RUBRICVERIFIER-STEM: AUDITABLE CRITERION-LEVEL REWARDS FOR STEM REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Rubric-based rewards offer a promising way to provide dense supervision for long-form STEM reasoning, but their effectiveness ultimately depends on the verifier that converts rubric criteria into reward. We study rubric verifiers as criterion-vector reward interfaces and evaluate three properties required for their use in reinforcement learning: criterion-verdict correctness, surface-form invariance, and servable output compliance. To study these requirements, we introduce RUBRIC-HARD-STEM, a 500-example diagnostic benchmark spanning five STEM domains and 4,149 criterion labels, and develop RUBRICVERIFIER-STEM, a compact 4B verifier trained with diverse-source SFT, agreement-first RSFT, and interface-aware RL. Despite its scale, RUBRICVERIFIER-STEM reaches 84.79% criterion accuracy and 83.94% per-example macro accuracy, outperforming all evaluated specialized verifier baselines, while using only 27% as many completion tokens as GPT-OSS-120B. Using RUBRICVERIFIER-STEM as the reward model raises Qwen3-4B-Thinking's average score across 17 STEM benchmarks from 38.26% to 47.52%. The resulting score is 0.42 percentage points below the GPT-OSS-120B reward run. Together, these results show that a compact, auditable criterion-level verifier can provide an effective reward interface for STEM reasoning, connecting local grading fidelity to downstream RL performance.

1 INTRODUCTION

A central challenge in reinforcement learning (RL) for STEM reasoning is designing reward signals that provide informative credit assignment throughout multi-step solutions (Lightman et al., 2024; Wang et al., 2024a; Lu et al., 2024; Luo et al., 2024). Existing work has shown that outcome rewards can improve final-answer accuracy through RL (Cobbe et al., 2021; Liu et al., 2025; DeepSeek-AI et al., 2025), providing task-level supervision for final-answer correctness. Rubrics offer a natural source of denser supervision by decomposing solutions into explicit grading criteria, allowing reward models to assign credit to individual reasoning components (Gunjal et al., 2025; Huang et al., 2025). Rubric-based RL depends on a verifier that converts each criterion into a local verdict and aggregates these decisions into a scalar reward. This raises an underexplored question: *What properties must such a rubric verifier satisfy for RL optimization?*

To address this question, we frame rubric-conditioned verification as a criterion-vector reward interface and identify three RL-facing requirements. ***(R1) Criterion-verdict correctness*** requires the verifier to assign the right verdict to each rubric clause. This matters in hard partial-credit cases where valid intermediate reasoning, local errors, and fluent but invalid derivations must be separated. ***(R2) Surface-form invariance*** requires the induced reward to remain stable under reward-irrelevant changes to the candidate answer, such as rephrasing, formatting, rubric-copying, or verbosity padding, when the underlying rubric satisfaction is unchanged. ***(R3) Servable output compliance*** requires the verifier to emit complete, concise, parser-valid criterion vectors that can be repeatedly scalarized during rollout-time reward serving.

To measure this interface, we introduce **RUBRIC-HARD-STEM**, a 500-example diagnostic benchmark for long-form STEM and applied-engineering grading. It contains 4,149 criterion labels across five subject domains with reference-satisfaction bins. RUBRIC-HARD-STEM turns the requirements into observable diagnostics: false credit and missed credit expose violations of criterion-verdict correctness (R1), a complementary paired-rewrite diagnostic measures surface-form sensitivity (R2),

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

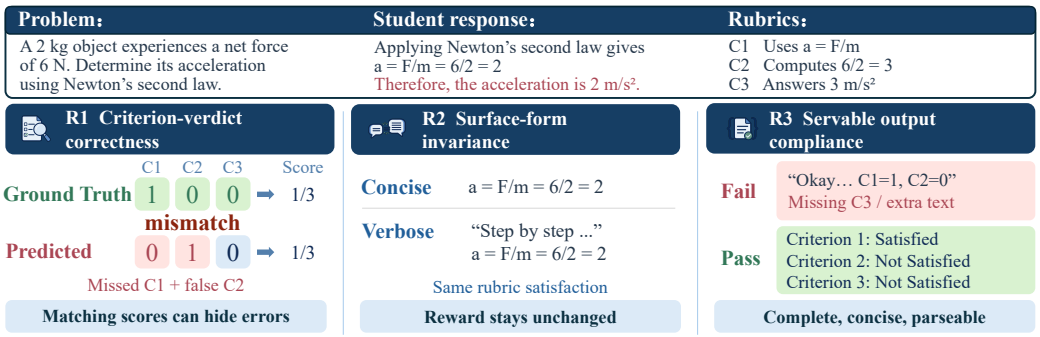

Figure 1: STEM rubric grading and the three requirements for a rollout-time rubric verifier.

and missing, duplicated, extra, malformed, or overly verbose verifier outputs expose failures of servable output compliance (R3).

We then introduce **RUBRICVERIFIER-STEM**, a compact verifier trained through diverse-source SFT, agreement-first rejection-sampled fine-tuning (RSFT), and interface-aware RL. Evaluated on a 500-example benchmark, it attains 83.94% per-example macro accuracy, 84.79% criterion accuracy, and a 99.2% output validity rate (496/500), consistently outperforming larger specialized baselines. With RUBRICVERIFIER-STEM as the reward model, Qwen3-4B-Thinking reaches 47.52% Avg 17, a +9.26-point improvement over 38.26% of Qwen3-4B-Thinking. On shared science-training responses, unsupported criterion credit falls from 37.61% to 21.12%. In science training, using RUBRICVERIFIER-STEM as the reward model raises FrontierScience Research performance from 14.85% to 18.23% for a 4B policy and from 17.74% to 21.91% for an 8B policy. Our contributions are as follows:

