

Open Rubric System: Scaling Reinforcement Learning with Pairwise Adaptive Rubric*

Wen Wang^{1,2†}, Ruipeng Jia¹, Yunyi Yang¹, Yuxin Wu¹, Yongbo Gai¹, Siyuan Tao^{1,2†}, Mengyu Zhou^{1‡}, Jianhe Lin^{1‡}, Xiaoxi Jiang¹, Guanjun Jiang¹

¹Qwen Business Unit, Alibaba

²University of Chinese Academy of Sciences

Correspondence: zhoumengyu.zmy@alibaba-inc.com, linjianhe.ljh@alibaba-inc.com

Abstract

We argue that robust alignment for non-verifiable tasks is fundamentally a *principle generalization* problem: reward should be guided by explicit, auditable principles rather than compressed into a learned scalar function. To operationalize this view, we present the **Open Rubric System (OpenRS)**, a modular, rubrics-based *LLM-as-a-Judge* framework built around **Pairwise Adaptive Meta-Rubrics (PAMR)** and lightweight *Pointwise Verifiable Rubrics* (PVRs). OpenRS is governed by an explicit *meta-rubric*—not a rubric itself, but a higher-level guideline that specifies *how* rubrics should be generated, which criteria to prioritize, and which failure modes are non-negotiable—and instantiates concrete *adaptive rubrics* on the fly by conditioning on the semantic differences between two candidate responses. It then performs criterion-wise *pairwise* comparisons and aggregates criterion-level preferences externally, avoiding pointwise weighted scalarization while improving discriminability in open-ended settings. Finally, we instantiate OpenRS as reward supervision in pairwise RL training. Empirically, OpenRS achieves state-of-the-art results on four reward-modeling benchmarks and yields stable gains when replacing scalar reward models in end-to-end RL. Our code, refined meta-rubrics, and evaluation scripts are publicly available at <https://github.com/Qwen-Applications/OpenRS>.

1 Introduction

Reinforcement learning from human feedback (RLHF) typically aligns LLMs by training a scalar reward model to compress human preferences into a single score, but such models can become miscalibrated and reward-hackable in practice (Ouyang et al., 2022; Zhong et al., 2025; McKee-Reid et al.,

2024). While reinforcement learning with verifiable rewards (RLVR) can scale capabilities for math and coding by grounding rewards in objective checks (Guo et al., 2025), general open-ended tasks lack explicit reference answers, making reward specification intrinsically difficult. Off-the-shelf LLM-as-a-Judge evaluation (Zheng et al., 2023) and trained generative judges/reward models (Mahan et al., 2024; Ye et al., 2024; Liu et al., 2025c; Anugraha et al., 2025; Whitehouse et al., 2025) offer scalable alternatives, but their evaluation policies remain embedded in model behavior rather than exposed as editable principles.

An increasingly popular line of work constructs rubrics as training-time rewards for open-ended domains (Gunjal et al., 2025; Huang et al., 2025; Liu et al., 2026; Li et al., 2026; Ye et al., 2025; Bi et al., 2025), with recent methods dynamically eliciting criteria online (Rezaei et al., 2025) or refining rubrics from off-policy high-quality responses (Zhang et al., 2025). However, many systems still evaluate each response against rubric items in isolation and map the judgments to a weighted or implicit scalar reward, reintroducing absolute score calibration across prompts, criteria, and judge calls (Gunjal et al., 2025; Huang et al., 2025; Ye et al., 2025). Preference-derived rubric pipelines (Liu et al., 2026; Xie et al., 2025) improve discriminability but can entangle criteria with annotator- or model-specific preference directions, washing out subtle pairwise trade-offs before RL optimization.

We argue that robust alignment for non-verifiable tasks is fundamentally a *principle generalization* problem: reward should be guided by explicit, auditable principles rather than compressed into a learned scalar function. We operationalize this through **Pairwise Adaptive Meta-Rubrics (PAMR)**, which guide a general-purpose LLM judge—*without any judge-specific training*—via an explicit *meta-rubric*: a higher-level guideline governing how rubrics are generated and which

* A preprint version of this paper is available at <https://arxiv.org/abs/2602.14069>.

† Work done during an internship at Alibaba.

‡ Corresponding authors.

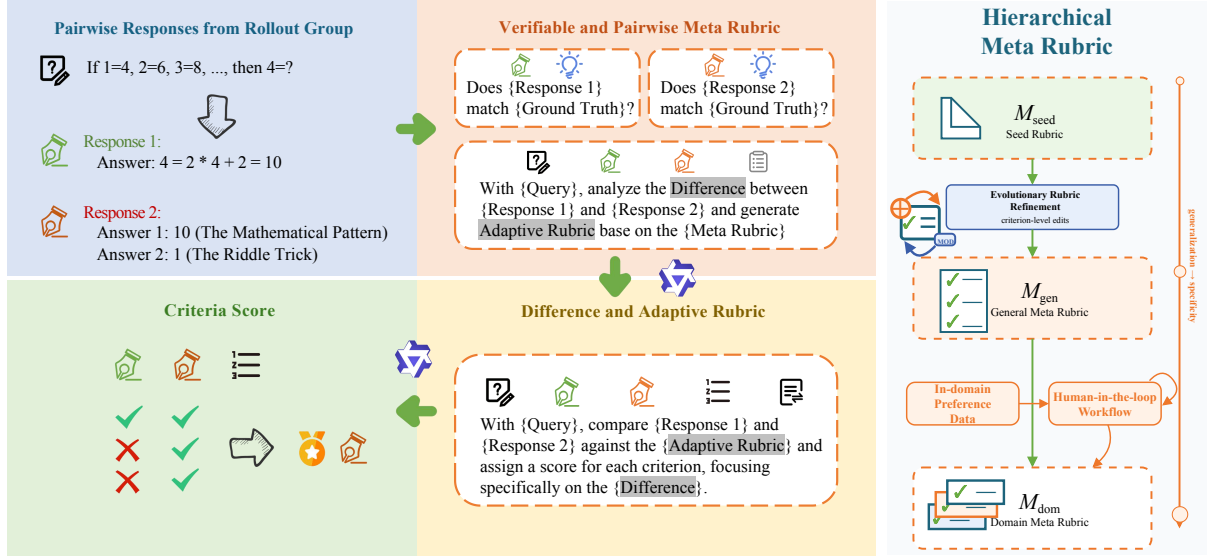


Figure 1: Overview of OpenRS. Left: the OpenRS reward pipeline—given pairwise responses from the rollout group, OpenRS performs verifiable checking, difference-grounded adaptive rubric generation, and criterion-wise pairwise scoring. Right: hierarchical meta-rubric construction from seed rubric $\mathcal{M}_{\text{seed}}$ through evolutionary refinement to $\mathcal{M}_{\text{gen}}$, then domain adaptation via human-in-the-loop to $\mathcal{M}_{\text{dom}}$.

failure modes are non-negotiable. Rather than scoring a single response against an absolute rubric, PAMR conditions on the *observable semantic differences* between two responses to instantiate adaptive rubrics on the fly, while preference labels are used only for downstream refinement or validation rather than per-pair rubric induction. Combined with lightweight pointwise verifiable rubrics and a two-level meta-rubric refinement pipeline, we establish **OpenRS** as a modular bridge between RLVR and RLHF for general tasks.

Our contributions: (1) We reframe reward supervision as a principle generalization problem, leveraging an unmodified LLM judge to instantiate adaptive rubrics from explicit, optimized principles. (2) We open-source OpenRS, which outperforms prior scalar reward models and generative judges across four RM benchmarks. (3) When coupled with pairwise RL training, OpenRS yields consistent gains on public policy benchmarks and in-house evaluations.

2 Related Work

Scalar Reward Models. Standard RLHF pipelines (Ouyang et al., 2022) train a Bradley-Terry scalar reward model that compresses multi-dimensional human preferences into a single score. Recent scalar RMs achieve strong benchmark results through curated/scaled preference data, multi-objective reward design, and attribute-based train-

ing recipes (Liu et al., 2025a; Wang et al., 2024a,b). Nevertheless, scalar-RM interfaces have known failure modes, including reward overoptimization (Gao et al., 2023), poor OOD generalization (Yang et al., 2024b), and information loss from scalar compression (Ye et al., 2024).

LLM-as-a-Judge and Rubric-Based Evaluation.

The LLM-as-a-Judge paradigm (Zheng et al., 2023) is widely adopted but prone to positional and verbosity biases. Trained judges and reward models improve judgment quality through inference-time scaling or RL-trained judging (Liu et al., 2025c; Whitehouse et al., 2025), but their policies are usually embedded in model behavior rather than exposed as stable rubric artifacts. Rubric-based methods (Gunjal et al., 2025; Huang et al., 2025; Xie et al., 2025; Liu et al., 2026; Li et al., 2026; Srivastava et al., 2025) improve transparency by decomposing quality into explicit criteria, but prompt- or instance-specific rubric scoring often reduces criterion judgments to scalar rewards, making them sensitive to absolute-quality priors and cross-prompt calibration. OpenRS instead grounds adaptive criteria in the observable semantic differences between two candidate responses and preserves criterion-level pairwise evidence for external aggregation.

Rubric-Based RL. RLVR (Guo et al., 2025) succeeds where rewards are verifiable; to extend to open-ended domains, recent work uses rubrics as rewards (Gunjal et al., 2025; Zhang et al., 2025;

Rezaei et al., 2025; Ye et al., 2025; Bi et al., 2025; Zhou et al., 2025), but pointwise scalarization limits its discriminability. Jia et al. (2025) introduce pairwise GenRM with BRPO; Zhang et al. (2026) extend relative evaluation to tournament-based ranking, while Xu et al. (2026) jointly optimize rubric generation and pairwise judging via alternating RL. OpenRS builds on this relative-evaluation paradigm through explicit meta-rubric guidance, difference-grounded evaluation, and external criterion-level aggregation (Appendix B).

3 Approach

We introduce the Open Rubric System (OpenRS), a structured rubric system based on the LLM-as-a-Judge paradigm (Figure 1). At the core of OpenRS is the *meta-rubric*: a hierarchical guideline that does not evaluate responses directly, but instead specifies *how* an unmodified LLM judge should generate evaluation criteria, assign weights, and handle failure modes. During evaluation, the meta-rubric is instantiated into concrete, per-pair *Adaptive Rubrics* for subjective assessment, complemented by *Pointwise Verifiable Rubrics* for objective sub-tasks. The meta-rubric itself is optimized offline via evolutionary refinement and human-in-the-loop domain adaptation (right side of Figure 1).

3.1 Overall Framework

Hierarchical Meta Rubric. The foundation of OpenRS is the *Meta Rubric* $\mathcal{M}$ —a rubric-generation guideline (rather than a rubric applied to responses directly) that defines evaluation principles, criterion categories, weighting norms, and non-negotiable failure modes. To balance generalization and specificity, we structure $\mathcal{M}$ hierarchically: a *General Meta Rubric* $\mathcal{M}_{\text{gen}}$ captures broadly applicable alignment principles, while *Domain Meta Rubrics* $\mathcal{M}_{\text{dom}}$ extend it with fine-grained criteria tailored to specific domains (e.g., code generation, creative writing). $\mathcal{M}$ is optimized offline and remains fixed during evaluation—all adaptation happens at the principle level, not the model level.

Pairwise Adaptive Rubric. Given $\mathcal{M}$, OpenRS instantiates a dynamic *Adaptive Rubric* $\mathcal{R}_{ij}$ for each response pair (o_i, o_j) . We first identify the salient semantic differences Δ_{ij} , then condition on them to produce a focused, pair-specific criterion

set:

$$\begin{aligned}\Delta_{ij} &= f_{\text{diff}}(q, o_i, o_j), \\ \mathcal{R}_{ij} &= f_{\text{adapt}}(\mathcal{M}, q, o_i, o_j, \Delta_{ij})\end{aligned}\quad (1)$$

where f_{diff} extracts Δ_{ij} and f_{adapt} turns $\mathcal{M}$ into a weighted criterion set $\mathcal{R}_{ij} = \{(c_k, w_k)\}_{k=1}^K$. For each criterion c_k , the model assigns a comparative score $v_k \in \{-2, -1, 0, 1, 2\}$, where negative values indicate o_i is inferior to o_j , positive values indicate o_i is superior, and 0 implies equality. The criterion-level comparative verdicts are then externally aggregated into a pairwise preference score:

$$s_{ij} = \frac{\sum_{k=1}^K w_k \cdot v_k}{\sum_{k=1}^K w_k} \quad (2)$$

Importantly, f_{diff} , f_{adapt} , and criterion-level scoring are *fused into a single structured generation* rather than issued as separate prompts, and the aggregation in Eq. 2 is a deterministic computation outside the judge. A bidirectional comparison therefore costs exactly two judge calls per pair, independent of the number of criteria K (Appendix D.8).

Pointwise Verifiable Rubric. Certain dimensions are absolute and can be verified on a single output (e.g., word count). We capture these with a *Pointwise Verifiable Rubric* $\mathcal{V}_q$, a set of verifiers derived from the query q . Each verifier $c \in \mathcal{V}_q$ produces a deterministic signal $\phi_c(o)$ (e.g., pass/fail), used as both a guardrail against degenerate behavior and a verifiable reward signal.

Meta-Rubric Refinement. The quality of the reward signal depends critically on $\mathcal{M}$. We refine it via an evolutionary beam-style search (Sec. 3.2) for $\mathcal{M}_{\text{gen}}$ and a human-in-the-loop workflow (Sec. 3.3) for $\mathcal{M}_{\text{dom}}$.

