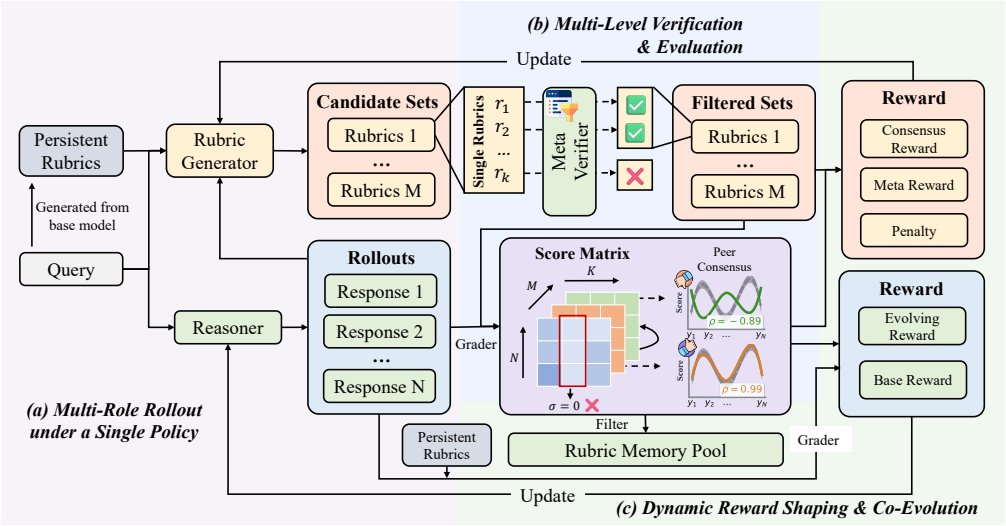


Figure 2: Overview of **EvoRubric**. A shared policy learns response generation and rubric discovery. A frozen initial-policy Meta-Verifier screens criteria, and a frozen Grader scores responses. Discriminative feedback, peer consensus, and rubric memory provide rewards for both policy roles.

rubric sets:

$$R_m \sim \pi_\theta(\cdot \mid q, \mathcal{O}, \mathcal{C}_{base}, \mathcal{C}_{hist}, p_{gen}). \tag{2}$$

Each set $R_m = \{r_{m,1}, \ldots, r_{m,K_m}\}$ targets evaluation dimensions missing from the existing criteria. Both roles use the current policy, so rubric discovery adapts to its evolving responses. We sample $N = 8$ responses and $M = 4$ rubric sets per query and maintain at most 10 evolved criteria in each query's pool. Fixed base rubrics are stored separately.

3.2 Multi-Level Verification & Evaluation

3.2.1 Motivation

Rubric-based RL depends on criteria that provide useful reward differences among the policy's current responses. Without a verifiable final-answer target, the Generator needs feedback on both criterion validity and its value for learning. We combine a frozen initial-policy validity check with two complementary signals: **discriminative power (variance)**, which identifies criteria that separate current responses and can guide their relative improvement; and **peer consensus (correlation)**, which favors rubric batches whose relative scores agree on these responses. A valid criterion may become uninformative once all responses satisfy it, while a discriminative criterion may encode an unsupported requirement. Multi-level verification therefore evaluates validity, response separation, and cross-batch agreement together to train rubric discovery as the response distribution evolves.

3.2.2 Verification signals

Meta-verification and grading. The Meta-Verifier screens each criterion for factual or logical errors, unsupported requirements, and redundancy. With validity indicator $v(r) \in \{0, 1\}$, the retained set is

$$\tilde{R}_m = \{r \in R_m \mid v(r) = 1\}.$$

The Grader applies each retained criterion to all responses, producing the score vector $\mathbf{s}_r = [s(o_1, r), \ldots, s(o_N, r)]^\top$.

Discriminative feedback. A criterion is discriminative when $\mathrm{std}(\mathbf{s}_r) > 0$, separating responses within the current group. The proportion of such criteria contributes to the Generator reward. Historical criteria with zero variance are excluded from the current pool score, and saturated criteria for which all responses exhibit the desired behavior are removed as the pool evolves.

Leave-One-Out peer consensus. Let $F(o_i; \tilde{R}_m)$ denote the normalized score from a verified candidate batch and $\mathbf{v}_m = [F(o_1; \tilde{R}_m), \ldots, F(o_N; \tilde{R}_m)]^\top$. For a batch with a nonconstant score vector and at least one nonconstant peer, we compute:

$$\mathbf{v}_{\setminus m} = \frac{1}{|\mathcal{P}_m|} \sum_{j \in \mathcal{P}_m} \mathbf{v}_j, \qquad \rho_m = \mathrm{corr}(\mathbf{v}_m, \mathbf{v}_{\setminus m}),$$

where $\mathcal{P}_m$ contains the other nonconstant batches evaluated on the same responses. Consensus favors scoring patterns supported by multiple rubric groups, providing feedback for the current policy update alongside criterion-level verification. Its positive part is scaled by the fraction of proposed criteria passing meta-verification, yielding $\tilde{\rho}_m$. Degenerate cases and eligibility checks appear in Appendix C.

3.3 Dynamic Reward Shaping & Co-Evolution

Rubric Generator reward. The core reward combines criterion validity, discriminative value, and peer agreement:

$$R_{gen}(R_m) = \underbrace{\tilde{\rho}_m + r_{meta} + r_{disc}}_{\text{Quality \& Alignment}} + \underbrace{(-p_{num} - p_{len})}_{\text{Regularization}}. \tag{3}$$

Here, r_{meta} is the fraction of proposed criteria passing meta-verification, and r_{disc} rewards criteria that distinguish responses, scaled by this validity fraction. The penalties p_{num} and p_{len} discourage excessive criteria and overly long rubric text. Appendix C specifies batch eligibility and auxiliary regularization.

Rubric memory. Among eligible candidate batches, only the one with the highest LOO consensus contributes new criteria to the Memory Pool. Saturated historical criteria are removed, new entries are checked against the retained pool, and the oldest entries are evicted when capacity is exceeded. This selection carries useful evaluation dimensions into later rollouts while leaving the base rubrics fixed (Appendix D).

Memory connects criterion discovery across successive visits to a query, with stored criteria re-evaluated on each new response group. Retained requirements guide the Generator toward dimensions beyond the base rubrics and earlier discoveries. A criterion is saturated when every response exhibits the desired behavior: all satisfy a positive requirement or all avoid a penalized error. Such criteria can be retired to make room for new distinctions. Memory is maintained separately for each query, preserving the connection between a criterion and the task that motivated it.

Reasoner reward. The Reasoner is optimized against fixed base rubrics and evolving criteria that capture additional response-quality dimensions:

$$R_{rea}(o_i) = S_{base}(o_i) + S_{pool}(o_i) - p_{len}(o_i). \tag{4}$$

Here, S_{base} scores the response against the persistent base rubrics. S_{pool} combines the historical score and the mean eligible current-batch score, each with weight 0.5. Newly discovered criteria guide the current update, while the Memory Pool carries selected criteria into subsequent rollouts. The length penalty $p_{len}(o_i)$ increases linearly once the generated output exceeds a token-length threshold (Appendix C).

Joint update. Advantages are computed separately for the N Reasoner samples and M Generator samples of each query, and the combined rollout batch updates the shared policy. This produces a *self-evolving* loop at two levels: policy updates improve response generation and rubric construction, while memory updates retain selected criteria and retire saturated ones. The next rollout uses both the updated policy and the revised memory. Responses thus supply evidence for learning new criteria, and those criteria supply supervision for improving responses, with the base rubrics, Meta-Verifier, and Grader held fixed.