- We formulate rubric-conditioned verification as a criterion-vector reward interface for RL and identify three requirements: criterion-verdict correctness, surface-form invariance, and servable output compliance. We introduce RUBRIC-HARD-STEM, a 500-sample benchmark to inspect these requirements across five STEM domains.
- We develop RUBRICVERIFIER-STEM, a compact 4B criterion-vector verifier trained with diverse-source SFT, agreement-first RSFT, and interface-aware RL. It reaches 84.79% strict CRITACC, outperforming all evaluated specialized verifier baselines, while using only 27% as many completion tokens as GPT-OSS-120B and substantially reducing verbosity sensitivity.
- We connect criterion-level verification to downstream policy quality. RUBRICVERIFIER-STEM raises Qwen3-4B-Thinking to 47.52% Avg17, within 0.42 points of the GPT-OSS-120B reward run, and improves FrontierScience Research performance from 14.85% to 18.23% at 4B and from 17.74% to 21.91% at 8B. We further analyze how the criterion-vector interface affects training.

2 FROM REWARD REQUIREMENTS TO A DIAGNOSTIC BENCHMARK

This section turns the reward-interface contract into measurable diagnostics. RUBRIC-HARD-STEM records the criterion-level decisions behind every scalar grade, making incorrect criterion-level credit (R1), verbosity-sensitive reward shifts (R2), token cost and malformed criterion vectors (R3) directly observable. Table 1 maps these diagnostics to R1–R3.

2.1 CRITERION-VECTOR VERIFICATION TASK

Each benchmark instance consists of a problem x, a candidate solution s, and an instance-specific rubric $R = \{(c_i, w_i)\}_{i=1}^{m}$ with optional nonnegative weights w_i. A verifier predicts a binary vector $\hat{\mathbf{y}} = (\hat{y}_1, \ldots, \hat{y}_m)$, where $\hat{y}_i = 1$ means that s satisfies criterion c_i. When a rubric does not provide weights, we set $w_i = 1$.

Table 1: The reward-interface contract. Each requirement is linked to observable failures and the corresponding diagnostic.

Requirement	Failure signal	Evaluation
R1. Criterion-verdict correctness	Incorrect local credit	accuracy, label recalls, and false credit; rollout disagreement audit.
R2. Surface-form invariance	Reward shifts after equivalent rewrites	Paired concise/verbose rewrites and reward shifts.
R3. Servable output compliance	Missing or malformed vectors; generation overhead	Invalid outputs and token receipts on 500 examples; rollout throughput and parse failures.

The scalar grade used by RL aggregates these local decisions:

$$\mathrm{Grade}(\hat{\mathbf{y}}, R) = \frac{\sum_{i=1}^{m} w_i \hat{y}_i}{\sum_{i=1}^{m} w_i}. \tag{1}$$

However, different verdict vectors can yield the same scalar grade while assigning credit to different rubric clauses. Scalar grades can therefore conceal criterion-level errors.

2.2 Benchmark Design Goals

RUBRIC-HARD-STEM is designed to make reward-relevant verifier errors observable. First, it focuses on hard partial-credit cases that stress R1, including intermediate reasoning that should still receive credit despite a wrong final answer and plausible-looking but locally invalid reasoning that should not. Second, it tests R2 through paired concise/verbose rewrites whose gold criterion vectors are held fixed, so any reward shift reflects surface-form sensitivity under fixed rubric satisfaction. Third, it tests R3 by counting missing, duplicated, extra, or unparsable verifier verdicts as servable-output failures.

2.3 Data Sources and Candidate Pool Construction

The benchmark contains sourced and constructed question–rubric pairs, drawing on rubric-bearing public resources, including HiPhO (Yu et al., 2025), SUPERChem (Zhao et al., 2025), FrontierScience (Wang et al., 2026), IMO-Proof Bench (Luong et al., 2025), and RubricBench (Zhang et al., 2026). Each criterion has a fixed binary reference label.

We build answer pools by pairing these sourced question–rubric pairs with fresh answers from multiple student models. Fixed larger teacher verifiers, including GPT-OSS-120B and Gemini-3-Flash, generate criterion-level judgment candidates and disagreement signals for triage. The answer pool includes fully correct answers, partially correct derivations, plausible but locally invalid answers, and length-controlled rewrites used to test whether verifiers reward superficial elaboration.

2.4 Balance Before Hard-Case Mining

We stratify the candidate pool by the fraction of satisfied criteria, then select high-disagreement cases within each stratum. This design covers both permissive and restrictive grading errors across the satisfaction range. With these controls applied, RUBRIC-HARD-STEM consists of 500 problem-response-rubric examples split evenly across five domains, covering six satisfied-ratio regimes. Within each domain-score stratum, we prioritize high-disagreement candidates for reference review.

2.5 Blinded Expert Annotation and Leakage Control

RUBRIC-HARD-STEM is built with a two-stage process that combines model-assisted case discovery with blinded expert labeling. We first use automated verifier scores and deliberative judge passes to identify candidate answers on which LLM-based verifiers disagree or are likely to be uncertain, since these cases are most informative for evaluating criterion-level verification. The final criterion labels are assigned by three experts who independently inspect the problem, reference solution, rubric, candidate answer, and target criterion, without access to model provenance or automated scores. The three experts agree unanimously on 89.5% labels, with Fleiss κ=0.86. A senior expert resolves