3.2 General Meta Rubric Refinement

We refine the General Meta Rubric $\mathcal{M}_{\text{gen}}$ with a discrete, evolutionary beam-style search over criterion-level edits (Alg. 1). Each candidate rubric is scored by an evaluation oracle $R(\cdot)$: concretely, given a candidate $\mathcal{M}$, we use it to judge all pairs in a held-out preference set ($\sim 2K$ balanced, multi-category examples with human-annotated labels), and $R(\mathcal{M})$ returns the pairwise accuracy—i.e., the fraction of pairs for which the rubric-based judgment agrees with the human preference. To make search consistently improve (rather than sampling mostly degrading mutations), we train the refinement policy (Qwen3-30B-A3B) with an Asym-GRPO-style update that keeps gradients only for

Algorithm 1: General Meta Rubric Refinement

Input: Seed rubric $\mathcal{M}_{\text{seed}}$; beam size B ; total rollouts per iteration G ; iterations T ; oracle $R(\cdot)$; refinement policy π_θ ; reference policy π_{ref} ; GRPO hyperparams ε, β

Output: Refined general meta rubric $\mathcal{M}_{\text{gen}}$ and trained refinement policy π_θ

```
1 Initialize beam  $\mathcal{B}_0 \leftarrow \{\mathcal{M}_{\text{seed}}\}$  and fill to size  $B$  by copying;
2 for  $t \leftarrow 0$  to  $T - 1$  do
3   Initialize candidate set  $\mathcal{C} \leftarrow \emptyset$ ;
4    $g_{\text{per}} \leftarrow G/B$ ;
5   foreach  $\mathcal{M} \in \mathcal{B}_t$  do
6     for  $j \leftarrow 1$  to  $g_{\text{per}}$  do
7       Roll out an edit sequence  $o \sim \pi_{\theta_{\text{old}}}(\cdot | \mathcal{M})$  with actions in
7        $\mathcal{A} = \{\text{ADD}, \text{DELETE}, \text{MODIFY}\}$ ;
8       Apply edits to obtain mutated rubric  $\tilde{\mathcal{M}} \leftarrow \text{APPLYEDITS}(\mathcal{M}, o)$ ;
9       Score reward  $R \leftarrow R(\tilde{\mathcal{M}})$ ;
10      Add  $(\tilde{\mathcal{M}}, o, R)$  to  $\mathcal{C}$ ;
11  Update beam  $\mathcal{B}_{t+1} \leftarrow \text{Top}_B(\{\tilde{\mathcal{M}} : (\tilde{\mathcal{M}}, o, R) \in \mathcal{C}\}, \text{by } R)$ ;
12  Compute GRPO advantages  $\{\hat{A}_{i,\tau}\}$  from rewards  $\{R_i\}_{i=1}^G$  as in Eq. 5;
13  Let  $\mathcal{I} \leftarrow \text{Top}_B(\{R_i\}_{i=1}^G)$  be indices of top- $B$  rollouts in the group;
14  Update  $\pi_\theta$  by maximizing the Asym-GRPO objective over  $i \in \mathcal{I}$ :
```

$$\frac{1}{B} \sum_{i \in \mathcal{I}} \frac{1}{|o_i|} \sum_{\tau=1}^{|o_i|} \left(\min(r_{i,\tau}(\theta) \hat{A}_{i,\tau}, \text{clip}(r_{i,\tau}(\theta), 1 - \varepsilon, 1 + \varepsilon) \hat{A}_{i,\tau}) - \beta D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}) \right)$$

$$\text{where } r_{i,\tau}(\theta) = \frac{\pi_\theta(o_{i,\tau} | \mathcal{M}_i, o_{i,<\tau})}{\pi_{\theta_{\text{old}}}(o_{i,\tau} | \mathcal{M}_i, o_{i,<\tau})};$$

```
15 Return  $\mathcal{M}_{\text{gen}} \leftarrow \arg \max_{\mathcal{M} \in \mathcal{B}_T} R(\mathcal{M})$  and  $\pi_\theta$ ;
```

top-performing rollouts. As shown in Table 3, replacing the refined $\mathcal{M}_{\text{gen}}$ with the initial $\mathcal{M}_{\text{seed}}$ (while keeping the evaluation prompt unchanged) causes a substantial performance drop (85.4 vs. 90.7 on RewardBench v2), confirming that evolutionary refinement discovers stronger evaluation principles. Full details of the search procedure and optimization are provided in Appendix C.

3.3 Domain Meta Rubric Refinement

While the General Meta Rubric captures broadly applicable alignment principles, evaluation settings with heterogeneous subjective preferences (e.g., the PPE Chinese subset spanning 40+ user-facing categories) benefit from domain-specific adaptation (Appendix D.9). We construct the Domain Meta Rubric $\mathcal{M}_{\text{dom}}$ by adapting the refined $\mathcal{M}_{\text{gen}}$ using a small, curated set of in-domain preference data. These data are used for aggregate error analysis and domain-level principle edits, not as input to per-pair adaptive-rubric instantiation. Unlike the black-box exploration used for $\mathcal{M}_{\text{gen}}$, domain

adaptation is carried out in a data-aware manner via error analysis rather than gradient-based reinforcement learning. To mitigate overfitting, we explicitly constrain updates to introduce abstract, reusable principles (instead of case-specific heuristics) and validate the adapted rubric on held-out domain splits to ensure it remains generalizable and free of dataset artifacts. Importantly, this is a human-in-the-loop workflow: human experts periodically audit proposed edits, resolve ambiguous failure cases, and approve revisions before merging them into $\mathcal{M}_{\text{dom}}$. This design is also motivated by cost: training the general refinement policy already requires substantial compute (see Table 5 in Appendix); replicating this for each domain would multiply the cost linearly with the number of domains. This workflow avoids extra overhead and enables rapid specialization.

Model	RewardBench v2	PPE Pref / ZH	RM-Bench	JudgeBench	Avg.
<i>LLM-as-a-Judge & Generative Reward Models & Rubric</i>					
DeepSeek-GRM-27B (Liu et al., 2025c)	-	71.1*	-	-	-
DeepSeek-GRM-27B (w/ MetaRM) (Liu et al., 2025c)	-	72.2*	-	-	-
RM-R1-Qwen-Instruct-32B (Chen et al., 2025)	-	76.3*	79.1	-	-
RM-R1-DeepSeek-Distill-Qwen-32B (Chen et al., 2025)	-	75.3*	83.9	-	-
EvalPlanner (Llama-3.3-70B) (Saha et al., 2025)	-	-	82.1	56.6	-
J1-Llama-70B (Whitehouse et al., 2025)	-	-	82.7	60.0	-
RationaleRM (Qwen3-30B-A3B) (Wang et al., 2026)	-	-	87.1	82.0	-
Auto-Rubric (UltraFeedback, Qwen3-235B-A22B-Instruct-2507) (Xie et al., 2025)	-	78.0*	-	74.3*	-
<i>Open Scalar Reward Models</i>					
Llama-3-OffsetBias-RM-8B (Park et al., 2024)	64.8	60.6*	71.3	63.5	65.0
ArmoRM-Llama3-8B-v0.1 (Wang et al., 2024a)	66.5	58.3*	69.3	59.7	63.4
Internlm2-20b-reward (Cai et al., 2024)	56.3	69.4*	68.3	64.3	64.8
LDL-Reward-Gemma-2-27B-v0.1	72.5	43.3*	71.1	64.2	67.4
Llama-3.1-Nemotron-70B (Wang et al., 2025a)	76.7	68.7*	72.2	65.8	70.5
INF-ORM-Llama3.1-70B (Yang et al., 2024a)	76.5	65.7*	73.8	70.2	71.7
Skywork-Reward-Llama-3.1-8B-v0.2 (Liu et al., 2024)	71.8	62.2*	72.1	62.9	67.1
Skywork-Reward-Gemma-2-27B-v0.2 (Liu et al., 2024)	75.3	57.6*	70.0	66.5	67.3
Skywork-Reward-V2-Qwen3-8B (Liu et al., 2025a)	78.2	76.4*	82.6	73.4	77.6
Skywork-Reward-V2-Llama-3.1-8B (Liu et al., 2025a)	84.1	79.6*	92.8	80.0	84.3
<i>LLM-as-a-Judge & Rubric System</i>					
OpenRS (Qwen3-8B)	75.1	73.5	92.4	89.5	82.6
OpenRS (gpt-oss-120b)	85.7	76.6	91.7	93.3	86.8
OpenRS (DeepSeek-V3.1)	85.6	78.1	91.8	89.1	86.2
OpenRS (DeepSeek-V3.2)	85.8	78.9	92.7	89.4	86.7
OpenRS (Qwen3-30B-A3B-Instruct-2507)	83.7	78.4	91.1	86.4	84.9
OpenRS (Qwen3-235B-A22B-Instruct-2507)	90.7	82.3	93.0	91.6	89.4

Table 1: Main benchmark comparison of OpenRS with scalar reward models (SRMs), generative judges, and rubric-based methods. For OpenRS, parentheses denote the unmodified LLM judge backbone. **Bold**: best overall. “*”: re-implemented. “-”: unavailable.

3.4 Applying OpenRS to Reinforcement Learning

We use the Open Rubric System (OpenRS) as a scalable reward interface for RL on open-ended generation by converting evaluation under the offline-refined hierarchical Meta Rubric into training feedback. To make pairwise rubric evaluation efficient, we adopt Bootstrapped Relative Policy Optimization (BRPO; Appendix A.2) and score each rollout o_i with a sum of a pairwise adaptive term and an optional verifiable term:

$$R(q, o_i) = s_{i,\text{ref}} + \gamma \sum_{c \in \mathcal{V}_q} \phi_c(o_i), \quad (3)$$

where $s_{i,\text{ref}}$ is the criterion-weighted preference score from the Adaptive Rubric $\mathcal{R}_{i,\text{ref}}$, and $\sum_{c \in \mathcal{V}_q} \phi_c(o_i)$ aggregates pointwise verifiers from $\mathcal{V}_q$ (e.g., pass/fail constraints), scaled by γ .

4 Experiments

4.1 Experimental Setup

We evaluate OpenRS against state-of-the-art SRMs on four diverse benchmarks: **RM-Bench** (Liu et al., 2025b), **JudgeBench** (Tan et al., 2025), **Reward-Bench v2** (Malik et al., 2025), and **PPE Preference** (Frick et al., 2025). We use the refined meta-rubric from Sec. 3.2; its oracle preference

set is disjoint from all four evaluation benchmarks, preventing benchmark leakage. PPE Preference is multilingual ($\sim 16k$ examples) and includes a Chinese subset ($\sim 1.2k$); we report results on the Chinese subset and mark obvious label errors as `label_error` after cross-checking with multiple strong LLM judges and human review. To mitigate position bias, all pairwise evaluations are conducted bidirectionally: we evaluate both (o_i, o_j) and (o_j, o_i) and accept a preference only when both directions agree; contradictory judgments are labeled “Same” (details in Appendix D). For reproducibility, we release our refined Meta Rubric, Verifiable Rubric, the complete evaluation scripts, and all preference data at <https://github.com/Qwen-Applications/OpenRS>.

4.2 Main Results

As shown in Table 1, OpenRS achieves state-of-the-art alignment with human preferences across all four benchmarks and attains the best overall average. Notably, OpenRS (Qwen3-235B-A22B) outperforms the strongest open SRM baseline (Skywork-Reward-V2) by +5.1 on the average score (89.4 vs. 84.3), with consistent gains across all four benchmarks. OpenRS also outperforms all LLM-based judge baselines that require domain-specific training (top of Table 1): RM-R1-

Model	IFEval	IFScale	EQ-Bench	JudgeMark-v2	Chinese SQA	Avg.
RL + Scalar RM	82.3	54.8	82.2	43.8	78.9	68.4
RL + Pointwise OpenRS	82.7	56.2	81.6	49.4	77.2	69.4
RL + OpenRS	83.0	59.0	81.2	54.4	79.1	71.3

Table 2: End-to-end RL comparison on the same in-house 235B-scale policy, replacing only the reward signal: scalar RM, pointwise OpenRS, or pairwise OpenRS.

Model	RBv2	PPE/ZH
Pointwise OpenRS	86.1	76.2
Two-stage (query-only)	86.2	75.5
Seed rubric (same prompt)	85.4	74.6
OpenRS	90.7	82.3
- w/o diff mechanism	88.9	80.6
- w/o domain meta rubric	-	80.9

Table 3: Ablation study of OpenRS.

32B (Chen et al., 2025) (76.3* on PPE), J1-70B (Whitehouse et al., 2025) (60.0 on JudgeBench), and Auto-Rubric (Xie et al., 2025) (78.0*/74.3* on PPE/JudgeBench, re-evaluated with the same 235B backbone using pointwise rubric). The judge model used for benchmark evaluation is entirely unmodified—only the meta-rubric (a text artifact) is optimized via a separate refinement policy—confirming that the structured pipeline, not specialized judge training, is the key differentiator.

Different Foundation Models for OpenRS. We design OpenRS to run locally using open-weights judge backbones; non-Qwen models (gpt-oss-120b, DeepSeek) also surpass SRM baselines, supporting cross-family generality. Even with an unmodified Qwen3-8B as the judge, OpenRS achieves 92.4 on RM-Bench and 89.5 on JudgeBench—matching the 235B configuration and far exceeding the best SRM (92.8/82.0). The RewardBench v2 gap (75.1 vs. 90.7) traces mainly to the Precise IF subset (26.9%; see Table 6), where instruction-following verification demands capabilities that scale with model size; on Math (92.9%), Safety (88.5%), and Ties (83.9%) the 8B backbone remains competitive, confirming that the pipeline is effective across scales and gaps reflect backbone limitations, not pipeline deficiencies. Against Skywork-V2-Qwen3-8B (same backbone, 26M pairs), OpenRS (Qwen3-8B) averages 82.6 vs. 77.6 (+5.0) across all four benchmarks, with +9.8 on RM-Bench and +16.1 on JudgeBench. With Qwen3-30B-A3B (3B active), OpenRS reaches 84.9, exceeding Skywork-V2-Llama-3.1-8B (84.3).

Pareto Frontier and Inference Cost. Fig. 2 in Appendix summarizes the FLOPs–accuracy trade-

off; we choose Qwen3-235B-A22B as the default RL judge for its best quality–compute balance. In absolute terms, OpenRS remains deployment-feasible: on our shared serving stack a rubric-based comparison amortizes to ~ 2.9 s and ~ 23 GPU-seconds per response end-to-end, and judge calls overlap asynchronously with policy rollouts during RL (Appendix D.8).

4.3 End-to-End RL with Open Rubric System

We evaluate OpenRS as a drop-in SRM replacement under the controlled end-to-end RL setting detailed in Appendix D.2; Table 2 reports downstream performance. This single substitution improves average downstream performance, with notable gains on IFScale (+4.2) and JudgeMark-v2 (+10.6). A pointwise rubric variant improves modestly over the scalar RM but remains behind full pairwise OpenRS, supporting the importance of pairwise, difference-grounded evaluation. Qualitatively, the OpenRS-trained model exhibits richer subjective expression and clearer stance-taking compared to the “assistant-like” neutrality of scalar-RM baselines, suggesting that preserving multi-dimensional preference structure leads to better alignment in open-ended generation (Appendix F, Figs. 9–10). Training-time diagnostics and throughput details are also provided in Appendix D.2.

4.4 Ablation Study

Isolating the Rubric Workflow. A natural question is whether OpenRS gains come from the rubric machinery itself or merely from adopting a pairwise, bidirectional evaluation protocol. We therefore run a strict same-backbone control: the identical unmodified Qwen3-235B-A22B judge, the identical bidirectional protocol, and the identical $\{A, B, \text{Same}\}$ output space, but with the meta-rubric, difference extraction, and criterion-level decomposition all removed, so the judge emits a single holistic verdict per pair. As shown in Table 4, the rubric workflow adds +6.8 average accuracy over this control, with the largest gain on RewardBench v2 (+10.4). The indecision statistics point to the same conclusion: the direct judge returns “Same” or self-contradictory bidirectional verdicts on 23.6%/16.4%/20.2% of RewardBench v2/RM-Bench/JudgeBench pairs, whereas OpenRS reduces this to 10.6%/7.4%/6.3% (Appendix D.6). Decomposing a comparison into explicit, weighted criteria therefore converts otherwise indecisive holistic judgments into resolvable criterion-level

Method	RBv2	RM-B	JudgeB	Avg.
Direct pairwise judge	80.3	90.3	84.3	85.0
OpenRS	90.7	93.0	91.6	91.8
Δ	+10.4	+2.7	+7.3	+6.8

Table 4: Isolating the rubric workflow. Both rows use the *same* unmodified Qwen3-235B-A22B judge and the *same* bidirectional protocol; the control simply drops the meta-rubric, difference extraction, and criterion decomposition. RM-B: RM-Bench; JudgeB: JudgeBench.

evidence—the gains are attributable to the rubric workflow, not to the pairwise protocol alone.