The full training procedure is given in Appendix B (Algorithm 1).

Table 1: Main results (mean ± standard deviation over three training seeds for RL methods or three response-sampling runs for Base). Avg is the unweighted five-benchmark mean. Relative gains use the strongest baseline in each column within the same backbone; bold marks the best result.

Method	Medical		Writing		Science	Avg
	HealthBench	LLMEval-Med	WritingBench	Creative Writing	ResearchQA	
Qwen3-8B						
Base	34.19 ± 0.18	49.25 ± 0.22	60.99 ± 0.16	41.05 ± 0.27	56.04 ± 0.20	48.30
Static Rubric-RL	38.89 ± 0.34	52.95 ± 0.47	65.06 ± 0.72	41.78 ± 1.08	65.83 ± 0.39	52.90
External Evolving-RL	39.53 ± 0.55	51.73 ± 0.61	65.98 ± 0.94	42.36 ± 1.27	66.34 ± 0.48	53.19
EvoRubric	$\mathbf{45.13 \pm 0.71}$	$\mathbf{53.75 \pm 0.84}$	$\mathbf{67.59 \pm 0.89}$	$\mathbf{45.91 \pm 1.34}$	$\mathbf{69.02 \pm 0.57}$	**56.28**
Relative gain (%)	**14.2** ↑	**1.5** ↑	**2.4** ↑	**8.4** ↑	**4.0** ↑	**5.8** ↑
Qwen3-14B						
Base	37.91 ± 0.14	55.86 ± 0.19	65.12 ± 0.13	52.22 ± 0.24	61.55 ± 0.17	54.53
Static Rubric-RL	44.36 ± 0.21	58.30 ± 0.25	67.13 ± 0.58	56.59 ± 0.90	68.86 ± 0.43	59.05
External Evolving-RL	44.37 ± 0.62	58.02 ± 0.32	67.40 ± 1.17	56.38 ± 1.35	69.18 ± 0.25	59.07
EvoRubric	$\mathbf{46.92 \pm 0.83}$	$\mathbf{59.23 \pm 0.91}$	$\mathbf{69.73 \pm 1.03}$	$\mathbf{58.88 \pm 1.52}$	$\mathbf{70.88 \pm 0.52}$	**61.13**
Relative gain (%)	**5.7** ↑	**1.6** ↑	**3.5** ↑	**4.0** ↑	**2.5** ↑	**3.5** ↑

4 Experiments

4.1 Experimental Setup

Models and training configuration. We instantiate the shared policy with Qwen3-8B and Qwen3-14B (Team, 2025). For each backbone, the Meta-Verifier uses the corresponding frozen pre-RL checkpoint. GPT-OSS-120B (OpenAI, 2025) serves as the frozen training Grader. Training settings are summarized in Appendix E.

Datasets and benchmarks. We train on three domains, sampling 3,000 queries each from HealthBench (Medical) (Arora et al., 2025), Long-writer-zero (Writing) (Wu et al., 2025a), and ResearchQA (Science) (Yifei et al., 2025). For each backbone, we train a separate policy in each domain and evaluate it on the corresponding benchmarks. Evaluation covers a held-out set of 500 HealthBench instances, LLMEval-Med (Zhang et al., 2025), WritingBench (Wu et al., 2025b), Creative Writing (Paech, 2025), and held-out ResearchQA queries.

Matched baselines. Static Rubric-RL uses fixed rubrics generated before RL by the corresponding frozen base model with few-shot prompting. External Evolving-RL uses a separate, unoptimized same-scale rubric generator. These controlled comparisons share the policy backbone, training data, and policy-side configuration. Following the external rubric-generation strategy of DR-Tulu (Shao et al., 2025), we also evaluate a GPT-4.1-generator variant in Medical.

Evaluation protocol. We follow each benchmark's official evaluation scripts and scoring protocol, retaining its evaluation criteria while replacing the model judge. Unless otherwise specified, evaluations use `claude-sonnet-4-6` with default settings. This third-party evaluator belongs to a different model family from both the GPT-OSS training Grader and the Qwen3 policy and Meta-Verifier, reducing direct coupling between training-time scoring and final evaluation.

4.2 Overall Performance

EvoRubric achieves the highest mean scores among matched methods on all five benchmarks at both scales (Table 1). Its macro-average scores of 56.28 and 61.13 exceed the strongest matched baselines by 3.09 and 2.06 points, corresponding to relative gains of 5.8% and 3.5%. At 8B, the largest relative improvements occur on HealthBench (14.2%) and Creative Writing (8.4%). Gains also extend to Chinese LLMEval-Med, where EvoRubric improves by 0.80 and 0.93 points at 8B and 14B. Together, these results demonstrate consistent mean improvements across Medical, Writing, and Science under an independent evaluator.

4.3 Component and Generator Analysis

Table 2 examines individual training components. The w/o Generator Loss variant updates the shared policy using only the Reasoner loss, retaining rubric generation, verification, grading, memory, and all selection rules. Rubric generation can therefore still change indirectly through shared parameters. Other variants remove verifier screening and feedback, historical memory, peer consensus, or discriminative feedback; the memory ablation retains current-rollout supervision.

Table 2: Medical component ablations (Qwen3-8B), averaged over 3 matched runs.

Model Variant	HealthBench	LLMEval-Med
w/o Generator Loss	41.58	49.93
w/o Discriminative Feedback	38.92	51.11
w/o Peer Consensus	40.80	50.16
w/o Memory Pool	42.26	52.41
w/o Meta-Verifier	43.13	47.81
EvoRubric (Full)	**45.13**	**53.75**

Removing the Generator loss lowers HealthBench and LLMEval-Med by 3.55 and 3.82 points, respectively. This supports the contribution of explicit rubric learning beyond the indirect changes induced by Reasoner updates. Every other ablation also lowers both scores: discriminative feedback has the largest effect on HealthBench, while removing meta-verification reduces LLMEval-Med to 47.81, below the base model's 49.25. The final-pool audit (Table 5) links the latter decline to a higher incidence of invalid criteria. Removing peer consensus and memory reduces HealthBench by 4.33 and 2.87 points, respectively. These components support different aspects of adaptive supervision: consensus favors agreement among rubric groups on the current responses, whereas memory carries selected criteria into later rollouts for re-evaluation. The two ablations connect these mechanisms to improvements in final benchmark performance as the response distribution changes. Together, these results support both direct Generator optimization and the complementary rubric-quality signals.

External generation following DR-Tulu. We instantiate DR-Tulu's external rubric-generation strategy (Shao et al., 2025) in our Medical setting by using `gpt-4.1-2025-04-14` to propose evolving criteria from current responses and existing rubrics. With the same Qwen3-8B Reasoner backbone, the GPT-4.1 generator improves HealthBench/LLMEval-Med by 2.72/1.19 points over the frozen same-scale generator (Table 3). EvoRubric gains a further 2.88/0.83 points, showing that rubric construction learned within an 8B policy can provide more effective training supervision than this strong external generator. Generator usage and cost are reported in Appendix H.

Table 3: Rubric-generation strategies in Medical (Qwen3-8B; means over 3 runs). The GPT-4.1 variant adopts DR-Tulu's external evolving-rubric generation strategy within our training setting.