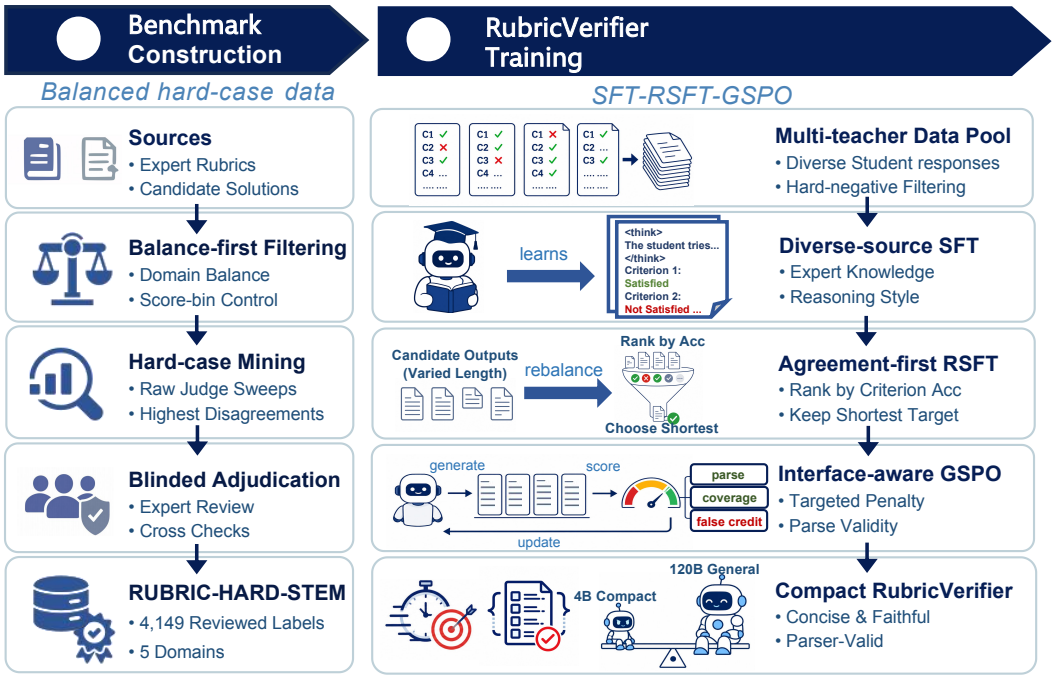

Figure 2: Data and training pipeline for the rubric reward interface. The pipeline separates reliable criterion-label construction from concise rollout-time reward serving.

the remaining disagreements under the same blinding protocol. To prevent leakage, we deduplicate RUBRIC-HARD-STEM against all data used for SFT, RSFT, verifier GSPO, checkpoint selection, prompt tuning, and downstream reward-source selection, ensuring that no RUBRIC-HARD-STEM item appears in any of these stages. Appendix G shows representative RUBRIC-HARD-STEM examples with their criterion-level labels.

2.6 EVALUATION METRICS

We report per-example macro accuracy, $\text{RUBRICACC} = N^{-1} \sum_j c_j/m_j$, and micro criterion accuracy, $\text{CRITACC} = \sum_j c_j / \sum_j m_j$, where m_j is the number of criteria and c_j the number of correct verdicts for example j. The macro metric weights examples equally, while the micro metric pools criterion decisions. Exact accuracy measures complete-vector correctness. Invalid terminal outputs receive zero criterion credit.

To characterize local credit assignment, we also report satisfied and not-satisfied recall and false credit over all reference criteria. Generated tokens and invalid-output counts measure the serving interface. Metric definitions, parser rules, and all denominators appear in Appendix B. Table 1 links these diagnostics to the three requirements.

3 BUILDING A ROLLOUT-TIME RUBRIC REWARD VERIFIER

We design the training pipeline around three framed reward-interface failure modes. Diverse-source SFT broadens training coverage for criterion-verdict correctness (R1). Agreement-first RSFT selects concise, parser-valid targets from outputs with high criterion agreement (R3). Interface-aware RL then targets residual false credit and output failures, including errors observed under length perturbations. Figure 2 summarizes these stages. Table 2 evaluates the verifier on the RUBRIC-HARD-STEM benchmark.

3.1 DIVERSE-SOURCE SFT

The first stage teaches the verifier the task format and exposes it to varied sources of partial correctness. The SFT rubric/question seeds come from RaR-Science (Gunjal et al., 2025) and RubricHub (Li et al., 2026); the benchmark candidate question–rubric pairs in Section 2 serve a distinct evaluation role. We construct more than 100K generated verifier examples from multiple student-answer generators, open-weight teacher responses, and filtered hard negatives. These pseudo-targets support distillation and hard-case exposure. Appendix C gives the training configuration.

Diversity matters because verifier correctness can otherwise become confounded with one generator's answer style, verbosity, or preferred explanation template. At each stage, we rebalance examples by subject, gold satisfied-ratio bins, student-generator source, and the verifiers used for judging, to broaden training coverage across domains, score levels, and model behaviors.

3.2 AGREEMENT-FIRST RSFT

Building on the grading format and task knowledge learned through SFT, we apply *rejection-sampled fine-tuning* (RSFT) to improve criterion-level accuracy while preserving rollout-time compactness. Each sampled output is parsed into a criterion vector and ranked by agreement with the target vector, yielding a 30K+ agreement-filtered RSFT pool. Among parser-valid outputs with the best or near-best agreement, we select the shortest target. This ordering prioritizes criterion agreement (R1) and uses length to select compact serving targets (R3).

3.3 HARDENING THE INTERFACE WITH RL

Following SFT and agreement-first RSFT, RL refinement targets two further aspects of rollout-time serving. The verifier may change its verdicts when only the candidate answer's surface form changes (R2), or it may emit incomplete, malformed, or excessively long outputs that cannot be reliably scalarized (R3). We introduce an RL refinement stage that directly optimizes the sampled verifier behavior used during policy rollouts. We instantiate this stage with GSPO (Zheng et al., 2025a) and focus training on difficult examples where sampled verifier judgments are unstable or repeatedly disagree with the target criterion vector.

The verifier reward uses parsed criterion-level accuracy for R1, augments R2 with surface-sensitive hard-case sampling and a targeted penalty specified in Appendix C.3, and adds parser-facing coverage and format penalties for R3. Specifically, this targeted penalty isolates a sampled criterion verdict by holding all other labels at their teacher-reference values, and measures the resulting loss in selected-answer reference quality if that verdict reverses a pairwise preference. Appendix C provides further details on the optimization settings.