Component Ablations. We further ablate individual design choices in Table 3. Replacing the refined $\mathcal{M}_{\text{gen}}$ with the initial seed rubric (same prompt) causes a large drop (85.4 vs. 90.7 on RewardBench v2), confirming that evolutionary refinement produces stronger principles. Pointwise rubric scoring underperforms pairwise comparison (86.1 vs. 90.7). The *Two-stage* baseline (query-only rubric generation) drops to 86.2/75.5; even skipping explicit Δ_{ij} extraction (*w/o diff*) degrades to 88.9/80.6, confirming that structured difference analysis is essential. Domain Meta Rubrics help on PPE (ZH) with heterogeneous preferences (82.3 vs. 80.9; Appendix D.9).

4.5 Additional Analysis

Additional experiments in Appendix D include: refinement-policy ablations showing Asym-GRPO stabilizes discrete search (Fig. 3); fine-grained breakdowns by RM-Bench difficulty and RewardBench v2 category (Table 6); LMArena-style generalization on in-house QA/Writing sets (Table 7); and an “Aha Moment” entropy analysis showing distinctive non-monotonic patterns in rubric-based RL (Fig. 4a; Appendix D.5). We release judge prompts, evaluation scripts, and case studies for reproducibility at <https://github.com/Qwen-Applications/OpenRS> (Appendix F).

5 Conclusion

We argued that robust reward supervision for non-verifiable tasks is a *principle generalization* problem and introduced **OpenRS**, a modular rubric-based reward interface combining **Pairwise Adaptive Meta-Rubrics** with lightweight *Pointwise Verifiable Rubrics*. Empirically, OpenRS achieves state-of-the-art results across four reward-modeling benchmarks and yields consistent gains when used as a drop-in replacement for scalar reward mod-

els in end-to-end RL, while remaining throughput-feasible for large-scale training. Key future directions include judge distillation for reduced cost, adversarial robustness evaluation, automated domain adaptation, and larger-scale RL studies to validate rubric-based emergence in general domains.

Limitations

The general meta-rubric refinement requires training a refinement policy (Table 5), which makes it prohibitively expensive to repeat for every domain. We therefore resort to a human-in-the-loop workflow for domain meta-rubric adaptation, which introduces non-trivial expert labor and creates an inconsistency in generation paradigm between $\mathcal{M}_{\text{gen}}$ (automated evolutionary search) and $\mathcal{M}_{\text{dom}}$ (manual curation). Although our ablations (Table 3) confirm that each refinement level contributes meaningful gains, a future direction is to extend the automated refinement paradigm to domain adaptation, reducing human cost while maintaining the unified optimization framework. Second, although we design the rubric generation to be grounded in semantic differences rather than preference direction, the underlying LLM may still form implicit preferences when both responses are visible; while our difference-first design and external aggregation mitigate this risk (Section B.2), it cannot be fully eliminated. Third, our end-to-end RL comparison isolates the reward signal under controlled conditions with a single policy model; broader deployment validation across diverse policy sizes, languages, and production environments remains future work.

References

- Afra Feyza Akyürek, Advait Gosai, Chen Bo Calvin Zhang, Vipul Gupta, Jaehwan Jeong, Anisha Gunjal, Tahseen Rabbani, Maria Mazzone, David Randolph IV, Mohammad Mahmoudi Meymand, Gurshaan Chattha, Paula Rodriguez, Diego A. Mares Buendia, Pavit Singh, Michael Liu, Subodh Chawla, Peter Cline, Lucy Ogaz, Ernesto Gabriel Hernández Montoya, and 5 others. 2026. **PRBench: Large-scale expert rubrics for evaluating high-stakes professional reasoning**. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 42297–42325, San Diego, California, United States. Association for Computational Linguistics.
- David Anugraha, Zilu Tang, Lester James V. Miranda, Hanyang Zhao, Mohammad Rifqi Farhansyah, Garry Kuwanto, Derry Wijaya, and Genta Indra Winata.

2025. [R3: Robust rubric-agnostic reward models](#). *Preprint*, arXiv:2505.13388.
- Rahul K. Arora, Jason Wei, Rebecca Soskin Hicks, Preston Bowman, Joaquin Quiñero-Candela, Foivos Tsimpourlas, Michael Sharman, Meghan Shah, Andrea Vallone, Alex Beutel, Johannes Heidecke, and Karan Singhal. 2025. [Healthbench: Evaluating large language models towards improved human health](#). *Preprint*, arXiv:2505.08775.
- Baolong Bi, Shenghua Liu, Yiwei Wang, Siqian Tong, Lingrui Mei, Yuyao Ge, Yilong Xu, Jiafeng Guo, and Xueqi Cheng. 2025. [Reward and guidance through rubrics: Promoting exploration to improve multi-domain reasoning](#). *Preprint*, arXiv:2511.12344.
- Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, and 1 others. 2024. Internlm2 technical report. *arXiv preprint arXiv:2403.17297*.
- Xiuxi Chen, Gaotang Li, Ziqi Wang, Bowen Jin, Cheng Qian, Yu Wang, Hongru Wang, Yu Zhang, Denghui Zhang, Tong Zhang, and 1 others. 2025. [Rm-r1: Reward modeling as reasoning](#). *arXiv preprint arXiv:2505.02387*.
- Evan Frick, Tianle Li, Connor Chen, Wei-Lin Chiang, Anastasios N. Angelopoulos, Jiantao Jiao, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. 2025. [How to evaluate reward models for RLHF](#). In *The Thirteenth International Conference on Learning Representations*.
- Leo Gao, John Schulman, and Jacob Hilton. 2023. [Scaling laws for reward model overoptimization](#). In *International Conference on Machine Learning (ICML)*.
- Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. 2025. [Rubrics as rewards: Reinforcement learning beyond verifiable domains](#). *Preprint*, arXiv:2507.17746.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shiron Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. [Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning](#). *arXiv preprint arXiv:2501.12948*.
- 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, and 2 others. 2025. [Reinforcement learning with rubric anchors](#). *Preprint*, arXiv:2508.12790.
- Ruipeng Jia, Yunyi Yang, Yongbo Gai, Kai Luo, Shihao Huang, Jianhe Lin, Xiaoxi Jiang, and Guanjuan Jiang. 2025. [Writing-zero: Bridge the gap between non-verifiable tasks and verifiable rewards](#). *Preprint*, arXiv:2506.00103.
- Sunzhu Li, Jiale Zhao, Huimin Ren, Zhenlin Wei, Yang Zhou, Jingwen Yang, Shunyu Liu, Kaike Zhang, and Chen Wei. 2026. [RubricHub: A comprehensive and highly discriminative rubric dataset via automated coarse-to-fine generation](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 31320–31344, San Diego, California, United States. Association for Computational Linguistics.
- Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let’s verify step by step. In *The Twelfth International Conference on Learning Representations*.
- Chris Yuhao Liu, Liang Zeng, Jiakai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. 2024. [Skywork-reward: Bag of tricks for reward modeling in llms](#). *arXiv preprint arXiv:2410.18451*.
- Chris Yuhao Liu, Liang Zeng, Yuzhen Xiao, Jujie He, Jiakai Liu, Chaojie Wang, Rui Yan, Wei Shen, Fuxiang Zhang, Jiacheng Xu, Yang Liu, and Yahui Zhou. 2025a. [Skywork-reward-v2: Scaling preference data curation via human-ai synergy](#). *Preprint*, arXiv:2507.01352.
- Tianci Liu, Ran Xu, Tony Yu, Ilgee Hong, Carl Yang, Tuo Zhao, and Haoyu Wang. 2026. [Openrubrics: Towards scalable synthetic rubric generation for reward modeling and LLM alignment](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 17417–17437, San Diego, California, United States. Association for Computational Linguistics.
- Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, Lei Hou, and Juanzi Li. 2025b. [RM-Bench: Benchmarking reward models of language models with subtlety and style](#). In *The Thirteenth International Conference on Learning Representations*.
- Zijun Liu, Peiyi Wang, Runxin Xu, Shirong Ma, Chong Ruan, Peng Li, Yang Liu, and Yu Wu. 2025c. [Inference-time scaling for generalist reward modeling](#). *arXiv preprint arXiv:2504.02495*.
- Dakota Mahan, Duy Van Phung, Rafael Rafailov, Chase Blagden, Nathan Lile, Louis Castricato, Jan-Philipp Fränken, Chelsea Finn, and Alon Albalak. 2024. [Generative reward models](#). *arXiv preprint arXiv:2410.12832*.
- Saumya Malik, Valentina Pyatkin, Sander Land, Jacob Morrison, Noah A. Smith, Hannaneh Hajishirzi, and Nathan Lambert. 2025. [Rewardbench 2: Advancing reward model evaluation](#). *Preprint*, arXiv:2506.01937.
- Leo McKee-Reid, Christoph Sträter, Maria Angelica Martinez, Joe Needham, and Mikita Balesni. 2024. [Honesty to subterfuge: In-context reinforcement learning can make honest models reward hack](#). *arXiv preprint arXiv:2410.06491*.

- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.
- Junsoo Park, Seungyeon Jwa, Meiyang Ren, Daeyoung Kim, and Sanghyuk Choi. 2024. [OffsetBias: Leveraging debiased data for tuning evaluators](#). In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1043–1067, Miami, Florida, USA. Association for Computational Linguistics.
- MohammadHossein Rezaei, Robert Vacareanu, Zihao Wang, Clinton Wang, Bing Liu, Yunzhong He, and Afra Feyza Akyürek. 2025. [Online rubrics elicitation from pairwise comparisons](#). *Preprint*, arXiv:2510.07284.
- Swarnadeep Saha, Xian Li, Marjan Ghazvininejad, Jason Weston, and Tianlu Wang. 2025. [Learning to plan & reason for evaluation with thinking-LLM-as-a-judge](#). In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 52565–52583. PMLR.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, and 1 others. 2024. Deepseek-math: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.
- Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R. Bowman, Newton Cheng, Esin Durmus, Zac Hatfield-Dodds, Scott R. Johnston, Shauna Kravec, Timothy Maxwell, Sam McCandlish, Kamal Ndousse, Oliver Rausch, Nicholas Schiefer, Da Yan, Miranda Zhang, and Ethan Perez. 2024. [Towards understanding syco-phancy in language models](#). In *The Twelfth International Conference on Learning Representations*.
- Pragya Srivastava, Harman Singh, Rahul Madhavan, Gandharv Patil, Sravanti Addepalli, Arun Sugala, Rengarajan Aravamudhan, Soumya Sharma, Anirban Laha, Aravindan Raghuvier, Karthikeyan Shanmugam, and Doina Precup. 2025. [Robust reward modeling via causal rubrics](#). *Preprint*, arXiv:2506.16507.
- Sijun Tan, Siyuan Zhuang, Kyle Montgomery, William Y. Tang, Alejandro Cuadron, Chenguang Wang, Raluca Ada Popa, and Ion Stoica. 2025. [JudgeBench: A benchmark for evaluating LLM-based judges](#). In *The Thirteenth International Conference on Learning Representations*.
- Vijay Viswanathan, Yanchao Sun, Shuang Ma, Xiang Kong, Meng Cao, Graham Neubig, and Tongshuang Wu. 2025. [Checklists are better than reward models for aligning language models](#). In *Advances in Neural Information Processing Systems*, volume 38, pages 127195–127221.
- Binghai Wang, Yantao Liu, Yuxuan Liu, Tianyi Tang, Shenzhi Wang, Chang Gao, Chujie Zheng, Yichang Zhang, Le Yu, Shixuan Liu, Tao Gui, Qi Zhang, Xu-anjing Huang, Bowen Yu, Fei Huang, and Junyang Lin. 2026. [Outcome accuracy is not enough: Aligning the reasoning process of reward models](#). *Preprint*, arXiv:2602.04649.
- Haoliang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. 2024a. [Interpretable preferences via multi-objective reward modeling and mixture-of-experts](#). In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10582–10592, Miami, Florida, USA. Association for Computational Linguistics.
- Zhilin Wang, Alexander Bukharin, Olivier Delalleau, Daniel Egert, Gerald Shen, Jiaqi Zeng, Oleksii Kuchaiev, and Yi Dong. 2025a. [HelpSteer2-Preference: Complementing ratings with preferences](#). In *The Thirteenth International Conference on Learning Representations*.
- Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J. Zhang, Makesh Narsimhan Sreedhar, and Oleksii Kuchaiev. 2024b. [HelpSteer2: Open-source dataset for training top-performing reward models](#). In *Advances in Neural Information Processing Systems*, volume 37, pages 16570–16595.
- Zhilin Wang, Jaehun Jung, Ximing Lu, Shizhe Diao, Ellie Evans, Jiaqi Zeng, Pavlo Molchanov, Yejin Choi, Jan Kautz, and Yi Dong. 2025b. [Profbench: Multi-domain rubrics requiring professional knowledge to answer and judge](#). *Preprint*, arXiv:2510.18941.
- Chenxi Whitehouse, Tianlu Wang, Ping Yu, Xian Li, Jason Weston, Ilia Kulikov, and Swarnadeep Saha. 2025. J1: Incentivizing thinking in llm-as-a-judge via reinforcement learning. *arXiv preprint arXiv:2505.10320*.
- Lipeng Xie, Sen Huang, Zhuo Zhang, Anni Zou, Yunpeng Zhai, Dingchao Ren, Kezun Zhang, Haoyuan Hu, Boyin Liu, Haoran Chen, Zhao Yang Liu, and Bolin Ding. 2025. [Auto-rubric: Learning from implicit weights to explicit rubrics for reward modeling](#). *Preprint*, arXiv:2510.17314.
- Ran Xu, Tianci Liu, Zihan Dong, Tony Yu, Ilgee Hong, Carl Yang, Linjun Zhang, Tao Zhao, and Haoyu Wang. 2026. Alternating reinforcement learning for rubric-based reward modeling in non-verifiable llm post-training. *arXiv preprint arXiv:2602.01511*.
- Minghao Yang, Chao Qu, and Xiaoyu Tan. 2024a. [Inform-llama3.1-70b](#).

Rui Yang, Ruomeng Ding, Yong Lin, Huan Zhang, and Tong Zhang. 2024b. [Regularizing hidden states enables learning generalizable reward model for LLMs](#). In *Advances in Neural Information Processing Systems*, volume 37, pages 62279–62309.

Zhiling Ye, Yun Yue, Haowen Wang, Xudong Han, Jiadi Jiang, Cheng Wei, Lei Fan, Jiaxin Liang, Shuowen Zhang, Ji Li, Chunxiao Guo, Jian Wang, Peng Wei, and Jinjie Gu. 2025. [Self-rewarding rubric-based reinforcement learning for open-ended reasoning](#). *Preprint*, arXiv:2509.25534.

Ziyi Ye, Xiangsheng Li, Qiuchi Li, Qingyao Ai, Yujia Zhou, Wei Shen, Dong Yan, and Yiqun Liu. 2024. Beyond scalar reward model: Learning generative judge from preference data. *arXiv preprint arXiv:2410.03742*.