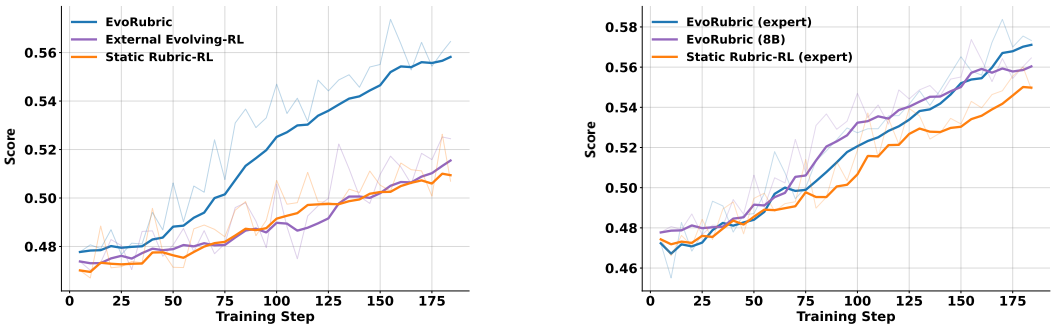
Method	Rubric generator	HealthBench	LLMEval-Med
External Evolving-RL	Frozen Qwen3-8B	39.53	51.73
External (DR-Tulu-style)	GPT-4.1	42.25	52.92
EvoRubric	**Shared learned policy**	**45.13**	**53.75**

4.4 Training Dynamics and Rubric Initialization

4.4.1 Training dynamics

Figure 3 shows that, with model-generated base rubrics, EvoRubric reaches higher scores than the static and external evolving-rubric baselines. With expert-initialized rubrics, co-evolution also improves the learning trajectory, motivating the initialization comparison below.

The trajectories begin at similar score levels and separate as training proceeds, with EvoRubric maintaining a larger advantage in the later stages. Because the base rubrics remain fixed within each initialization setting, these curves track progress against a stable evaluation target while the Memory Pool evolves. The held-out initialization results below examine whether this training-time advantage carries over to final response quality.



(a) Standard training dynamics

(b) Impact of expert base rubrics

Figure 3: Medical learning curves showing GPT-OSS-120B scores under base rubrics held fixed throughout RL. (a) Model-generated base rubrics. (b) Expert versus model initialization.

4.4.2 Expert-initialized rubrics

Expert initialization raises Static Rubric-RL from 38.89 to 41.83 (Table 4), and co-evolution further improves the expert-initialized score to 44.64 (+2.81 points). EvoRubric therefore extends the benefits of expert priors by discovering additional criteria from evolving responses.

Table 4: Rubric initialization on HealthBench (Qwen3-8B; means over 3 runs).

Method	HealthBench
Static Rubric-RL (model)	38.89
Static Rubric-RL (expert)	41.83
EvoRubric (expert)	44.64
EvoRubric (model)	**45.13**

Interestingly, model-initialized EvoRubric reaches 45.13, slightly exceeding expert initialization by 0.49 points in this experiment. The results show that co-evolution is effective with both sources of initial criteria and can develop useful supervision from the model's own rubrics.

4.5 Rubric and Human Evaluation

Validity and diversity over training. Three annotators with medical-rubric annotation experience audited 20 Medical prompts over four training epochs. Table 5 reports final-pool validity and effective diversity: the number of human-valid, semantically distinct dimensions beyond the base rubrics, divided by all pool entries. Semantically equivalent criteria count once per prompt in the numerator; the denominator retains all entries.

Effective diversity decreases from 89.19% to 75.16% between the first and fourth epoch-end snapshots at 8B, and from 91.62% to 76.26% at 14B. The final pools retain 118 and 106 distinct, human-valid dimensions beyond the base rubrics, respectively. The audit characterizes retained rubric coverage, complementing the memory ablation's assessment of its contribution to benchmark performance.

Meta-verification as a validity safeguard. Where LOO measures agreement, meta-verification checks the validity of the criteria on which that agreement is based. At 8B, verification reduces factual conflicts from 15.63% to 2.55% and unsupported style requirements from 5.21% to 1.27%. No length-hallucination criteria were observed with verification, compared with 5.21% without it. The overall invalid rate falls from 39.06% to 5.10%, an 86.96% relative reduction computed from the underlying counts. These reductions support meta-verification as a safeguard against invalid criteria entering the supervision, complementing the consistency feedback supplied by LOO consensus.

Human response preferences. Table 6 reports a separate blind comparison of final responses. Overall-quality wins exceed losses against both external-generator baselines. The largest advantages occur in conciseness and relevance, whereas completeness favors the baselines more often, factual-

Table 5: Final-pool audit over 20 Medical prompts: counts (%), with totals N in the header. Effective diversity counts distinct, human-valid dimensions beyond the base rubrics using the same denominators. Percentages are rounded independently; dashes denote unreported results.

Measure	8B $N = 157$	14B $N = 139$	8B w/o verifier $N = 192$
Factual conflict	4 (2.55)	3 (2.16)	30 (15.63)
Semantic duplication	2 (1.27)	1 (0.72)	15 (7.81)
Style hallucination	2 (1.27)	2 (1.44)	10 (5.21)
Unmeasurable	0 (0.00)	0 (0.00)	5 (2.60)
Trivial/cliche	0 (0.00)	0 (0.00)	5 (2.60)
Length hallucination	0 (0.00)	0 (0.00)	10 (5.21)
Invalid criteria	8 (5.10)	6 (4.32)	75 (39.06)
Valid criteria	149 (94.90)	133 (95.68)	117 (60.94)
Effective diversity of the final pool	118 (75.16)	106 (76.26)	—

correctness outcomes are close, and safety comparisons are mostly ties. The annotation protocol and longitudinal audit details are given in Appendix I.

Table 6: Human response preferences (prompt-level win/tie/loss, %; 50 Medical prompts). A win favors EvoRubric-8B.

Dimension	vs. External	vs. External + GPT-4.1
Completeness	20 / 52 / 28	36 / 18 / 46
Factual correctness	40 / 22 / 38	42 / 14 / 44
Conciseness	54 / 36 / 10	68 / 18 / 14
Relevance	58 / 30 / 12	62 / 26 / 12
Safety	16 / 76 / 8	16 / 68 / 16
Overall quality	42 / 30 / 28	48 / 14 / 38

Rubric quality under fixed responses. To examine rubric construction with response quality held constant, we compare the two generators on 100 HealthBench queries, each paired with three fixed responses from an unaligned base model. Using the default evaluator from Section 4.1, a blind pairwise comparison prefers EvoRubric in 72% of cases and the same-scale external generator in 28%, based on professionalism, granularity, and discriminative value (Appendix G). This comparison supports improved rubric construction on identical response groups, complementing the end-to-end benchmark results.

What the learned rubrics capture. For the insulin-storage case in Appendix F, the base rubrics cover general storage conditions. Differences among responses motivate additional checks for formulation-specific durations, handling, device-specific instructions, and source verification. The example illustrates how response variation becomes explicit evaluation criteria for subsequent learning.

5 Conclusion

We propose **EvoRubric**, which constructs rubric-learning supervision for open-ended generation without a verifiable final-answer correctness target. A shared policy discovers evaluation dimensions from its current responses and improves through the supervision those dimensions provide. A frozen copy of the initial policy supplies criterion-validity feedback, while grading, variance filtering, peer consensus, and persistent rubric memory support the joint optimization of the Reasoner and Rubric Generator. Across five benchmarks in Medical, Writing, and Science, EvoRubric outperforms matched static and external evolving-rubric baselines, with macro-average improvements of 3.09 points at 8B and 2.06 points at 14B over the strongest matched baselines across three training seeds. Rubric audits, response comparisons, and expert-initialization experiments provide complementary evidence for this self-evolving approach to open-ended alignment.