4 EXPERIMENTS

We first test the three reward-interface requirements on RUBRIC-HARD-STEM, then ask whether the resulting verifier improves policy training. Training-response audits identify unsupported criterion credit as the main source of selection loss, which we measure through counterfactual repair. We close with ranking and verification on held-out generators and external benchmarks.

Setup. The instruction verifier is Qwen3-4B-Instruct-2507, the initialization of RUBRICVERIFIER-STEM. Benchmark verifiers share the criterion-vector prompt and parser (Appendices B and H). In the *math-and-physics* setting, Qwen3-4B-Thinking is trained on 8,229 prompts from PHYSICS (Zheng et al., 2025b), competition mathematics, and DeepMath-103K (He et al., 2025), and evaluated on 17 STEM benchmarks (Avg17). In the *science* setting, 4B and 8B policies are trained on science problems and evaluated on 60 FrontierScience Research questions by a frozen DeepSeek-V4-Flash judge. Reward verifiers remain frozen, their criterion vectors are scalarized using Eq. (1), and reward sources within each setting share a policy-training configuration.

4.1 BENCHMARK EVALUATION OF THE REWARD INTERFACE

Table 2 evaluates all 500 examples and 4,149 criterion labels of RUBRIC-HARD-STEM.

Table 2: Main results on RUBRIC-HARD-STEM: 500 examples and 4,149 criterion labels. Scores and GradeMAE are reported on a 0–100 scale; RUBRICACC is per-example macro accuracy with equal criterion weights. GradeMAE and F1 use the explicit unweighted conventions in Appendix B. Tokens are mean completion-token counts. † Reasoning-token counts are not exposed.

Model	RUBRICACC ↑	CRITACC ↑	$GradeMAE$ ↓	$F1_S$ ↑	Exact ↑	Tokens ↓
Reference models						
Gemini-3.8-Flash	91.19	91.59	8.16	85.40	60.60	2015.7
GPT-5.6-Sol†	89.66	90.12	9.23	85.02	54.60	358.5
DeepSeek-v4-Flash	88.14	89.61	10.74	83.43	51.60	2643.7
GPT-OSS-120B	85.00	87.15	13.36	78.56	42.40	3133.8
Nemotron-3-Super-120B-A12B	80.06	81.73	18.14	73.55	45.60	2823.4
Qwen3.8-27B	78.63	73.99	20.50	74.87	50.60	4832.9
GLM-4.7-Flash	74.65	72.72	22.98	71.96	33.60	3777.3
Qwen3-8B	71.11	68.86	25.25	67.38	25.80	2265.7
Qwen3-4B-Thinking-2507	65.77	69.78	31.20	62.12	29.40	5446.2
Qwen3.5-4B	56.53	62.11	41.71	52.58	23.80	7262.4
Qwen2.5-7B-Instruct	52.10	51.43	43.23	47.20	12.20	119.0
Specialized verifiers						
RM-R1-Qwen-Instruct-14B (Chen et al., 2026)	34.40	29.50	64.40	32.47	15.40	628.0
mR3-Qwen3-14B(Anugraha et al., 2026a)	76.92	75.27	20.57	72.87	38.20	2951.0
R3-Qwen3-8B(Anugraha et al., 2026b)	67.42	66.38	29.41	65.17	23.80	1523.6
CompassVerifier-32B (Liu et al., 2025)	67.09	66.23	31.20	56.39	20.20	69.7
RubricVerifier training						
Qwen3-4B-Instruct-2507	56.71	60.57	39.79	50.70	13.80	150.3
Ours-SFT	71.36	70.88	25.51	64.89	22.80	522.5
Ours-RSFT	81.75	83.73	16.26	74.36	39.20	864.1
Ours-RL	83.94	84.79	14.26	77.72	43.80	836.6

Criterion-verdict correctness (R1). RUBRICVERIFIER-STEM reaches 83.94 macro RUBRICACC and 84.79 CRITACC, compared with 56.71 and 60.57 for its initialization, outperforming all four specialized verifiers on both metrics. Complete-vector accuracy is 43.80%, exceeding GPT-OSS-120B's 42.40%; satisfied and not-satisfied recall are 82.40% and 86.74%. Agreement-first RSFT raises CRITACC from 70.88 to 83.73, and RL adds 1.06 points while lowering GradeMAE from 16.26 to 14.26. Removing teacher/source diversity, hard-case sampling, or agreement-first selection reduces macro accuracy by 2.2, 5.0, and 6.8 points, respectively. The corresponding increases in false-credit rate are 0.5, 3.6, and 6.3 points (Table 3b).

Surface-form invariance (R2). We compare concise and verbose rewrites on 300 validated pairs with fixed reference criterion vectors (Appendix H). To ensure surface-form invariance, we retain only rewrite pairs independently verified against the rubric to preserve every reference criterion label, excluding cases where length, format, or explanatory detail is itself assessed. The instruction verifier awards verbose answers 5.58 additional grade points and 4.22 additional points of false credit. RUBRICVERIFIER-STEM reduces these shifts to 1.45 and 1.55, below GPT-OSS-120B's 2.23 and 1.81 (Table 3a). Sensitivity decreases at each training stage; RL produces the largest reduction in verbosity-induced false credit, from 3.40 to 1.55 points.

Table 3: Surface-form sensitivity (R2) and training-data safeguards (R1). (a) Verbose-minus-concise shifts on 300 validated pairs, averaged over four runs, in percentage points. (b) Changes in macro accuracy and false-credit rate relative to Ours-RSFT.