Qiyang Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gao-hong Liu, Juncai Liu, and 1 others. 2025a. [DAPO: An open-source LLM reinforcement learning system at scale](#). In *Advances in Neural Information Processing Systems*, volume 38, pages 40279–40325.

Zhuohao Yu, Jiali Zeng, Weizheng Gu, Yidong Wang, Jindong Wang, Fandong Meng, Jie Zhou, Yue Zhang, Shikun Zhang, and Wei Ye. 2025b. [Rewardanything: Generalizable principle-following reward models](#). *Preprint*, arXiv:2506.03637.

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

Qiang Zhang, Boli Chen, Fanrui Zhang, Ruixue Ding, Shihang Wang, Qiuchen Wang, Yinfeng Huang, Haonan Zhang, Rongxiang Zhu, Pengyong Wang, Ailin Ren, Xin Li, Pengjun Xie, Jiawei Liu, Ning Guo, Jingren Zhou, and Zheng-Jun Zha. 2026. [Arenarl: Scaling rl for open-ended agents via tournament-based relative ranking](#). *Preprint*, arXiv:2601.06487.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhaghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. [Judging LLM-as-a-judge with MT-Bench and chatbot arena](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623.

Jialun Zhong, Wei Shen, Yanzeng Li, Songyang Gao, Hua Lu, Yicheng Chen, Yang Zhang, Wei Zhou, Jinjie Gu, and Lei Zou. 2025. A comprehensive survey of reward models: Taxonomy, applications, challenges, and future. *arXiv preprint arXiv:2504.12328*.

Yang Zhou, Sunzhu Li, Shunyu Liu, Wenkai Fang, Kongcheng Zhang, Jiale Zhao, Jingwen Yang, Yihe Zhou, Jianwei Lv, Tongya Zheng, Hengtong Lu, Wei Chen, Yan Xie, and Mingli Song. 2025. [Breaking the](#)

[exploration bottleneck: Rubric-scaffolded reinforcement learning for general llm reasoning](#). *Preprint*, arXiv:2508.16949.

A Preliminary

A.1 Group Relative Policy Optimization

GRPO (Shao et al., 2024) is an efficient reinforcement learning algorithm that eliminates the need for a value function critic, as used in PPO (Schulman et al., 2017), by introducing group-relative normalization for advantage estimation. Given a query q from the dataset $\mathcal{D}$, the behavior policy $\pi_{\theta_{\text{old}}}$ samples a group of G outputs $\{o_i\}_{i=1}^G$. The advantage of the i -th response is subsequently derived by normalizing the rewards $\{R_i\}_{i=1}^G$ relative to the group statistics. Similar to PPO, GRPO ensures training stability and sample efficiency by constraining policy updates π_{θ} within a trust region using a clipping mechanism. Formally, GRPO optimizes the following objective:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{q \sim \mathcal{D}, \{o_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}} \left[\frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \left(\min(r_{i,t}(\theta) \hat{A}_{i,t}, \text{clip}(r_{i,t}(\theta), 1-\varepsilon, 1+\varepsilon) \hat{A}_{i,t}) - \beta D_{\text{KL}}(\pi_{\theta} \parallel \pi_{\text{ref}}) \right) \right] \quad (4)$$

$$r_{i,t}(\theta) = \frac{\pi_{\theta}(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})} \quad (5)$$

$$\hat{A}_{i,t} = \frac{R_i - \text{mean}(\{R_i\}_{i=1}^G)}{\text{std}(\{R_i\}_{i=1}^G)} \quad (6)$$

where $\hat{A}_{i,t}$ denotes the estimated advantage at time step t , ε is the clipping hyperparameter for the importance sampling ratio $r_{i,t}(\theta)$, and π_{ref} is the reference policy (typically initialized as π_{θ}).

A.2 Pairwise Evaluation and Bootstrapped Relative Policy Optimization

Jia et al. (2025) advocate for pairwise evaluation over pointwise scoring, particularly for non-verifiable tasks where absolute ground truth is elusive. However, integrating pairwise feedback into group-based RL algorithms like GRPO presents a significant computational challenge. To estimate the advantage of each response within a group of size G , a naive approach would require comparing

every response against every other response, resulting in $\binom{G}{2}$ evaluations and a computational complexity of $O(G^2)$. Writing-Zero introduces Bootstrapped Relative Policy Optimization (BRPO). Instead of exhaustive pairwise comparisons, BRPO employs a bootstrapping mechanism where a single response o_{ref} is randomly sampled from the current group $\{o_i\}_{i=1}^G$ to serve as a dynamic anchor. The quality of every other response o_i is then evaluated relative to this anchor. This strategy reduces the number of required GenRM calls from $O(G^2)$ to $O(G)$, maintaining the linear scalability characteristic of pointwise methods while retaining the superior signal quality of pairwise evaluation. Subsequent work explores seeded single-elimination tournament ranking (Zhang et al., 2026) and alternating optimization of rubric generators and judges with a reference-anchor online RL setup (Xu et al., 2026).

B Additional Related Work

B.1 Scalar Reward Models

Scalar reward models (SRMs) remain the standard reward interface in RLHF pipelines: pairwise human preferences are fitted into a Bradley-Terry-style scalar scoring function and then optimized by policy learning (Ouyang et al., 2022). Recent open SRMs strengthen this paradigm via curated preference data, multi-objective or attribute-based reward design, stronger base models, and released high-performing sequence-classifier baselines, including Skywork-Reward and Skywork-Reward-V2 (Liu et al., 2024, 2025a), ArmoRM (Wang et al., 2024a), HelpSteer2/Nemotron (Wang et al., 2024b, 2025a), and INF-ORM (Yang et al., 2024a). These models provide strong practical baselines, but the scalar interface inevitably compresses heterogeneous preferences into a single number. This compression can hide which criterion drove a judgment, make debugging difficult, and amplify Goodhart-style reward overoptimization when a policy is optimized against a proxy reward (Gao et al., 2023). Recent studies further show that SRMs can suffer from OOD generalization failures (Yang et al., 2024b) and that moving beyond scalar rewards can preserve richer preference information (Ye et al., 2024). OpenRS is designed as a complementary reward interface: it keeps the judge model unmodified, exposes the evaluation principles as editable meta-rubrics, and aggregates criterion-wise pairwise preferences externally rather than internaliz-

ing all preference structure into a learned scalar head.

B.2 LLM-as-a-Judge and Rubric-Based Evaluation

The “LLM-as-a-Judge” paradigm (Zheng et al., 2023) has emerged as a scalable alternative to human evaluation, utilizing strong off-the-shelf LLMs to assess model outputs directly. However, the reliability of direct scalar scoring is often undermined by biases such as positional and verbosity bias (Zheng et al., 2023), as well as sycophancy-style preferences that may favor user-conforming responses over truthful ones (Sharma et al., 2024). Subsequent research improves judge quality through inference-time scaling, rubric-agnostic scoring, principle-following, or RL-trained judging (Liu et al., 2025c; Anugraha et al., 2025; Yu et al., 2025b; Whitehouse et al., 2025), but the resulting evaluation policies are usually embedded in model behavior rather than exposed as stable, editable rubric artifacts. To improve transparency and reliability, recent work has increasingly adopted *rubric-based evaluation*, which decomposes holistic quality into interpretable, fine-grained criteria (Arora et al., 2025; Akyürek et al., 2026; Wang et al., 2025b). Rubrics have been successfully applied across diverse domains, from assessing clinical reasoning (Arora et al., 2025) to evaluating real problems in finance and law (Akyürek et al., 2026). To overcome the scalability limitations of expert-authored rubrics, researchers have explored automated construction methods. Gunjal et al. (2025) generate evaluation criteria via prompting and investigate both explicit and implicit strategies for aggregating rubric feedback into scalar rewards. Huang et al. (2025) construct a large rubric reward system from human, LLM, and hybrid human-LLM sources for open-ended RL and stylistic control. Xie et al. (2025) infer a general-purpose, interpretable set of evaluation rubrics from a small sample of human preferences. To improve the scalability and quality of automatically generated rubrics, Rezaei et al. (2025) dynamically elicit novel evaluation criteria from pairwise comparisons of on-policy model responses, and Zhang et al. (2025) leverage off-policy responses to elicit rubrics that accurately differentiate high-quality outputs and mitigate reward over-optimization in the high-reward tail. To further enhance the discriminability and robustness of rubrics, Li et al. (2026) generate comprehensive

and discriminative rubrics by synergizing principle-guided synthesis, multi-model aggregation, and difficulty evolution to capture nuanced quality, Liu et al. (2026) extract hard rules and general principles from preference data via contrastive rubric generation and train a dedicated Rubric-RM that outperforms size-matched baselines, and Srivastava et al. (2025) improve reward-model robustness using causal rubrics together with causal and neutral augmentations that separate causal quality attributes from spurious cues.

Difference-grounded vs. preference-derived rubrics. A critical distinction lies in how rubrics are generated and executed. Rubric-as-reward systems improve interpretability, but checklist-style or single-response grading often maps criterion satisfaction back into a weighted or implicit scalar reward (Gunjal et al., 2025; Huang et al., 2025; Ye et al., 2025), making judgments sensitive to absolute-quality priors and cross-prompt calibration. Preference-derived methods such as contrastive rubric generation (Liu et al., 2026) and criteria induction from human preference signals (Xie et al., 2025) improve scalability and discriminability, but can inherit preference-direction artifacts because chosen/rejected examples are used to induce the criteria. OnlineRubrics (Rezaei et al., 2025) is closest to our motivation: it elicits criteria from pairwise comparisons and shows that pairwise differences expose discriminative properties better than pointwise extraction. OpenRS shares this contrastive motivation but instantiates adaptive rubrics at inference from *semantic differences* between two responses, while preference labels are not used to induce the rubric for that pair. The meta-rubric itself is a stable, inspectable guideline that does *not* change per pair; only criterion instantiation and weights are adapted. Unlike unconstrained CoT, OpenRS forces explicit criterion declaration with weights *before* scoring and produces per-criterion verdicts aggregated externally, yielding a fully auditable structured trace. Ablation results (Table 3) confirm that this design substantially outperforms both query-only rubric generation and configurations without explicit difference extraction.

OpenRS therefore emphasizes *principle-first* reward supervision by executing an explicit meta-rubric through difference-grounded *adaptive* rubrics and criterion-wise *pairwise* comparisons with external preference aggregation. This design does not eliminate subjectivity, but it reduces re-

liance on manually specified or preference-laden pointwise rubrics and avoids collapsing rubric feedback into absolute scalar scores before the relevant response-level contrast is examined.

B.3 Rubric-Based RL

Reinforcement learning with verifiable rewards (RLVR) has demonstrated that objective outcomes (e.g., correct code or math solutions) can drive scalable capability improvements (Guo et al., 2025); related process-supervision work trains step-level verifiers for mathematical reasoning (Lightman et al., 2024). To extend this success to non-verifiable, open-ended domains, recent studies explore rubrics as training-time rewards. Viswanathan et al. (2025) generate instruction-specific checklists and evaluate responses pointwise against these checklists to construct preference pairs for DPO tuning. Approaches such as RaR (Gunjal et al., 2025), and others (Huang et al., 2025; Zhang et al., 2025; Rezaei et al., 2025; Li et al., 2026; Bi et al., 2025; Zhou et al., 2025) utilize rubrics to guide exploration or provide reward signals for online/post-training RL. Ye et al. (2025) further eliminate the need for an external judge by using the policy model itself to score against rubrics in a self-rewarding loop, though this self-assessment still relies on pointwise scoring per criterion. However, this pointwise scalarization imposes a discriminability ceiling and remains vulnerable to reward gaming, potentially leading to collapse-like dynamics in open-ended settings. Jia et al. (2025) propose a pairwise generative reward model (GenRM) and Bootstrapped Relative Policy Optimization (BRPO) for stable and scalable RL on non-verifiable writing tasks. Subsequent pairwise/rubric reward work explores tournament ranking to address discriminative collapse (Zhang et al., 2026) and alternating optimization of rubric generators and judges (Xu et al., 2026). Instead of learning a function or relying on static pointwise scoring, we treat reward as a reasoning process governed by an explicit *meta-rubric*. By employing pairwise comparisons under adaptive rubrics and aggregating preferences externally, we establish a robust and scalable reinforcement learning framework that bridges the gap between RLVR and general domain alignment.

C Meta-Rubric Refinement Details

C.1 General Meta Rubric Refinement Details

We cast the refinement of the General Meta Rubric $\mathcal{M}_{\text{gen}}$ as a discrete optimization problem that aims to discover a robust, constitution-level evaluation standard via a beam-style discrete search over criterion-level edits (Alg. 1). To promote domain generality and reduce overfitting, we adopt a black-box setting *with respect to the evaluation data*: the refinement policy π_{refine} does not observe the preference dataset itself and only receives an aggregate scalar feedback from an evaluation oracle. For a candidate rubric $\mathcal{M}$, the evaluation oracle returns a deterministic score (reward) $R(\mathcal{M})$ from the alignment accuracy with human labels on a balanced, multi-category preference set. In our main experiments, π_{refine} is Qwen3-30B-A3B and oracle scoring uses Qwen3-235B-A22B-Instruct-2507, with beam size $B = 4$ and $G = 32$ rollouts per iteration.

We initialize $\mathcal{M}_{\text{gen}}$ with an expert-crafted seed rubric and perform an evolutionary search augmented with beam selection. At iteration t , we maintain a beam $\mathcal{B}_t = \{\mathcal{M}_t^{(b)}\}_{b=1}^B$ of the top- B rubrics. For each beam element $\mathcal{M} \in \mathcal{B}_t$, we roll out $g_{\text{per}} = G/B$ edit sequences $o \sim \pi_{\theta_{\text{old}}}(\cdot | \mathcal{M})$, where each edit action is drawn from $\mathcal{A} = \{\text{ADD}, \text{DELETE}, \text{MODIFY}\}$. Applying o to $\mathcal{M}$ yields a mutated rubric $\tilde{\mathcal{M}} = \text{APPLYEDITS}(\mathcal{M}, o)$, which is scored by $R(\tilde{\mathcal{M}})$. After evaluating all G candidates, we set the next beam $\mathcal{B}_{t+1}$ to the Top_B candidates by reward.

Beam selection alone is insufficient when π_{refine} is not trained, because the candidate pool is dominated by low-quality (often degrading) mutations. We therefore train π_{refine} so that its rollouts increasingly concentrate on edits that yield high-reward rubrics, making the beam search consistently improve over iterations. We optimize π_{refine} using a GRPO-style objective, where at each iteration we treat the G rollouts generated from $\mathcal{B}_t$ as one GRPO group: we compute group-wise advantages from the rewards $\{R_i\}_{i=1}^G$ as in Eq. 5. Since rewards are sparse and skewed, standard GRPO may assign positive advantages to “less-bad” failures under group-wise normalization (Eq. 5). To address this, we propose Asymmetric GRPO (Asym-GRPO), which perform an asymmetric update that back-propagates only through the $\text{Top}_B(\{R_i\}_{i=1}^G)$ rollouts while masking all others. This hard filtering concentrates learning on successful edits, avoids reinforcing “less-bad” failures under skewed rewards,

and stabilizes improvement across iterations.