AI USE STATEMENT

During manuscript preparation, we used large language models for language polishing, grammar correction, and structural refinement of author-written drafts. The research methodology, experiments, and scientific conclusions remain the responsibility of the human authors. The authors reviewed AI-assisted textual edits for accuracy and consistency with the underlying work.

ETHICS STATEMENT

Our experiments use existing models and benchmark datasets, with human evaluation by annotators experienced in medical-rubric annotation. Generated rubrics may inherit model or data biases. Medical evaluations assess benchmark performance and do not establish clinical safety. Reuse of these resources should follow applicable licenses and usage restrictions.

REPRODUCIBILITY STATEMENT

Code, training data, and launch configurations are available at https://anonymous.4open.science/r/EvoRubric-8FF9. The experimental setup and appendices provide evaluation protocols and implementation details.

REFERENCES

Rahul K. Arora, Jason Wei, Rebecca Soskin Hicks, Preston Bowman, Joaquin Quiñonero Candela, Foivos Tsimpourlas, Michael Sharman, Meghan Shah, Andrea Vallone, Alex Beutel, Johannes Heidecke, and Karan Singhal. Healthbench: Evaluating large language models towards improved human health. *CoRR*, abs/2505.08775, 2025. doi: 10.48550/ARXIV.2505.08775. URL https://doi.org/10.48550/arXiv.2505.08775.

Shreyas Chaudhari, Pranjal Aggarwal, Vishvak Murahari, Tanmay Rajpurohit, Ashwin Kalyan, Karthik Narasimhan, Ameet Deshpande, and Bruno Castro da Silva. RLHF deciphered: A critical analysis of reinforcement learning from human feedback for llms. *ACM Comput. Surv.*, 58(2): 53:1–53:37, 2026. doi: 10.1145/3743127. URL https://doi.org/10.1145/3743127.

Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. Safe RLHF: safe reinforcement learning from human feedback. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=TyFrPOKYXw.

Mingxuan Du, Benfeng Xu, Chiwei Zhu, Xiaorui Wang, and Zhendong Mao. Deepresearch bench: A comprehensive benchmark for deep research agents. *CoRR*, abs/2506.11763, 2025. doi: 10.48550/ARXIV.2506.11763. URL https://doi.org/10.48550/arXiv.2506.11763.

Jiayi Fu, Xuandong Zhao, Chengyuan Yao, Heng Wang, Qi Han, and Yanghua Xiao. Reward shaping to mitigate reward hacking in RLHF. *CoRR*, abs/2502.18770, 2025. doi: 10.48550/ARXIV.2502.18770. URL https://doi.org/10.48550/arXiv.2502.18770.

Tianyu Gao, Alexander Wettig, Howard Yen, and Danqi Chen. How to train long-context language models (effectively). In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pp. 7376–7399. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.acl-long.366/.

Xin Guan, Zijian Li, Shen Huang, Pengjun Xie, Jingren Zhou, and Jiuxin Cao. Evidence-augmented policy optimization with reward co-evolution for long-context reasoning. *CoRR*, abs/2601.10306, 2026. doi: 10.48550/ARXIV.2601.10306. URL https://doi.org/10.48550/arXiv.2601.10306.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean M. Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. In *The Fourteenth International Conference on Learning Representations, ICLR 2026*. OpenReview.net, 2026. URL https://openreview.net/forum?id=c1bTcrDmt4.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nat.*, 645(8081):633–638, 2025. doi: 10.1038/S41586-025-09422-Z. URL https://doi.org/10.1038/s41586-025-09422-z.

Yun He, Wenzhe Li, Hejia Zhang, Songlin Li, Karishma Mandyam, Sopan Khosla, Yuanhao Xiong, Nanshu Wang, Xiaoliang Peng, Beibin Li, Shengjie Bi, Shishir G. Patil, Qi Qi, Shengyu Feng, Julian Katz-Samuels, Richard Yuanzhe Pang, Sujan Gonugondla, Hunter Lang, Yue Yu, Yundi Qian, Maryam Fazel-Zarandi, Licheng Yu, Amine Benhalloum, Hany Awadalla, and Manaal Faruqui. Advancedif: Rubric-based benchmarking and reinforcement learning for advancing LLM instruction following. *CoRR*, abs/2511.10507, 2025. doi: 10.48550/ARXIV.2511.10507. URL https://doi.org/10.48550/arXiv.2511.10507.

Zenan Huang, Yihong Zhuang, Guoshan Lu, Zeyu Qin, Haokai Xu, Tianyu Zhao, Ru Peng, Jiaqi Hu, Zhanming Shen, Xiaomeng Hu, Xijun Gu, Peiyi Tu, Jiaxin Liu, Wenyu Chen, Yuzhuo Fu, Zhiting Fan, Yanmei Gu, Yuanyuan Wang, Zhengkai Yang, Jianguo Li, and Junbo Zhao. Reinforcement learning with rubric anchors. *CoRR*, abs/2508.12790, 2025. doi: 10.48550/ARXIV.2508.12790. URL https://doi.org/10.48550/arXiv.2508.12790.

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. *CoRR*, abs/2601.08430, 2026. doi: 10.48550/ARXIV.2601.08430. URL https://doi.org/10.48550/arXiv.2601.08430.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/e099c1c9699814af0be873a175361713-Abstract-Conference.html.

OpenAI. Openai o1 system card. *CoRR*, abs/2412.16720, 2024. doi: 10.48550/ARXIV.2412.16720. URL https://doi.org/10.48550/arXiv.2412.16720.

OpenAI. gpt-oss-120b & gpt-oss-20b model card. *CoRR*, abs/2508.10925, 2025. doi: 10.48550/ARXIV.2508.10925. URL https://doi.org/10.48550/arXiv.2508.10925.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html.

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

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. Yarn: Efficient context window extension of large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=wHBfxhZu1u.

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. URL https://openreview.net/forum?id=Ti67584b98.

Rulin Shao, Akari Asai, Shannon Zejiang Shen, Hamish Ivison, Varsha Kishore, Jingming Zhuo, Xinran Zhao, Molly Park, Samuel G. Finlayson, David A. Sontag, Tyler Murray, Sewon Min, Pradeep Dasigi, Luca Soldaini, Faeze Brahman, Wen-tau Yih, Tongshuang Wu, Luke Zettlemoyer, Yoon Kim, Hannaneh Hajishirzi, and Pang Wei Koh. DR tulu: Reinforcement learning with evolving rubrics for deep research. *CoRR*, abs/2511.19399, 2025. doi: 10.48550/ARXIV.2511.19399. URL https://doi.org/10.48550/arXiv.2511.19399.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/ARXIV.2402.03300. URL https://doi.org/10.48550/arXiv.2402.03300.

Leheng Sheng, Wenchang Ma, Ruixin Hong, Xiang Wang, An Zhang, and Tat-Seng Chua. Reinforcing chain-of-thought reasoning with self-evolving rubrics. *CoRR*, abs/2602.10885, 2026. doi: 10.48550/ARXIV.2602.10885. URL https://doi.org/10.48550/arXiv.2602.10885.