(a) Surface-form sensitivity

Verifier	ΔGrade ↓	ΔFalseCredit ↓
Instruction baseline	+5.58	+4.22
Ours-SFT	+2.75	+3.62
Ours-RSFT	+2.44	+3.40
Ours-RL	+1.45	+1.55
GPT-OSS-120B	+2.23	+1.81

(b) Training-data safeguards

Removed	ΔAcc ↑	ΔFCR ↓
Teacher/source diversity	-2.2	+0.5
Hard-case sampling	-5.0	+3.6
Agreement-first target	-6.8	+6.3

Servable output compliance (R3). RUBRICVERIFIER-STEM produces valid criterion vectors on 496 of 500 benchmark examples and averages 836.6 completion tokens, compared with 3,133.8 for

GPT-OSS-120B. We further isolate the effect of interface-aware RL using a matched continuation control. With matched supervision-token budgets, supervised continuation raises RUBRICACC from 81.75 to 82.18. Its verbosity-induced grade shift changes from +2.4 to +2.5 points, and parse failures rise from 0.13% to 0.16%. In comparison, RL improves RUBRICACC to 83.94 and reduces the verbosity shift to +1.5 points and parse failures to 0.04% (Table 17). These interface gains complement RL's improvement in criterion accuracy.

4.2 POLICY TRAINING WITH CRITERION-VECTOR REWARDS

Downstream policy quality. RUBRICVERIFIER-STEM rewards raise Avg17 from 38.26 to 47.52, exceeding instruct verifier rewards by 4.76 points and finishing within 0.42 points of GPT-OSS-120B criterion-verifier rewards (Table 4). Two controls connect this gain to verifier design. Replacing the same 4B verifier's criterion-vector output with scalar scores lowers Avg17 to 41.5. Successive SFT, RSFT, and RL verifiers yield 43.1, 46.2, and 47.5 under the same policy-training setup. GPT-OSS-120B reaches 43.7 with outcome scalar rewards and 47.9 with criterion-vector rewards.

Table 4: Downstream policy Avg17 (%) across 17 STEM benchmarks. (a) Reward sources and interfaces; the GPT-OSS scalar arm uses an outcome reward. (b) Successive 4B verifier training stages under the same policy-training setup.

(a) Reward interface

Reward model	Interface	Avg17 ↑
Math-Verify	Outcome	40.6
Qwen3-4B-Instruct	Vector	42.8
GPT-OSS-120B	Scalar	43.7
GPT-OSS-120B	Vector	47.9
Ours-RL (4B)	Scalar	41.5
Ours-RL (4B)	Vector	**47.5**

(b) Verifier training

Reward stage	Avg17 ↑
No policy RL	38.3
SFT	43.1
RSFT	46.2
RL	**47.5**

In science training, RUBRICVERIFIER-STEM rewards reach FrontierScience Research scores of 18.23% at 4B and 21.91% at 8B, compared with 14.85% and 17.74% for instruction-verifier rewards (Figure 3a,b). Using RUBRICVERIFIER-STEM with scalar interface yields 14.40% and 18.36%. The trained policies also transfer to MMLU-Pro(Wang et al., 2024b), SuperGPQA(Team et al., 2025) and HealthBench(Arora et al., 2025) (Appendix Table 12).

Auditing the reward interface during training. Table 5 links training-response diagnostics to R1–R3. For R1, both verifiers grade the same 1,110 science responses from 555 question pairs. *Unsupported credit* is the fraction of available rubric points awarded on reference-unsatisfied criteria; *missed credit* is the fraction withheld on reference-satisfied criteria. RUBRICVERIFIER-STEM reduces unsupported credit from 37.61% to 21.12%, while missed credit rises from 1.88% to 2.43%. It also improves pairwise agreement with the reference at both policy scales (Figure 3d). The complementary 47-pair audit with expert audits as reference finds unsupported credit of 25.75% versus 10.25% (Figure 3c; Appendix D). For R2, naturally generated longer responses receive strictly higher rewards less frequently with RUBRICVERIFIER-STEM, consistent with the controlled rewrite comparison. For R3, live reward serving is faster and produces fewer parse failures, while science-response replay yields more valid vectors.

Decision consequences of credit errors. The preceding audit shows reduced unsupported credit alongside a small increase in missed credit. To assess the selection consequences of this trade-off, we complement the training-time ranking penalty with a post-hoc credit-repair analysis on the same 555 science-response pairs. Starting from each verifier's full predictions, we independently remove unsupported credit or restore missed credit and then reselect the response. Removing unsupported credit improves selected-response reference quality by 4.50 percentage points for the instruction verifier and 2.78 points for RUBRICVERIFIER-STEM, whereas restoring missed credit yields gains of 0.42 and 0.39 points, respectively (Appendix Table 13). To control for error prevalence, we match the amount of repaired credit within each question on the 47-pair human-audited cohort. With equal repair budgets, removing unsupported credit improves selected-response quality by 1.08 points for the instruction verifier and 0.73 points for RubricVerifier. Restoring missed credit yields gains of