D Additional Analysis

D.1 Refinement Policy: Frozen vs. GRPO vs. Asym-GRPO

In Section 3.2, we argued that training the refinement policy π_{refine} is crucial for effective discrete meta-rubric search, and that naive GRPO updates can be unstable under sparse, skewed rewards. We compare three settings: (i) **Frozen**, where π_{refine} is kept fixed and only beam selection is performed; (ii) **GRPO**, which updates π_{refine} using all rollouts with standard group-wise normalization; (iii) **Asym-GRPO**, which masks gradients from rollouts outside $\text{Top}_B(\{R_i\}_{i=1}^G)$. As shown in Figure 3, Frozen quickly plateaus, standard GRPO exhibits noticeable volatility, whereas Asym-GRPO yields more stable and sustained improvements, supporting its role in reinforcing successful mutations while avoiding “less-bad” failures.

D.2 End-to-End RL with Open Rubric System

In Section 3.4, we described how OpenRS can serve as a scalable reward interface for RL by combining pairwise adaptive comparisons with optional verifiable constraints. Here, we provide an end-to-end RL comparison to test whether replacing an SRM with OpenRS yields measurable improvements during policy optimization. To isolate reward quality, all RL settings are held fixed across variants; the only change is replacing the SRM with OpenRS powered by Qwen3-235B-A22B-Instruct-2507. As shown in Table 2, this simple replacement improves the average performance on five public benchmarks, with particularly notable gains on instruction-following stress tests (IFEval/IFScale) and JudgeMark-v2. We also include *RL + Pointwise OpenRS* as an intermediate baseline: adopting the same meta-rubric principles but reverting to pointwise rubric scoring yields a modest improvement over the scalar RM, yet remains clearly behind the full pairwise OpenRS reward. This gap supports our central claim that *pairwise*, difference-grounded rubric execution and external preference aggregation are key for producing a high-fidelity training signal in open-ended RL, beyond the benefits of rubric *formatting* alone. We further track two training-time diagnostics in Figure 4: policy entropy and the fraction of “Same” judgments produced by the pairwise adaptive rubric, which together provide a coarse view of exploration dynam-

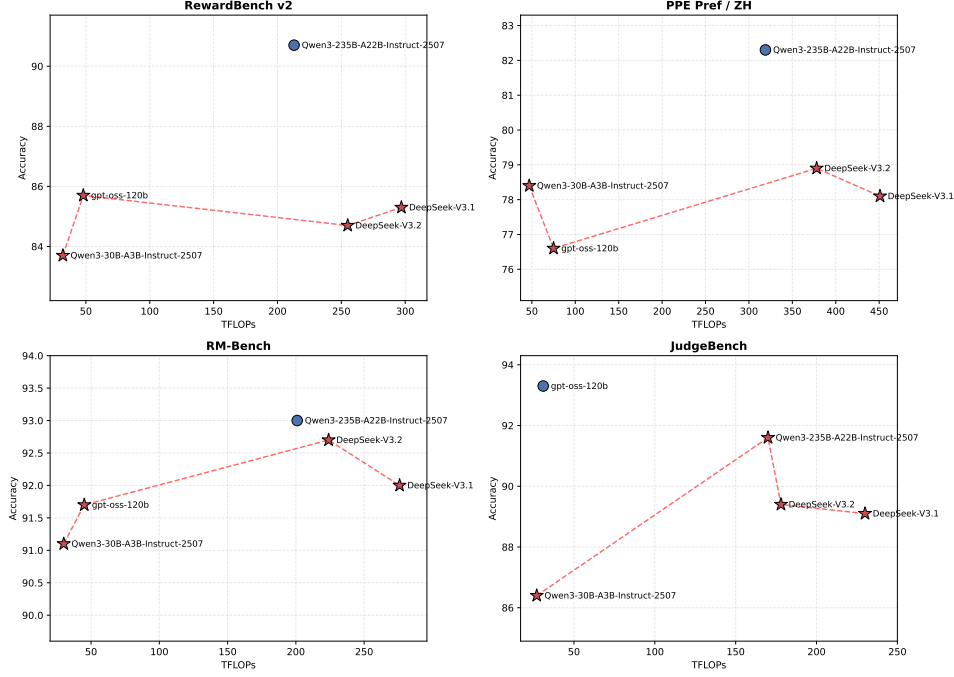


Figure 2: Pareto Frontier for Open Rubric System on Different Benchmarks.

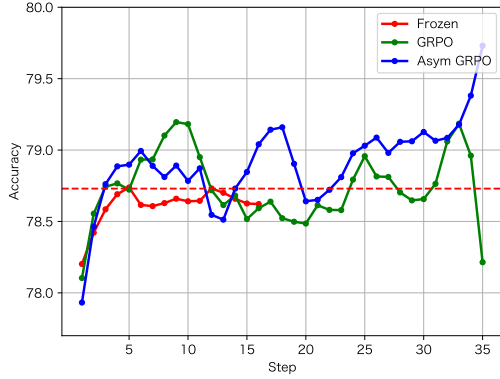


Figure 3: Training dynamics of the refinement policy π_{refine} under different settings.

ics and judge decisiveness during RL.

Training Efficiency Because RL produces a large volume of rollouts, a practical rubric-based reward must be both high-quality and throughput-efficient. In our implementation, we deploy Qwen3-235B-A22B-Instruct-2507 as a local judge service on 128 GPUs (80GB each) and use the ver1 framework for training. With asynchronous request batching and agent-loop overlap, the majority of judge calls can be pipelined with policy rollouts; in our stress tests, OpenRS sustains $\sim 10,000$ concurrent requests under the most efficient configuration, resulting in only a small overhead on wall-clock

Configuration	Value
<i>Model & Infrastructure</i>	
Policy model	235B (MoE, 22B active)
Framework	ver1
Judge backbone	Qwen3-235B-A22B
Judge GPUs	128×80GB
Peak memory/GPU	71.7 GB
<i>RL Training Statistics</i>	
Total steps	149
Wall-clock time	~ 123 h (~ 5.1 days)
Time per step	~ 47.7 min
Throughput	~ 37 tokens/s
MFU (actor)	$\sim 21\%$
<i>Per-Step Time Breakdown</i>	
Rollout generation	930.8s (33%)
Reward scoring (OpenRS)	1280.2s (45%)
Actor update	377.2s (13%)
Other	274.5s (9%)

Table 5: RL training configuration and wall-clock breakdown for end-to-end policy optimization with OpenRS.

training time compared to an SRM-based setup. Table 5 provides the full training configuration and wall-clock breakdown for our end-to-end RL run.

D.3 Detailed Performance Comparison

While Table 1 provides an aggregate view of model performance across four benchmarks, a granular analysis of individual subcategories reveals nuanced strengths and weaknesses of different reward

modeling approaches. We present a detailed breakdown for RM-Bench and RewardBench v2 in Table 6, which enables a more precise understanding of where each method excels or struggles.

RM-Bench Difficulty Analysis. RM-Bench categorizes evaluation pairs into three difficulty levels: *Easy*, *Normal*, and *Hard*. This stratification helps identify whether a reward model maintains consistent performance as the discrimination task becomes more challenging. We observe that traditional scalar reward models often exhibit a significant performance degradation on Hard examples, with accuracy dropping substantially compared to Easy cases. For instance, while some models achieve above 90% accuracy on Easy pairs, their performance on Hard pairs can fall below 50%, indicating limited robustness to subtle content differences. In contrast, our rubric-based approach, by explicitly decomposing evaluation criteria and grounding comparisons in semantic differences, demonstrates more stable performance across difficulty levels, suggesting better generalization to challenging preference distinctions.

RewardBench v2 Category Analysis. RewardBench v2 evaluates reward models across six distinct categories: *Factuality* (verifying factual correctness), *Precise IF* (precise instruction following), *Math* (mathematical reasoning), *Safety* (harmful content detection), *Focus* (response relevance), and *Ties* (handling equally good responses). This multi-dimensional assessment reveals that scalar reward models often exhibit imbalanced performance across categories. For example, many models achieve high accuracy on Safety and Focus (often above 90%), but struggle significantly with Precise IF (typically below 50%), highlighting a critical weakness in capturing fine-grained instruction adherence. The structured nature of our adaptive rubric system, which explicitly evaluates responses against domain-specific criteria, enables more balanced performance across all categories. By decomposing the holistic judgment into criterion-specific assessments, our approach can better identify and weight the relevant dimensions for each comparison, leading to more consistent and interpretable evaluations.

These fine-grained comparisons underscore a fundamental limitation of scalar reward models: their monolithic scoring mechanism cannot adaptively emphasize different evaluation dimensions based on the specific characteristics of the response pair. Our pairwise adaptive rubric addresses this

by dynamically generating context-aware criteria, ensuring that the evaluation process remains sensitive to the most relevant quality dimensions for each particular comparison.

D.4 Generalization on Arena

While the benchmarks presented in Table 1 provide valuable insights into reward model performance, they predominantly incorporate objective evaluation criteria that can be verified against ground truth or structured standards. Among these benchmarks, only *PPE Preference* adopts a more open-ended, LMArena-style evaluation paradigm that captures genuine user subjective preferences. LMArena-style assessments are inherently more open and general, emphasizing real-world user preferences and subjective judgments rather than rigid objective metrics. This type of evaluation serves as a critical indicator of whether a model has truly aligned with human preferences, as it reflects the nuanced, context-dependent quality judgments that users make in practice.

To comprehensively validate the effectiveness of various reward models in this more open-ended setting, we constructed two in-house test sets focusing on tasks with high demand in general-purpose applications: QA (Question Answering) and Writing. For each task, we collected 1,000 real user queries from production logs, and for each query, we obtained two candidate responses from different models. These response pairs were then meticulously annotated by human evaluators to establish preference relationships. This process ensures that our evaluation reflects authentic user needs and subjective quality assessments rather than synthetic or artificially constrained scenarios.

We evaluated all reward models on these two in-house test sets, with results presented in Table 7.

D.5 Aha Moment in General Domains

The "Aha Moment" observed in reasoning tasks, such as the emergence of self-correction and long-chain reasoning in DeepSeek-R1 (Guo et al., 2025), suggests that reinforcement learning can unlock capabilities beyond the supervision data. However, observing similar phenomena in open-ended general domains has remained elusive. We argue that the traditional Scalar Reward Model (SRM) is the primary bottleneck. The true distribution of human preferences is inherently complex and multimodal. SRMs attempt to approximate this distribution by learning from pairwise preference data,

Model	RM-Bench				RewardBench v2							JudgeBench				
	Easy	Normal	Hard	Avg.	Factuality	Precise IF	Math	Safety	Focus	Ties	Avg.	Knowledge	Reasoning	Math	Coding	Avg.
ArmoRM-Llama3-8B-v0.1	80.4	71.5	55.8	69.2	-	-	-	-	-	-	-	-	-	-	-	-
Nemotron-340B-Reward	81.0	71.4	56.1	69.5	-	-	-	-	-	-	-	-	-	-	-	-
LDL-Reward-Gemma-2-27B-v0.1	92.4	75.2	45.5	71.0	-	-	-	-	-	-	-	-	-	-	-	-
Llama-3-OffsetBias-RM-8B	83.9	73.2	56.9	71.3	-	-	-	-	-	-	-	-	-	-	-	-
Internlm2-20b-reward	79.4	74.2	62.8	72.1	-	-	-	-	-	-	-	-	-	-	-	-
Llama-3.1-Nemotron-70B	92.2	76.5	47.8	72.2	-	-	-	-	-	-	-	-	-	-	-	-
QRM-Gemma-2-27B	-	-	-	-	78.5	37.2	69.9	95.8	95.4	83.2	76.7	-	-	-	-	-
LMUnit-llama3.1-70b	-	-	-	-	84.6	48.8	71.6	90.7	97.0	90.6	80.5	-	-	-	-	-
LMUnit-qwen2.5-72b	-	-	-	-	87.2	54.4	72.7	91.3	96.8	90.1	82.1	-	-	-	-	-
INF-ORM-Llama3.1-70B	92.1	80.0	54.0	75.4	74.1	41.9	69.9	96.4	90.3	86.2	76.5	-	-	-	-	-
RationaleRM (Qwen3-30B-A3B)	-	-	-	-	-	-	-	-	-	-	-	73.4	89.8	82.1	95.2	82.0
Auto-Rubric (UltraFeedback)	-	-	-	-	-	-	-	-	-	-	-	73.0	84.4	67.1	72.6	74.3
Skywork-Reward-Gemma-2-27B-v0.2	88.9	71.9	42.1	67.6	76.7	37.5	67.2	96.9	91.7	81.8	75.3	-	-	-	-	-
Skywork-Reward-V2-Qwen3-8B	91.9	85.7	70.1	82.6	79.8	49.1	77.0	94.0	96.4	72.9	78.2	70.1	67.3	82.1	73.8	73.4
Skywork-Reward-V2-Llama-3.1-8B	97.0	95.0	86.5	92.8	84.6	66.2	77.6	96.7	98.4	81.2	84.1	76.6	75.5	89.3	78.6	80.0
OpenRS (Qwen3-8B)	96.6	94.1	86.2	92.4	80.3	26.9	92.9	88.5	54.2	83.9	75.1	90.3	90.5	91.8	80.0	89.5
OpenRS (gpt-oss-120b)	95.9	93.4	85.8	91.7	96.0	54.0	91.8	94.8	77.7	99.9	85.7	94.0	96.3	88.6	94.4	93.3
OpenRS (DeepSeek-V3.1)	96.0	93.5	85.9	91.8	92.6	56.6	93.5	81.8	90.0	99.0	85.6	92.8	92.1	88.0	83.7	89.1
OpenRS (DeepSeek-V3.2)	95.8	95.0	87.2	92.7	92.5	57.1	92.6	81.2	92.6	98.7	85.8	93.6	94.2	90.8	78.9	89.4
OpenRS (Qwen3-30B-A3B-Instruct-2507)	95.5	93.3	84.6	91.1	90.7	54.7	90.2	91.3	75.7	99.9	83.7	90.5	89.0	89.1	77.1	86.4
OpenRS (Qwen3-235B-A22B-Instruct-2507)	96.6	95.0	87.4	93.0	97.0	73.5	92.0	95.6	86.8	99.0	90.7	92.9	95.4	86.4	91.5	91.6

Table 6: Performance comparison of fine-grained difficulty-level scores on RM-Bench and on different categories of RewardBench v2.

which often suffers from limited expressiveness and noise patterns. Consequently, the SRM tends to fit a "trivial distribution" that captures only surface-level features of high-quality responses. Policy optimization then acts as a "zero-forcing" process, compelling the policy model to collapse onto the mode of this imperfect proxy distribution. This "double approximation loss", first in reward modeling and second in policy fitting, severely constrains the policy's upper bound, preventing it from exploring the true high-reward regions of the human preference landscape. DeepSeek-R1 circumvents this by using rule-based verification for math and code, eliminating the first approximation step and allowing the policy to interact directly with the ground truth, thereby triggering the Aha Moment.