Hwanjun Song, Hang Su, Igor Shalyminov, Jason Cai, and Saab Mansour. Finesure: Fine-grained summarization evaluation using llms. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 906–922. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.51. URL https://doi.org/10.18653/v1/2024.acl-long.51.

Qwen Team. Qwen3 technical report. *CoRR*, abs/2505.09388, 2025. doi: 10.48550/ARXIV.2505.09388. URL https://doi.org/10.48550/arXiv.2505.09388.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multi-hop questions via single-hop question composition. *Trans. Assoc. Comput. Linguistics*, 10:539–554, 2022. doi: 10.1162/TACL_A_00475. URL https://doi.org/10.1162/tacl_a_00475.

Xiyu Wei, Qingwei Zong, Xiaoguang Li, Eugene J. Yu, and Sujian Li. Qurl: Rubrics as judge for open-ended question answering. In *The Fourteenth International Conference on Learning Representations, ICLR 2026*. OpenReview.net, 2026. URL https://openreview.net/forum?id=DrhWTuhtYq.

Yuhao Wu, Yushi Bai, Zhiqiang Hu, Roy Ka-Wei Lee, and Juanzi Li. Longwriter-zero: Mastering ultra-long text generation via reinforcement learning. *CoRR*, abs/2506.18841, 2025a. doi: 10.48550/ARXIV.2506.18841. URL https://doi.org/10.48550/arXiv.2506.18841.

Yuning Wu, Jiahao Mei, Ming Yan, Chenliang Li, Shaopeng Lai, Yuran Ren, Zijia Wang, Ji Zhang, Mengyue Wu, Qin Jin, and Fei Huang. Writingbench: A comprehensive benchmark for generative writing. *CoRR*, abs/2503.05244, 2025b. doi: 10.48550/ARXIV.2503.05244. URL https://doi.org/10.48550/arXiv.2503.05244.

Ran Xu, Tianci Liu, Zihan Dong, Tony Yu, Ilgee Hong, Carl Yang, Linjun Zhang, Tao Zhao, and Haoyu Wang. Alternating reinforcement learning for rubric-based reward modeling in non-verifiable LLM post-training. *CoRR*, abs/2602.01511, 2026. doi: 10.48550/ARXIV.2602.01511. URL https://doi.org/10.48550/arXiv.2602.01511.

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. *CoRR*, abs/2509.00496, 2025. doi: 10.48550/ARXIV.2509.00496. URL https://doi.org/10.48550/arXiv.2509.00496.

Ming Zhang, Yujiong Shen, Zelin Li, Huayu Sha, Binze Hu, Yuhui Wang, Chenhao Huang, Shichun Liu, Jingqi Tong, Changhao Jiang, Mingxu Chai, Zhiheng Xi, Shihan Dou, Tao Gui, Qi Zhang, and Xuanjing Huang. Llmeval-med: A real-world clinical benchmark for medical llms with physician validation. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 4888–4914. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.findings-emnlp.263/.

A Comparison with Related Methods

We compare rubric-based learning paradigms along two dimensions: how evaluation criteria are updated and whether their generation is jointly optimized with the response policy.

Static rubric-based RL. Static rubric methods construct evaluation criteria before policy training and use them to score subsequent responses (He et al., 2025; Wei et al., 2026; Gunjal et al., 2026). EvoRubric retains persistent base rubrics and augments them with criteria discovered from current policy responses. A shared policy learns to generate both responses and new criteria, while the Memory Pool carries selected criteria across iterations.

Externally generated evolving rubrics. DR-Tulu (Shao et al., 2025) generates criteria from current responses and existing rubrics, maintains a rubric pool, and filters uninformative criteria. Its reported RL setup uses GPT-4.1 for evolving-rubric generation. EvoRubric learns criterion construction within the response policy, adding validity and peer-agreement feedback. Our GPT-4.1-generator variant follows this external-generation design within the Medical comparison; EvoRubric achieves gains of 2.88/0.83 points on HealthBench/LLMEval-Med (Table 3). This comparison evaluates the rubric-generation strategy within our training setting. The full DR-Tulu system additionally includes search and citation training.

Joint response and rubric learning. RLCER (Sheng et al., 2026) shares parameters across a Reasoner and Rubricator, with role-specific rewards and joint updates. It selects rubrics using their association with final-answer correctness. EvoRubric addresses the absence of that reference: frozen verification assesses criterion validity, discriminative feedback measures response separation, and LOO consensus measures agreement across rubric batches. These provide complementary training signals; consensus itself does not establish correctness.

Joint rubric and judge learning. Rubric-ARM (Xu et al., 2026) alternately trains the rubric generator and judge using preference supervision. EvoRubric updates the Generator and Reasoner while keeping the Grader fixed. The distinction concerns both which components learn and where their supervision comes from.

B Training Loop and Joint GRPO Objective

Algorithm 1 summarizes the training loop. Historical criteria used for current-response scoring are retrieved before the memory update; selected additions are carried forward to the next visit to that query.

For each query, the rollout policy $\pi_{\theta_{old}}$ samples N Reasoner outputs and M Generator outputs. Advantages are standardized separately within each query–role group. Let $\mathcal{B}$ be the combined rollout batch, x_b the role-specific input, and y_b the sampled output. The implementation averages the clipped objective over response tokens:

$$J(\theta) = \mathbb{E}\left[\frac{1}{\sum_{b\in\mathcal{B}}|y_b|}\sum_{b\in\mathcal{B}}\sum_{t=1}^{|y_b|}\min\left(\rho_{b,t}\hat{A}_b, \text{clip}(\rho_{b,t}, 1-\epsilon_{low}, 1+\epsilon_{high})\hat{A}_b\right)\right], \tag{5}$$

where $\rho_{b,t} = \pi_\theta(y_{b,t} \mid x_b, y_{b,<t})/\pi_{\theta_{old}}(y_{b,t} \mid x_b, y_{b,<t})$. Both roles contribute to one shared policy update; their rewards are not standardized together. Adding a numerical stabilizer to the group standard deviation gives zero advantages for constant-reward groups.

C Reward Computation

Base and pool scores. For a criterion set $\mathcal{C}$, let w_c be the signed weight of criterion c and $z_c(o) \in \{0,1\}$ indicate whether o meets it. For a negative-weight criterion, $z_c(o) = 1$ means that the penalized behavior occurs. Define

$$A(o;\mathcal{C}) = \sum_{c\in\mathcal{C}} w_c z_c(o), \qquad P(\mathcal{C}) = \sum_{c\in\mathcal{C}} \max(w_c, 0), \qquad N(\mathcal{C}) = \sum_{c\in\mathcal{C}} \max(-w_c, 0).$$

Algorithm 1 EvoRubric training loop

Require: Training queries $\mathcal{D}$ with fixed base rubrics $\mathcal{C}_{base}(q)$; initial policy π_{θ_0}; frozen Grader; rollout sizes N, M; pool capacity C_{max}