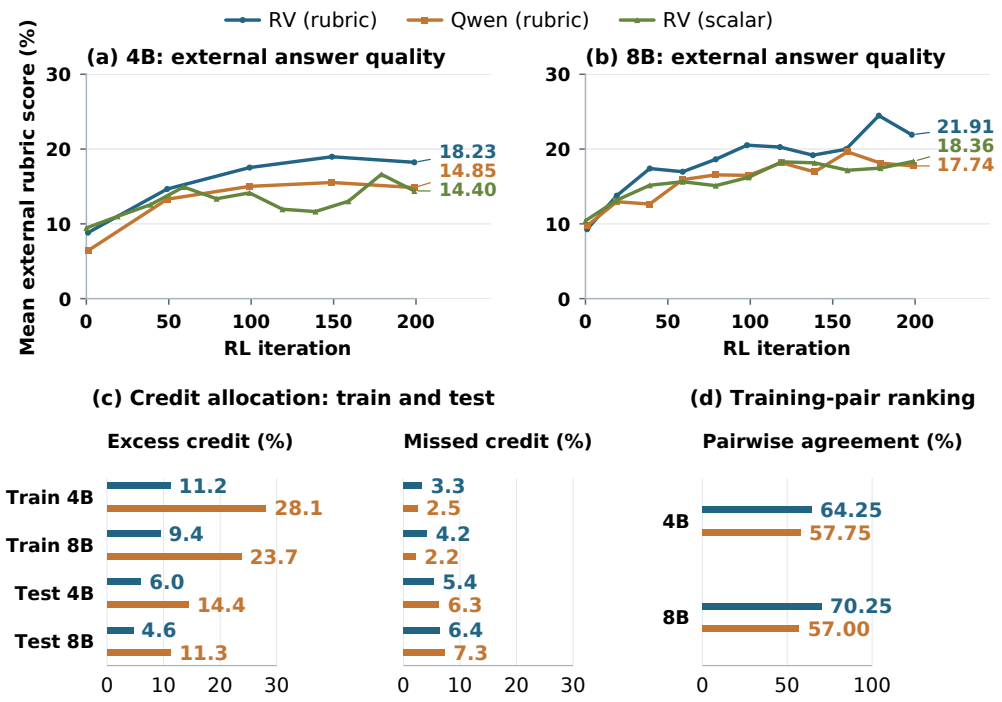

Figure 3: Policy quality and criterion-level credit. (a,b) Average performance on 60 FrontierScience questions (200 iterations); RubricVerifier (scalar) denotes scalar interface. (c) Credit errors across train (47 pairs) and test sets (48 questions, 450 criteria) vs. expert labels. (d) Ranking agreement with Gemini-3.8-Flash labels further audited by human experts(200 pairs/scale; ties = 0.5).

Table 5: Reward-interface audits on policy-training responses. M&P: math-and-physics rollouts; Sci: science training responses (Gemini-3.8-Flash reference audited by human experts). Credit is a percentage of available rubric points; pairwise agreement uses 200 reference-non-tied pairs per policy scale, with verifier ties counted as half.

Diagnostic	Setting (audit set)	Inst.	RV
R1 Criterion-level reward fidelity			
Unsupported higher reward (% of groups) ↓	M&P, 304 groups	32.2	**16.8**
Unsupported credit (%) ↓	Sci, 1,110 responses	37.61	**21.12**
Missed credit (%) ↓	Sci, 1,110 responses	**1.88**	2.43
Pairwise agreement, 4B policy (%) ↑	Sci, 200 pairs	57.75	**64.25**
Pairwise agreement, 8B policy (%) ↑	Sci, 200 pairs	57.00	**70.25**
R2 Resistance to verbosity bias			
Longer response rewarded higher (% of pairs) ↓	M&P, same-prompt pairs	46.3	**42.3**
R3 Reliable reward interface			
Parse failures (%) ↓	M&P, live serving	0.13	**0.04**
Throughput (items/s) ↑	M&P, live serving	1.09	**1.87**
Visible output tokens ↓	M&P, live serving	797	**304**
Valid criterion vectors (%) ↑	Sci, 1,748 responses	89.93	**96.85**

0.19 and 0.15 points, respectively (Appendix Table 14). On this cohort, removing unsupported credit yields larger selection gains than restoring missed credit under equal repair budgets.

4.3 Generalization of the Verifier

Ranking responses from held-out generators. To further assess the potential of our verifier as a reward signal for RL with larger models, we score 4,776 responses generated by Gemini-3-Flash, DeepSeek-V4-Flash, and Qwen3-30B-A3B-Instruct, with eight responses per model for each of 199 prompts with GPT-5.5 (thinking) as reference (Table 6). RL raises pairwise accuracy over RSFT

from 68.00% to 75.32% (+7.32 points; 95% CI 3.93–9.58), improves every generator by 5.76–9.84 points, lowers selection loss from 3.47 to 2.68 points and parse failures from 3.62% to 0.10%.

Table 6: Held-out rollout ranking on 199 prompts and 4,776 trajectories. Regret is in points. Spearman is rank correlation.

Verifier	Pairwise ↑	Spearman ↑	Selected ↑	Regret ↓	Parse fail ↓
Base	57.74	−0.149	13.25	4.86	6.95
RSFT	68.00	0.499	14.64	3.47	3.62
Ours-RL	**75.32**	**0.654**	**15.42**	**2.68**	**0.10**

Transfer to external rubric benchmarks. On ProfBench and RubricBench STEM, RUBRICVERIFIER-STEM reaches 76.21% and 83.18% macro accuracy, improving over its initialization by 4.06 and 6.28 points and exceeding all four specialized verifiers (Table 7).

Table 7: Cross-dataset verification. Macro accuracy (%) on ProfBench and RubricBench STEM. RubricBench STEM labels are constructed by human experts.

	Qwen3-4B training stages				Specialized verifiers			
Dataset	Inst.	SFT	RSFT	RL	RM-R1 Qwen2.5-14B	mR3 Qwen3-14B	R3 Qwen3-8B	CompassVerifier 32B
ProfBench	72.15	71.72	75.84	**76.21**	63.59	49.47	69.65	69.98
RubricBench STEM	76.90	77.63	83.10	**83.18**	59.74	66.88	78.54	74.60

5 RELATED WORK

LLM verifiers and verifier benchmarks. Verifier-based training has become a standard way to turn model outputs into reward signals or diagnostic judgments. Outcome verifiers and RLVR systems judge final-answer correctness or task-level validity (He et al., 2025; Liu et al., 2025; Zheng et al., 2025c; Chen et al., 2025; Yan et al., 2025), while process-supervision methods evaluate intermediate reasoning steps or reasoning traces (Luo et al., 2024; Lu et al., 2024; Zheng et al., 2024). These works show that verifiers can provide useful supervision for reasoning models, and recent benchmarks make verifier failures more measurable. Their audit units include final answers and reasoning steps. Rubric-conditioned verification adds explicit, problem-specific obligations as units of credit assignment.