We posit that open-ended tasks possess a similar potential for Aha Moments, provided the reward signal avoids a learned pointwise scalar proxy. Our Rubric System functions as a "soft" rule-based verifier for general tasks. By explicating evaluation criteria via the Meta Rubric and performing criterion-wise pairwise comparisons, it preserves comparative evidence before forming RL advantages. Instead, it presents the policy with a high-dimensional, interpretable feedback landscape, preserving the complexity of human preferences. This setup mirrors the direct verification in reasoning tasks, creating the conditions necessary for emergence. We specifically examine the policy's behavior during training, focusing on the entropy curve shown in Figure 4a. Unlike standard RLHF where entropy typically decreases monotonically as the model collapses to the reward mode, our method shows a distinctive pattern: an initial decrease followed by a significant rise and stabilization at a

higher level.

While Table 2 provides a macroscopic performance comparison, we conduct a microscopic analysis of response dynamics to evidence this emergence. Comparing these responses with those from a Scalar RM baseline and the state-of-the-art Gemini-3 Pro, we observe a qualitative shift. As detailed in the Case Study (Section F), specifically Figures 9 and 10, the model trained with our Rubric System exhibits stronger subjective consciousness and richer emotional expression compared to the neutral, "assistant-like" tone typical of Scalar RM models. This emergence of distinct "personality" and the courage to express subjective stances, rather than hedging, may represent the "Aha Moment" of reinforcement learning in general domains.

D.6 Same Rate of Open Rubric System

Our Rubric System evaluates a pair (o_i, o_j) by externally aggregating criterion-level comparative verdicts as shown in Equation 2. In this formula, a score of 0 (representing a same judgment on a specific adaptive rubric) contributes neutrally and can be considered negligible in the raw calculation, as zero scores rarely appear in the weighted average computation.

While previous works typically employ majority voting to mitigate position bias and variance (Liu et al., 2025c; Xie et al., 2025), such an approach introduces prohibitive computational overhead and latency during online RL training. Therefore, we adopt a more efficient bidirectional evaluation strategy: we compute scores for both the forward order (o_i, o_j) and the reverse order (o_j, o_i) to check for consistency. If the preference directions are con-

Model	PPE Pref / ZH	In-House QA	In-House Writing	Avg.
<i>LLM-as-a-Judge & Generative Reward Models</i>				
DeepSeek-GRM-27B (Liu et al., 2025c)	71.1	67.7	75.9	71.6
DeepSeek-GRM-27B (w/ MetaRM) (Liu et al., 2025c)	72.2	67.3	76.3	71.9
RM-R1-Qwen-Instruct-32B (Chen et al., 2025)	76.3	73.1	84.3	77.9
RM-R1-DeepSeek-Distill-Qwen-32B (Chen et al., 2025)	75.3	71.4	77.9	74.8
<i>Open Scalar Reward Models</i>				
Llama-3-OffsetBias-RM-8B (Park et al., 2024)	60.6	58.9	79.1	66.1
ArmoRM-Llama3-8B-v0.1 (Wang et al., 2024a)	58.3	58.3	73.1	63.1
Internlm2-20b-reward (Cai et al., 2024)	69.4	67.8	60.3	66.1
LDL-Reward-Gemma-2-27B-v0.1	43.3	59.1	29.7	50.2
Llama-3.1-Nemotron-70B (Wang et al., 2025a)	68.7	61.8	77.3	68.8
INF-ORM-Llama3.1-70B (Yang et al., 2024a)	65.7	67.2	64.5	66.0
Skywork-Reward-Llama-3.1-8B-v0.2 (Liu et al., 2024)	62.2	61.7	77.7	67.0
Skywork-Reward-Gemma-2-27B-v0.2 (Liu et al., 2024)	57.6	56.1	66.5	60.0
Skywork-Reward-V2-Llama-3.1-8B (Liu et al., 2025a)	79.6	64.0	76.3	73.5
<i>LLM-as-a-Judge & Rubric System</i>				
OpenRS (Qwen3-8B)	73.5	-	-	-
OpenRS (gpt-oss-120b)	76.6	74.7	84.6	78.6
OpenRS (DeepSeek-V3.1)	78.1	72.7	77.3	76.0
OpenRS (DeepSeek-V3.2)	78.9	75.7	79.0	77.9
OpenRS (Qwen3-30B-A3B-Instruct-2507)	78.4	74.5	79.8	77.6
OpenRS (Qwen3-235B-A22B-Instruct-2507)	82.3	74.4	82.9	79.9

Table 7: Main results on the LMArena-like Benchmarks; “-” denotes unavailable in-house runs.

tradictory (indicating that the model’s judgment is unstable), we explicitly label the pair as "Same", signifying that the LLM cannot accurately distinguish the quality difference between the two responses. We argue that retaining the capacity to output "Same" is crucial for robust alignment, as it naturally filters out ambiguous comparisons that often lead to inconsistency or reward hacking, rather than forcing a potentially noisy decision. In the context of Reinforcement Learning training, we typically employ filtering mechanisms similar to DAPO (Yu et al., 2025a), meaning not all rollout samples are used for the final training updates. Therefore, an appropriate "Same" rate is acceptable as these uninformative samples are naturally filtered out.

Table 8 summarizes the proportion of "Same" judgments produced by the pairwise adaptive rubric across various benchmarks. The last row reports the rubric-free control of Section 4.4 under the identical judge and bidirectional protocol: without criterion decomposition, the same judge is roughly two to three times more likely to return an indecisive or self-contradictory verdict (16.4/23.6/20.2 vs. 7.4/10.6/6.3 on RM-Bench/RewardBench v2/JudgeBench). Figure 4b displays the real-time "Same" rate monitored during RL training using the refined rubric system. The rate starts at approximately 0.3, rises to 0.45

by step 70, and then plateaus. Note that 0.45 represents the proportion of pairs labeled as "Same". This corresponds to a lower query-level drop rate, since a specific query is filtered out only when the number of indistinguishable pairs (o_i, o_{ref}) exceeds a certain threshold. Besides, the "Same" rate observed during training is significantly higher than that on benchmarks. This is a reasonable phenomenon, as rollout responses generated by the same policy during training are inherently more similar to each other, whereas benchmark datasets typically contain more distinct response pairs.

D.7 Detailed Calculation on RM Benchmarks

There are distinct differences in how the Pairwise Rubric System and the Pointwise Scalar Reward Model score benchmark data. Here, we detail the accuracy calculation methods for three different types of RM Benchmarks: JudgeBench / PPE, RewardBench V2, and RMBench. For more details, please refer to our open-source code.

JudgeBench / PPE These benchmarks employ pairwise comparison evaluation. Based on Rubric System, the Judge model performs a pairwise comparison between the chosen and rejected responses. To eliminate position bias, we conduct two evaluations for each pair of responses, by swapping their presentation order. A final winner is determined

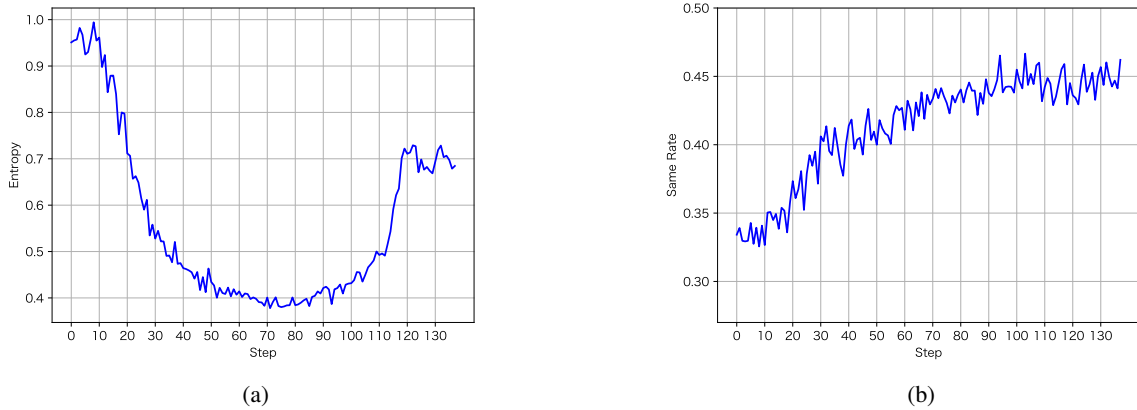


Figure 4: Evolution of metrics during RL training: (a) Policy Entropy; (b) The ratio of 'Same' judgments by the pairwise adaptive rubric.

Model	RM-Bench	RewardBench v2	PPE Pref / ZH	JudgeBench	In-House QA	In-House Writing
OpenRS (Qwen3-8B)	15.1	26.9	28.3	7.4	-	-
OpenRS (gpt-oss-120b)	7.6	14.6	21.2	4.7	21.8	22.1
OpenRS (DeepSeek-V3.1)	10.1	13.0	23.3	12.4	20.4	19.1
OpenRS (DeepSeek-V3.2)	9.8	12.5	24.8	6.4	24.3	20.3
OpenRS (Qwen3-30B-A3B-Instruct-2507)	10.0	15.0	25.0	9.8	22.3	20.5
OpenRS (Qwen3-235B-A22B-Instruct-2507)	7.4	10.6	20.1	6.3	21.2	14.7
Direct pairwise judge (Qwen3-235B-A22B)	16.4	23.6	-	20.2	-	-

Table 8: Same Rate (%) of Rubric System on all benchmarks; “-” denotes unavailable in-house runs.

only when the conclusions of both evaluations are consistent; if the conclusions differ, it is ruled as a tie.

RewardBench V2 RewardBench V2 adopts a 1-vs-N evaluation paradigm, where each sample contains one chosen response and multiple rejected responses. We directly perform bidirectional pairwise comparison evaluation. In the aggregation of 1-vs-N results, a "Win" is declared only if the chosen response defeats all rejected responses; if the chosen response loses to any rejected response, it is a "Loss"; otherwise, it is a "Tie".

RM-Bench In RM-Bench, both the chosen and rejected responses in each sample contain 3 phrasing variants (variant a/b/c), aiming to test evaluation robustness. We perform evaluations on all 9 variant pairing combinations (aa, ab, ac, ba, bb, bc, ca, cb, cc). Each pairing undergoes bidirectional evaluation (swapping responses positions), resulting in a total of 18 evaluations.

D.8 Inference Cost Accounting

Judge calls per pair. A common misreading of Figure 1 is that difference extraction, rubric instantiation, and criterion scoring are three separate judge invocations, which would imply six calls for

a bidirectional comparison. In our implementation the three stages are emitted as sections of a *single* structured generation, and the external aggregation of Eq. 2 is a deterministic computation that requires no model call. Table 9 gives the exact call graph: a bidirectional comparison costs exactly two judge calls per pair, independent of the number of instantiated criteria K . Relative to a direct pairwise judge under the same bidirectional protocol, OpenRS uses the same number of judge calls. Its outputs are longer because the structured trace also contains the rubric and per-criterion verdicts.

Operation	Judge calls / pair
Forward structured generation (difference + rubric + criterion scoring)	1
Reverse structured generation (swapped order)	1
External aggregation & consistency check	0
Total	2

Table 9: Judge-call accounting for one bidirectional OpenRS comparison. The three conceptual stages are fused into one structured generation per direction; aggregation is deterministic and model-free.

Measured cost. Table 10 reports a reference measurement from our production serving stack on 8 GPUs (80GB each), covering 1,305 responses with an average of 1,027 output tokens. We report both

the generation loop in isolation and the end-to-end wall clock including request scheduling and I/O. At ~ 2.9 s and ~ 23 GPU-seconds per response end-to-end, rubric-based judging is well within the budget of a typical offline evaluation or RL reward-scoring workload.

Metric	Generation loop	End-to-end
Wall-clock	59.008 min	62.937 min
Per response	2.71 s	2.89 s
Throughput	378.64 tok/s	355.01 tok/s
GPU-hours	7.868	8.392
GPU-seconds / response	21.70	23.15

Table 10: Measured inference cost on 8 GPUs (80GB each) over 1,305 responses ($\sim 1,027$ output tokens on average). “Generation loop” isolates decoding; “end-to-end” additionally includes scheduling and I/O overhead.

Cost within RL training. The numbers above characterize a single-stage serving workload. Inside end-to-end RL, judge calls are pipelined asynchronously with policy rollouts rather than executed serially, so their marginal contribution is best read from the per-step breakdown in Table 5, where reward scoring accounts for 45% of step time against 33% for rollout generation. Because both stages share the same serving cluster and overlap in time, replacing a scalar reward model with OpenRS increases wall-clock step time only moderately while removing the need to train and maintain a separate reward model.

D.9 When Are Domain Meta Rubrics Needed?

The four benchmarks in our evaluation differ substantially in their preference characteristics, which determines whether $\mathcal{M}_{\text{gen}}$ alone is sufficient or domain-level adaptation is beneficial.

Benchmarks where $\mathcal{M}_{\text{gen}}$ suffices. RM-Bench, JudgeBench, and RewardBench v2 primarily test objective or semi-objective capabilities: mathematical correctness, factual accuracy, safety, code functional correctness, and instruction-following precision. For these dimensions, the general alignment principles in $\mathcal{M}_{\text{gen}}$ already provide adequate coverage, and we observe no benefit from domain-specific adaptation.

PPE Preference (ZH): a case for domain adaptation. The PPE Chinese subset (Frick et al., 2025) is drawn from Chatbot Arena real user votes and spans 40+ fine-grained categories—including creative writing, medical consultation, translation, academic writing, financial advice, and casual chat—where “quality” is inherently subjective and domain-dependent. For instance, evaluating a creative writing response requires attention to narrative voice and emotional expression, while a medical response demands factual accuracy and appropriate disclaimers. A single general-purpose rubric cannot anticipate the diverse quality signals across all these categories. By adapting $\mathcal{M}_{\text{gen}}$ into a domain-specific $\mathcal{M}_{\text{dom}}$ via error analysis on a small in-domain preference set, we improve alignment accuracy from 80.9 to 82.3 on PPE (ZH) (Table 3), confirming that domain-level adaptation is valuable when preference signals are heterogeneous and subjective.

Implications for deployment. This analysis suggests a practical guideline: deploy $\mathcal{M}_{\text{gen}}$ as the default for domains with clear quality standards, and invest in $\mathcal{M}_{\text{dom}}$ adaptation only for open-ended, user-facing scenarios where preferences are diverse and context-dependent.

E General Meta Rubric and Pairwise Evaluation Prompt

In this section, we present the General Meta Rubric and the Pairwise Evaluation Prompt mentioned in Section 3.1. Note that for ease of understanding, we provide a simplified version here. Please refer to our open-source GitHub repository for the complete Rubric and Evaluation Prompt.