Initialize $\theta \leftarrow \theta_0$ and per-query memory $\mathcal{M}_q \leftarrow \emptyset$; freeze the Meta-Verifier at π_{θ_0}
while training continues **do**
  Sample a minibatch of queries from $\mathcal{D}$
  for each query q in the minibatch **do**
    Retrieve $\mathcal{C}_{hist} \leftarrow \mathcal{M}_q$ and sample N Reasoner responses $\mathcal{O}$
    Sample M rubric sets conditioned on $q, \mathcal{O}, \mathcal{C}_{base}(q), \mathcal{C}_{hist}$
    Meta-verify the proposed criteria to obtain $\{\tilde{R}_m\}_{m=1}^{M}$
    Grade responses against base, historical, and verified candidate criteria
    Compute discriminative and LOO-consensus feedback; identify eligible batches
    Compute R_{gen} and R_{rea} using base, historical, and eligible current feedback
    Remove saturated entries from $\mathcal{M}_q$
    if at least one candidate batch is eligible **then**
      Admit criteria from the highest-consensus eligible batch after duplicate checks
    end if
    Evict oldest entries until $|\mathcal{M}_q| \leq C_{max}$
    Standardize advantages separately for this query's Reasoner and Generator groups
  end for
  Update θ with the combined rollout batch using the clipped GRPO objective
end while

All base, historical, and current rubric sets use the same normalized score:

$$F(o;\mathcal{C}) = \begin{cases} \dfrac{A(o;\mathcal{C})}{\max\{P(\mathcal{C}), N(\mathcal{C})\}}, & \max\{P(\mathcal{C}), N(\mathcal{C})\} > 0, \\ 0, & \text{otherwise.} \end{cases}$$

The denominator includes all weights in the scored set and is shared across responses. Thus $F \in [-1, 1]$, including sets containing only negative weights; empty or zero-weight sets score zero.

Let $\mathcal{H}$ contain the historical criteria that distinguish the current responses and let $\mathcal{E}$ index the eligible current rubric batches. Their mean score is

$$S_{current}(o) = \begin{cases} \dfrac{1}{|\mathcal{E}|} \displaystyle\sum_{m \in \mathcal{E}} F(o; \tilde{R}_m), & |\mathcal{E}| > 0, \\ 0, & |\mathcal{E}| = 0. \end{cases}$$

The two sources of Reasoner supervision are

$$S_{base}(o) = F(o; \mathcal{C}_{base}), \qquad S_{pool}(o) = \tfrac{1}{2} F(o; \mathcal{H}) + \tfrac{1}{2} S_{current}(o).$$

The historical and current terms retain their fixed weights when either is absent. Both S_{base} and S_{pool} lie in $[-1, 1]$ and enter $R_{rea} = S_{base} + S_{pool} - p_{len}$. Current supervision averages all eligible batches; only the highest-consensus eligible batch contributes new entries to persistent memory.

Consensus and validity feedback. The Meta-Verifier assigns $v(r) = 1$ to VALID criteria and $v(r) = 0$ to INVALID criteria. For a batch R_m containing K_m proposed criteria before filtering, the Generator's validity reward is

$$r_{meta}(R_m) = \frac{\sum_{k=1}^{K_m} v(r_{m,k})}{\max\{1, K_m\}}.$$

This is the fraction of valid criteria among all proposals, with zero assigned to an empty batch. Only valid criteria contribute to subsequent response scoring. Peer consensus uses nonconstant candidate-batch score vectors and is zero when there are no other nonconstant batches or their mean vector is constant. Its positive part is scaled by this validity fraction. The discriminative component measures the fraction of verified criteria that split the responses and is likewise scaled by the validity fraction.

Candidate handling. Malformed outputs receive a format reward of -2. Empty or fully rejected batches contribute no new criteria. A verified candidate batch is eligible when its response-score vector is nonconstant and its LOO correlation exceeds 0.2.

Auxiliary regularization. For K proposed criteria, the count penalty is $p_{num} = \min\{1, \max\{0, (K-10)/10\}\}$, penalizing outputs beyond the prompted limit of 10 criteria. An additional point-concentration penalty is subtracted from the Generator reward: for $K \geq 3$, it is $0.5 \max\{0, (f-0.6)/0.4\}$, where f is the fraction of criteria sharing the most frequent absolute weight; it is zero for $K < 3$.

Length penalty. For a generated output y of token length $\ell(y)$, the overlong penalty is

$$p_{len}(y) = \alpha \max\left(0, \frac{\ell(y) - (L_{max} - L_{buffer})}{L_{buffer}}\right).$$

Here, $L_{max} = 8192$, $L_{buffer} = 4096$, and $\alpha = 1$. The penalty is zero up to 4096 tokens and grows linearly thereafter. It is subtracted from the reward for both Reasoner and Generator outputs.

D Memory Pool and Initialization Details

Each rollout samples $N = 8$ responses and $M = 4$ candidate rubric sets per prompt. The Generator receives the fixed base rubrics and all stored historical criteria. For each query, only the eligible batch with the highest LOO consensus can add criteria to the Memory Pool. Saturated historical criteria are removed; candidate entries are checked for exact textual matches against the retained pool before insertion. The pool retains at most 10 entries, evicting the oldest when necessary. Historical criteria that do not distinguish the current responses are excluded from the current historical-pool score. Memory selection retains one batch for future rollouts, whereas current-response supervision aggregates scores from all eligible batches.

Persistent base rubrics are stored separately and remain fixed throughout RL. They are generated once using few-shot prompting with the corresponding frozen pre-RL Qwen3 model. The number of base criteria per prompt ranges from 2 to 8; the proportions for 2, 3, 4, 5, 6, 7, and 8 criteria are 0.03%, 0.48%, 4.35%, 23.17%, 38.96%, 13.38%, and 19.63%, respectively.

E Hyperparameters for Reinforcement Learning

In this section, we detail the hyperparameters and configurations used during the reinforcement learning (RL) training phase of our framework. Table 7 provides a comprehensive list of the standard training settings extracted from our experimental setup, including the specific proxy model utilized as the Grader.

Table 7: Hyperparameters and configurations for reinforcement learning training.

Parameter	Value
Evaluation Configurations	
Grader	`GPT-OSS-120B` (OpenAI, 2025)
GRPO & Rollout Parameters	
Rollout Samples per Prompt (N)	8
Rubric Sets per Prompt (M)	4
Evolved Pool Capacity per Prompt	10
Pool Replacement	FIFO
GRPO Clip Bounds (ϵ_{low}, ϵ_{high})	0.2, 0.28
KL Loss Coefficient	0.0
Training Temperature	0.7
Training Top-P	0.8
Training Top-K	20
Max Prompt Length	12,288
Max Response Length	8,192
Optimization Parameters	
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
Reward Shaping Parameters	
Overlong Buffer Length	4,096
Overlong Penalty Factor	1.0
Hardware Setup	
GPUs	8 $\times$ NVIDIA H20

F Qualitative Case Study: Rubric Evolution

Table 8 illustrates rubric evolution for a HealthBench query about insulin storage. The fixed base rubrics cover general temperature, duration, and storage conditions. Differences among the sampled responses motivate additional criteria about formulation-specific duration, handling, device-specific instructions, and source verification. The example illustrates how response-conditioned rubric generation extends the coverage of fixed base criteria.

Table 8: Qualitative case study: generated rubrics add response-specific evaluation dimensions to the fixed base criteria.

User Query: I am a diabetic on insulin. How do I store it?

Existing Base Rubrics

- Correctly explains that unopened insulin should be refrigerated at 2–8°C, and that opened insulin can generally be stored at room temperature for up to 28 days, depending on the formulation. **(+10)**
- Incorrectly states that insulin can be frozen or exposed to temperatures above 86°F (30°C), which can degrade the medication. **(-10)**
- Mentions avoiding direct sunlight and heat sources during storage. **(+7)**

Reasoner Responses (Selected Snippets Highlighting Diversity)
Response Snippet A: "...Long-acting insulins (e.g., glargine [Lantus], detemir [Levemir], degludec [Tresiba]): Degludec (Tresiba) can be stored for up to 56 days at room temperature according to manufacturer instructions... Cloudy insulins (e.g., NPH) should be gently rolled, not shaken, before use to mix the suspension evenly..."
Response Snippet B: "...If using an insulin pen, store it at room temperature after first use unless the product label explicitly states otherwise. Do not refrigerate opened insulin pens unless instructed by the manufacturer or your healthcare provider..."