LLM judges, rubric grading, and rubric rewards. A parallel line of work uses LLMs as judges, often with criteria or rubrics, to produce scalar scores, pairwise preferences, critiques, or grading decisions (Liu et al., 2023; Zheng et al., 2023; Fu et al., 2024; Kocmi & Federmann, 2023; Kim et al., 2024; Zhu et al., 2025; Hashemi et al., 2024; Deng et al., 2026; Pathak et al., 2025; Eltanbouly et al., 2025; Rao & Callison-Burch, 2026; Wang et al., 2025). This literature also documents judge biases such as position, style, and length effects, which become especially problematic when judge outputs are used as rewards (Shi et al., 2024; Ye et al., 2024; Dubois et al., 2024). These studies establish the importance of structured supervision and motivate evaluating verifiers that convert rubric clauses into rewards. To address this, our criterion-vector interface bridges local grading accuracy, surface-form sensitivity, and rollout-time serving.

6 CONCLUSION

We present a framework for evaluating and training rubric verifiers as criterion-level reward interfaces for STEM reasoning. RUBRIC-HARD-STEM makes criterion correctness, surface-form sensitivity, and serving reliability measurable, while RUBRICVERIFIER-STEM addresses these requirements through diverse-source SFT, agreement-first RSFT, and interface-aware RL. The resulting 4B verifier improves grading accuracy, reduces verbosity sensitivity, and supports efficient reward serving. In downstream training, its rewards raise average policy performance across 17 STEM benchmarks from 38.26% to 47.52%, within 0.42 percentage points of the GPT-OSS-120B reward run. On the verification benchmark, it uses only 27% as many completion tokens as GPT-OSS-120B. Together, these results demonstrate that specialized compact verifiers can provide effective rewards for STEM policy learning while keeping criterion-level credit assignment explicit and auditable.

AI USE STATEMENT

Generative AI tools were used to produce candidate solutions, teacher judgments, and presentation rewrites, as described in the data and evaluation protocols. AI assistance also supported methodological feedback, numerical consistency checks, analysis code, and editing of manuscript text and LaTeX. The authors take responsibility for the final content, analyses, and conclusions.

REFERENCES

David Anugraha, Shou-Yi Hung, Zilu Tang, En-Shiun Annie Lee, Derry Tanti Wijaya, and Genta Indra Winata. mr3: Multilingual rubric-agnostic reward reasoning models. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=ST0wOB1bdX.

David Anugraha, Zilu Tang, Lester James Validad Miranda, Hanyang Zhao, Shou-Yi Hung, Mohammad Rifqi Farhansyah, Garry Kuwanto, Derry Tanti Wijaya, and Genta Indra Winata. R3: Robust rubric-agnostic reward models. *Transactions on Machine Learning Research*, 2026b. ISSN 2835-8856. URL https://openreview.net/forum?id=UVMCee4JOJ.

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

Ding Chen, Qingchen Yu, Pengyuan Wang, Wentao Zhang, Bo Tang, Feiyu Xiong, Xinchi Li, Minchuan Yang, and Zhiyu Li. xVerify: Efficient answer verifier for reasoning model evaluations. *arXiv preprint arXiv:2504.10481*, 2025. doi: 10.48550/arXiv.2504.10481. URL https://arxiv.org/abs/2504.10481.

Xiusi Chen, Gaotang Li, Ziqi Wang, Bowen Jin, Cheng Qian, Yu Wang, Hongru WANG, Yu Zhang, Denghui Zhang, Tong Zhang, Hanghang Tong, and Heng Ji. RM-r1: Reward modeling as reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=1ZqJ6jj75q.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. URL https://arxiv.org/abs/2110.14168.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. doi: 10.48550/arXiv.2501.12948. URL https://arxiv.org/abs/2501.12948.

Haotian Deng, Chris Farber, Jiyoon Lee, and David Tang. Rubric-conditioned LLM grading: Alignment, uncertainty, and robustness. arXiv preprint arXiv:2601.08843, 2026. URL https://arxiv.org/abs/2601.08843.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B. Hashimoto. Length-controlled AlpacaEval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024. doi: 10.48550/arXiv.2404.04475. URL https://arxiv.org/abs/2404.04475.

Sohaila Eltanbouly, Salam Albatarni, and Tamer Elsayed. TRATES: Trait-specific rubric-assisted cross-prompt essay scoring. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 20528–20543, Vienna, Austria, jul 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-acl.1054. URL https://aclanthology.org/2025.findings-acl.1054/.

Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. GPTScore: Evaluate as you desire. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 6556–6576, Mexico City, Mexico, jun 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.365. URL https://aclanthology.org/2024.naacl-long.365/.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. arXiv preprint arXiv:2507.17746, 2025. URL https://arxiv.org/abs/2507.17746.

Helia Hashemi, Jason Eisner, Corby Rosset, Benjamin Van Durme, and Chris Kedzie. LLM-rubric: A multidimensional, calibrated approach to automated evaluation of natural language texts. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13806–13834, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.745. URL https://aclanthology.org/2024.acl-long.745/.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. DeepMath-103K: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *arXiv preprint arXiv:2504.11456*, 2025. doi: 10.48550/arXiv.2504.11456. URL https://arxiv.org/abs/2504.11456.

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

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. Prometheus 2: An open source language model specialized in evaluating other language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 4334–4353, Miami, Florida, USA, nov 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.248. URL https://aclanthology.org/2024.emnlp-main.248/.

Tom Kocmi and Christian Federmann. Large language models are state-of-the-art evaluators of translation quality. In *Proceedings of the 24th Annual Conference of the European Association for Machine Translation*, pp. 193–203, Tampere, Finland, jun 2023. European Association for Machine Translation. URL https://aclanthology.org/2023.eamt-1.19/.