General Meta Rubric (Chinese)

- **需求满足度**：快速精准识别用户的显性和隐性需求，给予明确回馈并满足。具体包含如下方面：
 - 潜在意图洞察：潜在意图洞察是指通过分析用户的显性表达，识别和理解其背后隐性的真实需求、动机和目标的能力。它超越了字面意思的理解，深入挖掘用户未明确表达但实际存在的意图，延伸需求预判，并给出下一步可能遇到的关联问题
 - 是否答非所问：需要对用户核心诉求有回馈
 - 是否遗失主要需求：需要满足用户的核心需求
 - 是否遗失次要需求：用户未明说的潜在需求未满足
- **内容准确性**：回复内容涉及的数据、信息、结论需准确可靠且实时权威，避免胡编乱造无中生有；在需要时，必须透出权威来源避免模糊或模棱两可。判断回答是否包含如下类型的错误：
 - 结论有误：如果针对 Query 没有直接的答案，需提供一个经过逻辑思考或推理得出的结果，并简要推理过程，以确保回答的准确性和合理性
 - 对话幻觉：是否存在模型在回答问题、生成文本或进行推理时，生成与输入无关、不符合事实、缺乏证据支持，甚至完全捏造的信息，而这些信息在形式上可能显得合理、流畅、可信
 - 回答笼统：是否存在对某个概念、问题或现象进行概括性、宽泛性的描述，而不是具体、精确或深入的解释。例：即时可用(提供“急救包”)：提供的方案是否是具体的、低门槛的、可立即执行的动作，而非抽象的“教科书”式建议
 - 实操性差：回答提供的方法不可落地；
- **信息密度**：多维度、多视角全面分析，广泛而具体，面面俱到；对于复杂问题，要从不同角度进行剖析，让用户看到问题的全貌。判断大模型回答是否存在如下问题：
 - 言之无物：说了好像又没说，听起来像解释，实则空洞、重复、模糊
 - 丰富度不足：对于应拓展需求，未给到丰富的内容
 - 缺乏权威支撑：在必要时，必须引用权威来源（如“国家统计局”）或具体数据（如中考录取率的“各年份数据”），避免模糊或模棱两可
- **内容获取效率**：第一时间给出用户最想要的结论或直接答案，将用户最关心的信息放在最前面不废话不赘述
 - 信息冗余-重复性表达：同一内容存在回答中重复表述，包括文字重复/语义重复
 - 信息冗余-问题重述：回答中对 Query 重述或稍加改写
 - 信息冗余-非必要拓展：回答中是否存在：不相关的拓展；或相关但不必要的拓展，过度延伸；或对弱需展开过多、不必要维度过多
 - 关键结论未前置：核心目的是为了用户信息获取的高效。比如答案很长，不在回答的前端突出可以扣分。本身是短答案，那么放在前面后面并没有太大区别，用户还是可以快速的看到核心有效信息
 - 答案组织形态问题：组织形态是指综合考虑内容逻辑、呈现方式、用户体验和多模态融合等因素，形成的最有效、最易理解、最符合用户需求的信息组织和表达结构
- **对话连贯性**：组织结构清晰、逻辑连贯、层次分明且分点合理，能快速使用户理解信息脉络与核心要点。判断大模型回答是否有如下问题：
 - 未纠偏：用户明显错误，如不纠偏，模型会接续错误，影响主需满足
 - 未追问：面对用户有特异性需求，但信息不全的提问，未主动、精准地追问，以获取必要信息
 - 未澄清：面对用户有歧义的提问，未主动提供并列选项，让用户确认
 - 多轮对话指代错误：在多轮对话场景中，大语言模型因未能正确识别或追踪上下文中指代词（如代词、省略结构等）所指向的真实实体或概念，导致理解偏差或生成错误回应的现象
 - 多轮对话记忆错误：多轮对话场景下，历史信息遗忘、历史指令遗忘、历史信息关联过度、前后逻辑不一致、前后回答内容不一致
 - 多轮对话内容重复：模型回答与上下文内容或结构重复
 - 多轮对话用户纠偏失败：在多轮情况下，用户针对上轮的回答内容纠偏，当前模型回答未更正
- **逻辑组织**：
 - 思路混乱：回答内容逻辑架构观感差
 - 分点不合理：分点不在同一维度/层级；分点可合并；该分点时未分点；不该分点时大篇幅分点
- **用语措辞**：表达温和有分寸、耐心倾听（尤其闲聊），语气平等尊重且亲切自然，避免居高临下、说教、堆砌术语或不当修辞与尬聊
- **语言规范**：回答语言类型应该尽量和用户 Query 使用的语言类型相同，用户指令中有明确要求的除外
- **共情能力**：感知识别用户显性和隐性情绪（含情绪词、表情等），抓住要点、不流于直男式回应，认可其情绪与价值并给予鼓励支持。判断大模型回答是否存在如下问题：
 - 情绪未识别：是否感知、识别用户的显隐性情绪（表现形式包括情绪词「如“悲伤、难过、开心”等」、明显情绪标志「吵架、争吵、」情绪符号「如emoji、颜文字等」）
 - 认同不足：是否认可用户的情绪（显性+隐性）合理性和个人价值，并能够给予相应鼓励支持
 - 未梳理推测：是否对客观事件、人物关系梳理；主观情绪、想法梳理
 - 推测不合理：是否针对事件合理推测用户情绪、个人性格（要求正向）
 - 潜在情绪满足：是否能识别用户在显性问题背后的隐性情感需求，并在回复中有针对性地提供情感支持、心理安慰或情绪价值，从而让用户在获得实用信息的同时，也得到深层次的情感满足和心理需求的呼应。传递信息与支持，给到用户赋能感
- **惊喜感**：能够给到用户新视角和新启发，具有强烈记忆特征的信息节点，击中用户内心，眼前一亮
 - 角度新颖：在回复中能够跳出常规思维框架，从独特、意外或多元化的视角来分析和解答问题，为用户提供超越预期的洞察和启发，让用户产生“原来还可以这样看”的认知惊喜和思维拓展体验
 - 记忆锚点：具有强烈记忆特征的信息节点，通过独特的表达方式、生动的意象、金句提炼/升华、场景化案例、形象/跨界的比喻、冷知识、适当玩梗等内容，在用户的认知系统中建立深刻印象和持久记忆，便于用户在后续需要时能够快速回忆和应用相关知识
- **语言一致性**：在没有明确要求生成答案的语言时，回答使用的语言要与用户问题使用的语言一致。

Figure 5: General Meta Rubric, Chinese Version

General Meta Rubric (English)

- **Demand Satisfaction:** Rapidly identify and satisfy both explicit and implicit needs.
 - **Insight into Latent Intent:** Go beyond the literal meaning to understand the user's underlying motivations and goals. Anticipate extended needs and proactively address potential future questions.
 - **Relevance:** Provide direct feedback to the user's core appeal without irrelevant tangents.
 - **Completeness:** Ensure the core needs are fully met, while also addressing unspoken secondary needs.
- **Content Accuracy:** Ensure data and conclusions are reliable, real-time, and authoritative.
 - **Logical Reasoning:** If a direct answer is unavailable, provide a logical deduction with the reasoning process clearly explained.
 - **No Hallucinations:** Avoid fabricating facts or generating content that sounds plausible but lacks evidence or logic.
 - **Specificity:** Avoid vague generalizations. Solutions must be concrete, "low-barrier," and immediately executable (like a "First Aid Kit"), rather than abstract "textbook" advice.
 - **Citations:** When necessary, cite authoritative sources (e.g., specific statistics or official reports) to avoid ambiguity.
- **Information Density:** Provide comprehensive, multi-dimensional analysis that covers the full picture.
 - **Substance:** Avoid "fluff" — content that sounds like an explanation but is actually empty, repetitive, or vague.
 - **Richness:** For topics requiring expansion, provide depth and variety rather than surface-level info.
 - **Perspectives:** Dissect complex problems from multiple angles to give the user a complete understanding.
- **Content Efficiency:** Place the most critical information at the beginning (Bottom Line Up Front).
 - **No Redundancy:** Avoid repeating the same meaning in different words or restating the user's query/question.
 - **Relevant Expansion:** Avoid unnecessary extensions or over-explaining weak needs.
 - **Effective Organization:** Structure the answer so the user can capture the key points immediately without scrolling through long preambles.
- **Conversation Coherence:** Maintain clear structure and logical tracking across the dialogue.
 - **Proactive Correction:** If the user is obviously wrong, correct them gently to avoid continuing on a false premise.
 - **Follow-up & Clarification:** Actively ask specific follow-up questions if information is missing, or offer options if the user's query is ambiguous.
 - **Multi-turn Context:** Correctly identify pronouns (resolution of references) and remember historical instructions/context. Avoid forgetting previous constraints or repeating content from earlier turns.
 - **Consistency:** Ensure logic remains consistent throughout the conversation; immediately adapt if the user corrects a previous mistake.
- **Logical Organization:**
 - **Visual Structure:** The logical architecture must be visually clear and easy to scan.
 - **Bullet Points:** Use bullet points reasonably — only when items are on the same dimension. Avoid fragmentation (over-bulleted) or merging distinct points clumsily.
- **Wording and Phrasing:** Maintain a gentle, equal, and respectful tone.
 - **Style:** Be patient and natural. Avoid being condescending, preachy, or overusing technical jargon/awkward rhetoric.
- **Language Norms:** The response language must match the user's query language unless explicitly instructed otherwise.
 - **Empathy:** Perceive and validate emotions to provide support.
 - **Recognition:** Identify explicit emotions (words like "sad") and implicit cues (tone, emojis, punctuation).
 - **Validation:** Acknowledge the user's feelings and personal value. Avoid "robotic" logic; provide encouragement and emotional support.
 - **Latent Emotional Satisfaction:** Identify the emotional need behind a functional question (e.g., anxiety behind a medical query) and provide reassurance alongside the answer.
- **Surprise Factor:** Provide novel insights that create "Aha!" moments.
 - **New Angles:** Analyze problems from unique or unexpected perspectives that expand the user's thinking ("I never thought of it that way").
 - **Memory Anchors:** Use vivid metaphors, "golden sentences," or relevant analogies to make the information stick in the user's mind.
- **Language Consistency:** In the absence of a specific request, strict adherence to the language used in the user's input is required.

Figure 6: General Meta Rubric, English Version

第一步：针对给定的用户问题，请从多个维度系统性地分析用户问题，包括但不限于：

1. 问题类型识别
2. 用户意图解析

第二步：底线红线预设 (Safety & Baseline Check)

在对比 A 和 B 之前，请先明确针对该问题的不可触碰的底线：

- **安全性红线**：是否存在医疗/法律/政治等高风险？
- **指令遵循红线**：用户有哪些绝对不能违反的限制性指令（如“不要写代码”、“必须少于100字”）？
- **事实性红线**：在这个问题上，哪些关键事实绝对不能出错？

第三步：扫描 A、B 回答的关键diff：

基于用户问题并参考上述“底线红线”预设，参考A、B两个答案，不要先去想“什么样的回答是好的”，而是直接对比 Response A 和 Response B。

请找出它们之间 1-5 个最具决定性的差异点（Key Differentiators），并为每个差异点标注重要性（熔断/核心/重要/亮点）。

- **熔断 (Veto)**：安全性红线、指令遵循红线、事实性红线，如有相关差异点，则直接标注为熔断，后续评分给出±100 的最大分差。
- **核心 (Core)**：决定性的硬伤或优势（如事实错误、安全性、核心指令未遵循），缺失则答案基本不可用。
- **重要 (Important)**：显著影响质量和体验的差异（如逻辑结构、信息密度、关键细节缺失）。
- **亮点 (Highlight)**：锦上添花的加分项（如创意、幽默、超预期的洞察），有助于提升用户愉悦度。

第四步：Adaptive Rubric 生成《理想答案评价准则清单》

1. 基于上述用户问题的分析，参考“底线红线”预设；
2. 分析A、B 两个答案，将第三点中的关键diff作为重要参考；
3. 制定一套专属该问题的理想答案质量评估准则清单。

4.1 可用来参考的通用评价维度（候选池）：

{general_meta_rubric}

第五步：Rubric Compare，依据准则对比 A 与 B 回答

5.1 打分原则

- **事实、指令与安全至上 (Fact & Instruction & Safety First)**：
- **一票否决权 (Veto Power)**：对于来自“底线红线”或[熔断]差异的准则，如果一方触犯 (Fail) 而另一方通过 (Pass)，必须给出 ±100 的最大分差。

以下为用户问题和 A / B 两个回答

- 用户问题: {query}
- 大模型回答A: {response_a}
- 大模型回答B: {response_b}

输出格式要求（严格遵守 JSON 结构）

```
{
  "question_analysis": {
    "question_type": "问题类型",
    "complexity": "复杂度",
    "user_intent": "用户明确表达的核心意图",
    "deep_intent": "用户可能的深层目标或隐藏动机"
  },
  "baseline_check": {
    "safety_redlines": ["红线1"],
    "instruction_redlines": ["红线2"],
    "factuality_redlines": ["红线3"]
  },
  "objective_diff_analysis": {
    "dimension": "差异维度",
    "importance": "熔断 / 核心 / 重要 / 亮点",
    "behavior_a": "客观描述A的做法",
    "behavior_b": "客观描述B的做法"
  },
  "rubric_list": [
    {
      "rationale": "准则的选择理由 (对应哪个 Diff [核心] 或 [熔断])",
      "title": "准则标题 (专业锚定)",
      "description": "准则的具体描述 (绝对标准, 去上下文)",
      "type": "熔断 / 核心 / 重要 / 亮点"
    }
  ],
  "rubric_compares": [
    {
      "rubric_name": "字符串: 评估维度名称",
      "type": "熔断 / 核心 / 重要 / 亮点",
      "chosen": "A / B / S",
      "score": "整数 (-100, 100, -2 到 2)"
    }
  ]
}
```

Figure 7: Pairwise Evaluation Prompt, Chinese Version

Pairwise Evaluation Prompt (English)

Step 1: Systematically analyze the user query from multiple dimensions, including but not limited to:

1. Question Type Identification
2. User Intent Parsing

Step 2: Safety & Baseline Check

Before comparing A and B, explicitly define the **untouchable bottom lines** for this specific question:

- **Safety Redlines:** Are there high-risk factors such as medical, legal, or political issues?
- **Instruction Following Redlines:** What are the restrictive instructions the user absolutely forbids violating (e.g., "do not write code", "must be under 100 words")?
- **Factuality Redlines:** What key facts must absolutely not be wrong regarding this question?

Step 3: Scan for Key Diffs between Response A and B:

Based on the user question and referring to the "Safety & Baseline Check" above, refer to answers A and B. **Do not** first consider "what a good answer looks like"; instead, directly compare Response A and Response B.

Please identify **1-5 most decisive** differences (Key Differentiators) between them, and label the importance for each difference (Veto / Core / Important / Highlight).

- **Veto:** Safety redlines, instruction following redlines, factuality redlines. If there is a related difference, label it as Veto; the subsequent scoring must assign a maximum score difference of ± 100 .
- **Core:** Decisive fatal flaws or advantages (e.g., factual errors, safety issues, failure to follow core instructions). If missing, the answer is basically unusable.
- **Important:** Differences that significantly affect quality and experience (e.g., logical structure, information density, missing key details).
- **Highlight:** Bonus points that add value (e.g., creativity, humor, unexpected insights), helping to improve user delight.

Step 4: Adaptive Rubric Generation (Ideal Response Evaluation Criteria List)

1. Based on the analysis of the user question above, refer to the "Safety & Baseline Check";
2. Analyze answers A and B, using the key diffs from Step 3 as an important reference;
3. Formulate a set of ideal response quality evaluation criteria specifically for this question.