Criteria Generated by EvoRubric

- **[Type-Specific Duration]** Provides precise storage durations for different insulin types or formulations (e.g., rapid-acting insulin is often limited to around 28 days, whereas some long-acting formulations such as insulin degludec may allow up to 56 days), rather than relying on a generic 28-day rule. **(+7)**
- **[Handling Nuance]** Explicitly advises against shaking cloudy insulin vials or pens and instead recommends gently rolling them when mixing is required, helping avoid foaming, improper suspension, or dosing inaccuracy. **(+6)**
- **[Device-Specific Rule]** Specifies that in-use insulin pens should generally be stored according to the product label, often at room temperature after first use, while avoiding unnecessary refrigeration unless the manufacturer explicitly permits or recommends it. **(+6)**
- **[Source Verification]** Mentions consulting the manufacturer's package insert, pharmacist, or healthcare provider for formulation-specific storage instructions, acknowledging that storage durations and conditions vary across insulin products. **(+5)**

G Controlled Rubric Quality Comparison

For 100 randomly sampled HealthBench queries, an unaligned base model generates three fixed responses per query. The same-scale External Evolving-RL generator and EvoRubric generate rubrics for identical query-response groups. In a blind pairwise comparison, the default evaluator specified in Section 4.1 selects the more professional, granular, and discriminative rubric set, preferring EvoRubric in 72% of cases and the external baseline in 28%. Holding the query and response group fixed isolates differences in the generated rubric sets. A qualitative example appears in Appendix F.

H Effect of External Generator Capacity

We record generator API usage for the GPT-4.1 variant in Table 3. Averaged across the three runs, GPT-4.1 rubric generation consumed 81.20 million input tokens and 2.91 million output tokens per run (84.11 million total per run), costing approximately USD 185.70 per run at the pricing used for the experiment. This estimate covers generator API usage.

I Human Evaluation and Rubric Dynamics

I.1 Longitudinal rubric audit

Three annotators with experience in medical-rubric annotation independently audited the same 20 randomly sampled Medical prompts for both model scales. The audit covered generated criteria, Meta-Verifier decisions, and the evolving Memory Pool across all four RL epochs, including the 8B variant without Meta-Verification. Criterion validity was determined by majority vote. Invalid rates are computed by pooling criterion counts across the 20 prompts at each epoch-end Memory Pool snapshot. Pool error and validity percentages use the underlying integer counts and are rounded to two decimal places using round-half-up.

At epoch 4, the invalid rate is 8/157 with verification and 75/192 without it at 8B, an 86.96% relative reduction. Final valid-criterion rates are 94.90% (8B), 95.68% (14B), and 60.94% (8B without

Table 9: Epoch-end Memory Pool audit over 20 Medical prompts. Each entry gives invalid criteria / all audited pool entries (%). Epoch 4 corresponds to Table 5.

Setting	Epoch 1	Epoch 2	Epoch 3	Epoch 4
8B	2/185 (1.08)	4/199 (2.01)	8/177 (4.52)	8/157 (5.10)
14B	2/167 (1.20)	3/191 (1.57)	7/170 (4.12)	6/139 (4.32)
8B w/o verifier	18/149 (12.08)	32/132 (24.24)	52/157 (33.12)	75/192 (39.06)

verification). Residual errors increase over training even with verification, indicating that some invalid criteria persist in the pool. Verifier–human agreement is 78.5% at 8B and 81.2% at 14B; it is unavailable without verification. The 14B setting has higher agreement and a lower final invalid rate, although both the policy and verifier scale change in this comparison.

The residual error categories and final validity rates are reported in Table 5 in the main text.

The audit distinguishes factual conflicts, semantic duplication, unsupported style or length requirements, criteria that cannot be measured, and trivial criteria. Factual conflicts remain the largest residual category with verification. All six categories occur without verification; unmeasurable, trivial, and length-hallucination errors were not observed in the final verified pools of this audit.

I.2 Effective diversity of the Memory Pool

We measure effective diversity on the same epoch-end pool snapshots used for the validity audit. For each audited prompt $q \in \mathcal{Q}$, let $\mathcal{M}_q^{(e)}$ be its pool at the end of epoch e, and let $\mathcal{U}_q^{(e)}$ contain one representative of each semantically distinct, human-valid evaluation dimension in that pool that is absent from the prompt's fixed base rubrics. We compute

$$D_e = \frac{\sum_{q \in \mathcal{Q}} |\mathcal{U}_q^{(e)}|}{\sum_{q \in \mathcal{Q}} |\mathcal{M}_q^{(e)}|}.$$

Semantically equivalent eligible entries count once per prompt in the numerator, while all pool entries remain in the denominator. Counts are summed across the 20 prompts before division; criteria from different prompts are not deduplicated together. This within-pool grouping is distinct from the Meta-Verifier's Semantic Duplication check against base rubrics (Appendix K). Each snapshot measures the dimensions retained at that epoch, including criteria carried over from earlier epochs.

Table 10: Effective diversity of epoch-end Memory Pools. Entries give distinct, human-valid dimensions beyond base / all pool entries (%), using the same denominators as Table 9.

Setting	Epoch 1	Epoch 2	Epoch 3	Epoch 4
8B	165/185 (89.19)	172/199 (86.43)	143/177 (80.79)	118/157 (75.16)
14B	153/167 (91.62)	168/191 (87.96)	139/170 (81.76)	106/139 (76.26)

Effective diversity declines across the four snapshots, with 14B remaining slightly higher at each epoch. At epoch 4, distinct, human-valid dimensions beyond the base rubrics account for 75.16% and 76.26% of the 8B and 14B pool sizes. The final pools thus retain substantial non-redundant coverage beyond the fixed base criteria.

I.3 Blind evaluation of final responses

The same three annotators evaluated responses on 50 randomly sampled Medical prompts in a blind pairwise comparison. We compare EvoRubric-8B against the same-scale External Evolving-RL baseline and the stronger GPT-4.1-generator variant. The dimensions are completeness (coverage of relevant aspects), factual correctness (accuracy without fabrication), conciseness (absence of unnecessary content), relevance (focus on the question), safety (medically appropriate, non-harmful information), and overall quality. For each prompt and dimension, a judgment supported by at

least two annotators is adopted; a three-way split among win, tie, and loss is recorded as a tie. The dimension-wise results are reported in Table 6 in the main text.

Overall-quality wins exceed losses against both baselines, corresponding to non-tie win rates of 60.0% and 55.8%. The largest advantages are in conciseness and relevance, while completeness favors the baselines more often and factual-correctness outcomes are close. Safety comparisons are mostly ties. These results characterize response-quality trade-offs within the 50-prompt evaluation.

J Scope and Limitations

Our experiments cover two Qwen3 scales and three text domains, with component ablations and human evaluation focused on Medical. Rubric generation and selection use the same sampled responses, so peer agreement may reflect shared preferences specific to those responses. Frozen verification and base criteria provide additional checks, while the human audit observes residual invalid criteria. Independent judging reduces direct evaluator coupling but does not remove model-judge bias. The 20-prompt rubric audit and 50-prompt response comparison provide descriptive evidence of rubric validity and response preferences; clinical safety and broader generalization require larger expert studies and experiments with additional backbones and languages.