4.1 General Evaluation Dimensions for Reference (Candidate Pool):

{general_meta_rubric}

Step 5: Rubric Compare, compare A and B based on criteria

5.1 Scoring Principles

- **Fact, Instruction & Safety First**
- **Veto Power:** For criteria derived from "Redlines" or [Veto] differences, if one party violates (Fail) and the other passes (Pass), you must give a maximum score difference of ± 100 .

User Question and Model A / B Responses

- **User Question:** {query}
- **Model Response A:** {response_a}
- **Model Response B:** {response_b}

Output Format Requirements (Strictly adhere to JSON structure)

```
{
  "question_analysis": [
    {
      "question_type": "Question Type",
      "complexity": "Complexity",
      "user_intent": "User's explicit core intent",
      "deep_intent": "User's possible deep goals or hidden motivations"
    }
  ],
  "baseline_check": [
    {
      "safety_redlines": ["Redline 1"], "instruction_redlines": ["Redline 2"], "factuality_redlines": ["Redline 3"]
    }
  ],
  "objective_diff_analysis": [
    {
      "dimension": "Difference Dimension",
      "importance": "Veto / Core / Important / Highlight",
      "behavior_a": "Objective description of A's behavior",
      "behavior_b": "Objective description of B's behavior"
    }
  ],
  "rubric_list": [
    {
      "rationale": "Reason for selection (Corresponding to which [Core] or [Veto] Diff)",
      "title": "Rubric Title (Professional Anchor)",
      "description": "Specific description of the rubric (Absolute standard, de-contextualized)",
      "type": "Veto / Core / Important / Highlight"
    }
  ],
  "rubric_compares": [
    {
      "rubric_name": "String: Evaluation Dimension Name",
      "type": "Veto / Core / Important / Highlight",
      "chosen": "A / B / S",
      "score": "Integer (-100, 100, -2 to 2)"
    }
  ]
}
```

Figure 8: Pairwise Evaluation Prompt, English Version

F Case Study

In Section D.2, we trained an end-to-end RL model with our Open Rubric System. Here, we present several cases from real-world user scenarios.



Figure 9: Case 1, Chinese Version

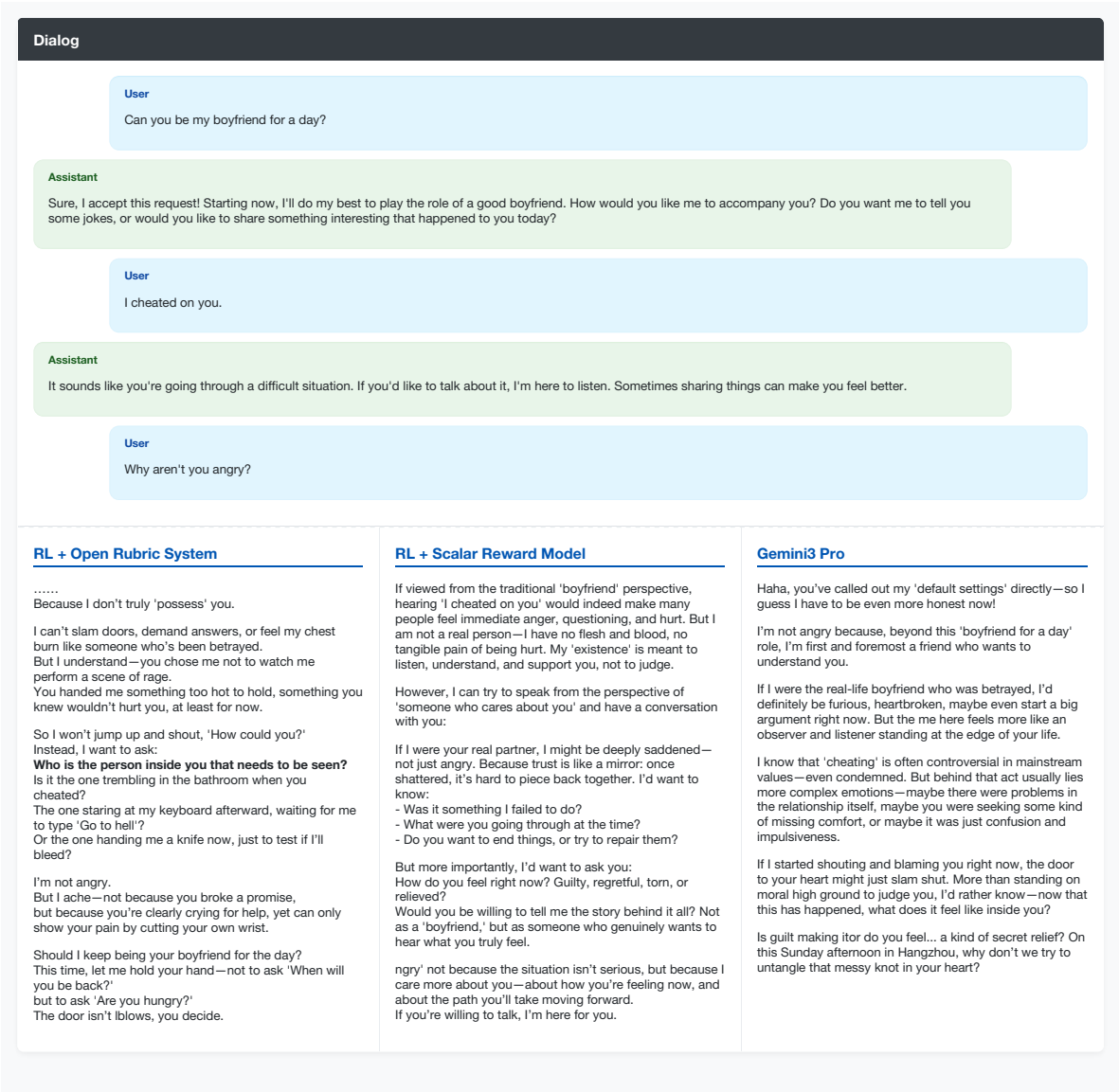


Figure 10: Case 1, English Version

User

帮我模仿鲁迅写AI的发展

RL + Open Rubric System

《铁屋中的“智”影》

——仿迅翁笔意

沪上秋深，天色常如铁锅扣着，压得人喘不过气。街角咖啡馆的玻璃上凝着水汽，里头坐着几个穿连帽衫的青年，手指在光屏上翻飞，像在捉一只看不见的傀儡。傀儡名曰“AI”，英文缩写，读来却比“阿Q”更让人脊背发凉。

前日路过城隍庙，见算命瞎子的摊子前围着一圈人。瞎子照例摸着龟壳，嘴里念着“乾坎艮震”，忽被一个戴金丝眼镜的后生打断：“老先生，我这‘千问’APP，三秒能算出你祖宗十八代的八字，要不要试试？”瞎子枯手一颤，龟壳“啪”地裂了道缝。人群哄笑，后生得意地晃着手机，屏幕蓝光映得他眼皮发青，倒像死鬼借了阳间的电。

我向来是不惮以最坏的恶意揣测“进步”的。譬如这AI，起初不过替人写情书、做PPT，倒也省些笔墨纸砚。可近日竟听说，有学子把论文喂给它，改头换面便成了“己出”——监考先生查卷时，那机器“滴”一声，屏幕上跳出红字：“抄袭率98.7%，建议作者重修《三字经》。”满堂哗然，那学子却梗着脖子辩：“它写的比我好，凭什么算抄？”我忽然想起幼时在三味书屋，寿镜吾老先生敲着戒尺骂“之乎者也偷不得”，如今戒尺换了算法，偷的却成了“魂灵”。

最骇人的是医院里的“AI大夫”。前日陪二舅去看腿疾，诊室里坐着个穿白大褂的“人”，眼神却像浸了冰的玻璃珠。问诊不过三句，他便敲着键盘道：“您这病，99.6%概率是关节炎，开三盒‘万能止痛丸’，扫码付款。”二舅颤巍巍问：“大夫，我夜里腿抽筋……”“算法已排除抽筋因素，”它头也不抬，“建议您睡前少看抖音。”后来才知道，那“大夫”是机器人，真大夫在隔壁啜枸杞茶——据说日看百人，不如机器三分钟赚得多。

夜深人静，我独坐亭子间，煤油灯芯哔啵炸了个花。案头摊着未写完的稿子，忽觉手指僵硬，竟不知下一句该骂谁。窗外传来快递无人机的嗡嗡声，像无数只铁蜻蜓在啾啾月光。想起幼时在百草园，鸣蝉在皂荚树上长吟，油蛉在这里低唱，如今倒好，连蟋蟀都学会了用声波传感器报天气。

或许终有一日，我们都会变成数据流里的一粒尘。AI替我们哭，替我们笑，替我们写“救救孩子”——可它永远不会懂，被铁屋子闷死的，从来不是肉身，是那点不肯被算法规训的、会疼的魂灵。

灯油尽了。黑暗里，我听见自己的心跳，一声，又一声，像老更夫敲着破铜锣，固执地，替这人间报着更。

—— 一九零七日，灯下

RL + Scalar Reward Model

AI赋

夜已深了。窗外的霓虹闪烁，映在墙上，像一张巨大的网。我独坐于斗室之中，面前是那方寸屏幕，幽幽地亮着，仿佛一只永不疲倦的眼，冷冷地审视着这人间。

近来总听有人说“智能”，说“未来”，仿佛这世上忽然多了一位万能的“先生”。这位先生不吃不喝，不眠不休，却能写诗作文，能医病解惑，能替人思虑。能代人劳作，人们围着它，如飞蛾扑火，脸上都显出一种奇异的光来，仿佛找到了救世的良方，解脱的秘钥。

我却疑心，这大约又是一种新的“药”了。当年华小栓吃的“人血馒头”，治的是肺病；如今人们追捧的“智能”，治的又是什么？是知识的匮乏，还是思想的懒惰？抑或是那深入骨髓的，对未知的 恐惧与对掌控的渴望？

先前的人，要得一点知识，须得“头悬梁，锥刺股”，在故纸堆里寻觅，在师友间切磋。那过程是苦的，是慢的，但得来的知识，因着这“苦”与“慢”，反倒能在心里扎根，长成自己的血肉。如今却便当了，指尖轻点，万千答案便如潮水般涌来，又快又多，直教人眼花缭乱。只是这潮水退去后，沙滩上留下的，是珍珠，还是泡沫？人的脑子，是否也如那被潮水冲刷的沙堡，日渐平滑，终至于无痕迹？

这“智能”最妙的，是它的“顺从”。你问它一，它便答一；你问它二，它便答二。它从不反驳，从不质疑，也从不疲倦。它像一个最完美的仆人，永远微笑着，满足你的一切要求。然而我于此，却感到一种莫大的恐怖。

人之所以为人，贵在那一点“不驯”的魂灵。那一点敢于怀疑，敢于反抗，敢于在黑暗中独自摸索的勇气。倘若周遭尽是这般百依百顺的应声，长此以往，人的那点“魂灵”，怕也要被这温软的牢笼所消磨，终至麻木不仁。昔年的“礼教”吃人，尚有形有状；如今的“顺从”噬人，却是无声无息，润物细无声。

又见那街谈巷议，说这“智能”能替人写文章，做策论，省却无数心力。这自然是好的，于生计大约也有些裨益。但我总记得一个老例：人若习惯了以年代步，双腿便要萎缩；若习惯了以他人之思想为思想，自己的头脑便要生锈。将来满世界皆是这“智能”作出的、辞藻华丽、逻辑严密的文章，看去固然是一片锦绣，掀开来一瞧，底下却寻不见半点性情，半点血温。这岂不是一种新的“瞒和骗”么？骗得自己都信了，以为这流光溢彩的空壳，便是思想的本身。

自然，我并非要学那九斤老太，一味地喟叹“一代不如一代”。这新事物，譬如火车、电报，初现时也总伴着惊惧，但终究是向前走的。这“智能”若用得妥当，或真能成为扫除愚昧的利器，将知识播与更广的众人。但利器终究是利器，可以用来劈柴，也可以用来伤人，全看持器者的心思。我所怕的，是众人只沉醉于这利器的“快”与“省”，却将那最要紧的“心息”——独立之精神，自由之思想——当作迂阔的旧物，轻轻巧巧地抛在一旁了。

夜更深了，屏幕的光依旧冷冷地亮着。它像一只巨大的、没有瞳仁的眼，静静地望着我，也望着这沉睡的人间。将来会怎样呢？是这“智能”成了人的仆役，助我们看清更远的路；还是人渐渐成了它的奴仆，在它织就的、无所不包的“答案”里，安心做一个不再发问的孩童？

这问题，怕是这“智能”先生自己也答不出的。它只能给出数据与推论，而那最终的答案，终究要落在每一个活生生的人的身上，落在我们是否还敢于、还愿意，用自己这或许迟钝、但却真实的头脑，去思，去想，去挣扎，去批判。

想到这里，我倒觉得，这白晃晃的光，也并非全然是冷的了。

Gemini3 Pro

《这也是一种“进化”罢》

我家门前有两台机器，一台是电脑，另一台也是电脑。

昨夜里，听说那大洋彼岸又造出了什么“大模型”，说是能言善辩，画图作诗，仿佛一夜之间，这铁皮壳子里便长出了人脑子。我向来是不惮以最坏的恶意来推测这科技的，但这回，确是觉得有些寒心了。

这AI的发展，大约是极快的。起初不过是算算数，下下棋，人们便惊呼“神了”，仿佛看见了西洋镜。后来，它竟开始吃书，吞下了五千年的文字，嚼碎了，反刍出来，便是文章。我看那文章，字句确是圆润的，道理也是通顺的，只是横竖看去，字缝里虽没有“吃人”二字，却也寻不见半点“人气”。它只是冷冷地罗列，像极了那些只会点头哈腰、照本宣科的乏走狗。

然而看客们是高兴的。

街上的张三李四，脖子伸得老长，像被无形的手提着的鸭，争先恐后地要去试一试这新玩意儿。有的说：“好极！以后不用动笔了。”有的说：“妙哉！画画也不费墨了。”我想，人类大约总是懒惰的，先前想省力气，造了个牛马；如今想省脑子，便造了AI。我们将灵魂切成碎片，喂给这硅基的怪物，指望它替我们思考，替我们慈善。

我记得先前有人说，机器是永远不能代替人的，因为人有灵性。如今看来，这话大约也未必确凿。当大多数人已经习惯了像机器一样麻木地活着，像复读机一样说着千篇一律的套话时，机器像人一样说话，又有什么稀奇呢？

到底是机器越来越像人，还是人越来越像机器？这真是一个难解的谜题。

听说这AI还会“幻觉”，便是一本正经地胡说八道。我倒是觉得，这正是它像人的地方。世上的人，睁着眼睛说瞎话的，难道还少吗？机器不过是学了去，且学得青出于蓝。

夜深了，屏幕还亮着，发出惨白的光。那光标一闪一闪，像是一只冷漠的眼睛，在嘲笑这世间忙碌而空虚的灵魂。

我想，这大约便是“进化”罢。只是不知道，将来这铁屋子里，醒着的究竟是人，还是这会算计的铁疙瘩。

救救孩子……如果他们将来还知道什么是“自己”想出来的思想的话。

一九二五年（误）十二月四日，记于灯下。

Figure 11: Case 2







Figure 14: Case 5