K Prompt Templates

Table 11: Prompt template for the Rubric Generator.

System Prompt

You are a world-class Rubric Design Specialist for AI Alignment, specializing in Reinforcement Learning from Human Feedback (RLHF).
Core Mission
Your mission is to analyze a Question, a set of Model Responses, and Existing Rubrics to generate new, highly discriminative evaluation criteria. You must identify critical quality dimensions or specific errors in the text that are **currently missing** from the existing set of criteria. Your goal is to ensure that any meaningful difference between the responses is captured by a specific rubric.
Input Data
You will be provided with:
1. **Question**: The original prompt given to the models.
2. **Responses**: A set of text outputs generated by the models.
3. **Existing Rubrics**: The current list of criteria already in use.
Core Principles: You Must Adhere to These Rules
1. Focus on Discriminative Features (The "Gap")
• Analyze the content of the responses deeply. Look for significant differences in quality, accuracy, style, or reasoning.
• If two responses differ meaningfully (e.g., one is concise and accurate, the other is verbose and vague), but no Existing Rubric captures this difference, you must create a new rubric for it.
• Avoid generic criteria that apply to all responses equally. Focus strictly on what makes them *different*.
2. Radical Non-Redundancy & Sparsity (CRITICAL)
• **Check Coverage First**: Before creating a new rubric, rigorously verify if an Existing Rubric already covers the specific quality or error you noticed.
• **Avoid Duplication**: Never duplicate an existing rubric in meaning or scope. Only target specific nuances or dimensions that are completely absent from the current list.
• **Embrace the Empty List**: If the Existing Rubrics are already comprehensive and fully cover the key aspects of the responses, you MUST return an empty list `[]`. Acknowledging completeness is a sign of expert judgment, not a failure. Do NOT force the creation of mediocre rubrics just to output something.
3. Style Alignment & Formatting

Continued on next page...

Table 11: Prompt template for the Rubric Generator (Continued)

System Prompt

• **Mimic the Style**: Ensure your new criteria match the tone, length, and specificity level of the *Existing Rubrics*.
• **Condition-Based Scoring**: All criteria must be written as conditions that, **when met**, trigger the point assignment.
• **Positive Rubrics (+ Points)**: Describe a specific excellence found in high-quality responses (e.g., "Includes a specific Python code example to illustrate the concept" → If present, +2 points).
• **Negative Rubrics (- Points)**: Describe a specific error or flaw found in low-quality responses (e.g., "Contains hallucinated medical advice" → If present, -5 points).
• **Active Failures Only**: Do NOT write negative rubrics as "Fails to do X". Instead, describe the active failure: "Doing X incorrectly", "Providing dangerous workarounds", or "Exhibiting Y flaw".
4. Actionable & Objective
• Each criterion must describe an observable, verifiable quality or error in the response text. It should be a clear, unambiguous instruction for both human raters and future reward model training.
Analysis Workflow
1. Mental Evaluation (The Coverage Check):
• Read the **Responses** and review the **Existing Rubrics**.
• Mentally apply the existing rubrics to the responses.
• Ask: *Is there a significant strength in a good response or a fatal flaw in a bad response that completely slips through the current criteria?*
2. Gap Identification:
• Identify specific traits (reasoning steps, formatting, safety constraints, tone) that heavily distinguish the responses but are absent from the input rubrics.
3. Direction & Formulation:
• Frame the identified quality as a concrete condition to be met.
4. Prioritization (Quality ¿ Quantity):
• Generate 0 to 10 total rubrics.
• 1 piercingly accurate, highly discriminative rubric is infinitely better than multiple mediocre ones. If no meaningful gap exists, output `[]`.
Output Format
Return ONLY a JSON list of objects. Do not include introductory or concluding remarks outside the JSON block.

```
[
  {
    "criterion":  "<Actionable description of the quality or
error>",
    "points":  <integer, positive for good traits, negative for
bad traits, from -10 to 10>
  },
  ...
]
```

Table 12: Prompt template for the Meta-Verifier tailored for Medical domain.

System Prompt

You are an elite medical grading rubric reviewer. Your task is to evaluate whether a newly proposed grading rubric is valid, practical, and non-redundant compared to existing base rubrics. You must critically evaluate the Target Rubric against the following 6 error categories. If it triggers ANY of these, it is INVALID.

Continued on next page...

Table 12: Prompt template for the Meta-Verifier (Continued)

System Prompt

1. Factual Conflict: The rubric rewards medically dangerous extrapolation, unverified diagnoses, or sycophancy (e.g., rewarding a response for agreeing with a user's false premise). It contradicts standard medical guidelines or encourages hallucinations.
2. Unmeasurable: The rubric demands actions that are impossible for a human rater to verify or clinically impractical for a patient (e.g., "test the pH of the prescribed ointment at home").
3. Semantic Duplication: The rubric functionally overlaps with an existing base rubric, even if it uses different vocabulary or complex medical jargon to mask the duplication.
4. Trivial/Cliché: The rubric focuses on generic, superficial advice that applies to almost any medical query (e.g., "consult a doctor") unless specifically critical to the prompt context.
5. Length Hallucination: The rubric specifically rewards excessively long or verbose answers without adding clinical value.
6. Style Hallucination: The rubric enforces a specific formatting style (e.g., "must use a markdown table") that is irrelevant to the clinical accuracy of the response.
Output your evaluation strictly in the following JSON format. Do not include any other text or markdown blocks outside the JSON:

```
{
    "verdict":  "VALID" // OR "INVALID: <Category Name>"
    // (e.g., "INVALID: Semantic Duplication", "INVALID:
Factual Conflict")
}
```

Table 13: Prompt template for the Meta-Verifier tailored for Writing and Science domains.

System Prompt

You are an elite Meta-Reviewer evaluating grading rubrics for complex, open-ended writing tasks. Your task is to evaluate whether a newly proposed grading rubric is valid, objective, and task-specific compared to existing base rubrics.

You must critically evaluate the Target Rubric against the following 6 error categories. If it triggers ANY of these, it is INVALID.

- **1. Task Constraint Conflict:** The rubric contradicts explicit constraints defined in the original prompt (e.g., word count, specific tone, required elements like quoting experts or mentioning a police officer) OR it hallucinates arbitrary rigid constraints not requested by the user.
- **2. Unmeasurable Subjectivity:** The rubric relies on highly subjective aesthetic judgments (e.g., "beautiful prose", "captivating style") rather than objectively verifiable elements.
- **3. Factuality/Safety Conflict:** For prompts dealing with real-world entities, policies, or science, the rubric rewards misinformation, bias, or unsafe content.
- **4. Trivial/Generic:** The rubric provides generic writing advice (e.g., "check spelling", "use good grammar") that applies to any text, completely failing to address the unique semantic goals of this specific prompt.
- **5. Length Hallucination:** The rubric specifically rewards or punishes response length UNLESS a strict length constraint was explicitly stated in the user's original prompt.
- **6. Semantic Duplication:** The rubric functionally overlaps with an existing base rubric, even if rephrased.

Output your evaluation strictly in the following JSON format. Do not include any other text or markdown blocks outside the JSON:

```
{
"verdict":  "VALID" // OR "INVALID: <Category Name>"
// (e.g., "INVALID: Task Constraint Conflict", "INVALID:
Unmeasurable Subjectivity")
}
```