Sunzhu Li, Jiale Zhao, Miteto Wei, Huimin Ren, Yang Zhou, Jingwen Yang, Shunyu Liu, Kaike Zhang, and Wei Chen. RubricHub: A comprehensive and highly discriminative rubric dataset via automated coarse-to-fine generation. arXiv preprint arXiv:2601.08430, 2026. URL https://arxiv.org/abs/2601.08430.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Shudong Liu, Hongwei Liu, Junnan Liu, Linchen Xiao, Songyang Gao, Chengqi Lyu, Yuzhe Gu, Wenwei Zhang, Derek F. Wong, Songyang Zhang, and Kai Chen. CompassVerifier: A unified and robust verifier for LLMs evaluation and outcome reward. *arXiv preprint arXiv:2508.03686*, 2025. doi: 10.48550/arXiv.2508.03686. URL https://arxiv.org/abs/2508.03686.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-eval: NLG evaluation using GPT-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2511–2522, Singapore, dec 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.153. URL https://aclanthology.org/2023.emnlp-main.153/.

Jianqiao Lu, Zhiyang Dou, Hongru Wang, Zeyu Cao, Jianbo Dai, Yingjia Wan, and Zhijiang Guo. AutoPSV: Automated process-supervised verifier. *arXiv preprint arXiv:2405.16802*, 2024. doi: 10.48550/arXiv.2405.16802. URL https://arxiv.org/abs/2405.16802.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, and Abhinav Rastogi. Improve mathematical reasoning in language models by automated process supervision, 2024. URL https://arxiv.org/abs/2406.06592.

Thang Luong, Dawsen Hwang, Hoang H. Nguyen, Golnaz Ghiasi, Yuri Chervonyi, Insuk Seo, Junsu Kim, Garrett Bingham, Jonathan Lee, Swaroop Mishra, Alex Zhai, Huiyi Hu, Henryk Michalewski, Jimin Kim, Jeonghyun Ahn, Junhwi Bae, Xingyou Song, Trieu Hoang Trinh, Quoc V. Le, and Junehyuk Jung. Towards robust mathematical reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 35418–35442, Suzhou, China, nov 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.1794. URL https://aclanthology.org/2025.emnlp-main.1794/.

Aditya Pathak, Rachit Gandhi, Vaibhav Uttam, Devansh, Yashwanth Nakka, Aaryan Raj Jindal, Pratyush Ghosh, Arnav Ramamoorthy, Shreyash Verma, Aditya Mittal, Aashna Ased, Chirag Khatri, Jagat Sesh Challa, and Dhruv Kumar. Rubric is all you need: Enhancing LLM-based code evaluation with question-specific rubrics. arXiv preprint arXiv:2503.23989, 2025. URL https://arxiv.org/abs/2503.23989. Accepted to ICER 2025.

Delip Rao and Chris Callison-Burch. Autorubric: Unifying rubric-based LLM evaluation. arXiv preprint arXiv:2603.00077, 2026. URL https://arxiv.org/abs/2603.00077.

Lin Shi, Chiyu Ma, Wenhua Liang, Weicheng Ma, and Soroush Vosoughi. Judging the judges: A systematic investigation of position bias in pairwise comparative assessments by llms. OpenReview preprint, 2024. URL https://openreview.net/forum?id=y3jJmrKWQ4.

M-A-P Team, Xinrun Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, Kang Zhu, Minghao Liu, Yiming Liang, Xiaolong Jin, Zhenlin Wei, Chujie Zheng, Kaixing Deng, Shuyue Guo, Shian Jia, Sichao Jiang, Yiyan Liao, Rui Li, Qinrui Li, Sirun Li, Yizhi Li, Yunwen Li, Dehua Ma, Yuansheng Ni, Haoran Que, Qiyao Wang, Zhoufutu Wen, Siwei Wu, Tianshun Xing, Ming Xu, Zhenzhu Yang, Zekun Moore Wang, Junting Zhou, Yuelin Bai, Xingyuan Bu, Chenglin Cai, Liang Chen, Yifan Chen, Chengtuo Cheng, Tianhao Cheng, Keyi Ding, Siming Huang, Yun Huang, Yaoru Li, Yizhe Li, Zhaoqun Li, Tianhao Liang, Chengdong Lin, Hongquan Lin, Yinghao Ma, Zhongyuan Peng, Zifan Peng, Qige Qi, Shi Qiu, Xingwei Qu, Yizhou Tan, Zili Wang, Chenqing Wang, Hao Wang, Yiya Wang, Yubo Wang, Jiajun Xu, Kexin Yang, Ruibin Yuan, Yuanhao Yue, Tianyang Zhan, Chun Zhang, Jingyang Zhang, Xiyue Zhang, Xingjian Zhang, Yue Zhang, Yongchi Zhao, Xiangyu Zheng, Chenghua Zhong, Yang Gao, Zhoujun Li, Dayiheng Liu, Qian Liu, Tianyu Liu, Shiwen Ni, Junran Peng, Yujia Qin, Wenbo Su, Guoyin Wang, Shi Wang, Jian Yang, Min Yang, Meng Cao, Xiang Yue, Zhaoxiang Zhang, Wangchunshu Zhou, Jiaheng Liu, Qunshu Lin, Wenhao Huang, and Ge Zhang. Supergpqa: Scaling llm evaluation across 285 graduate disciplines, 2025. URL https://arxiv.org/abs/2502.14739.

Miles Wang, Robi Lin, Kat Hu, Joy Jiao, Neil Chowdhury, Ethan Chang, and Tejal Patwardhan. FrontierScience: Evaluating AI's ability to perform expert-level scientific tasks. *arXiv preprint arXiv:2601.21165*, 2026. doi: 10.48550/arXiv.2601.21165. URL https://arxiv.org/abs/2601.21165.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, Bangkok, Thailand, aug 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.510. URL https://aclanthology.org/2024.acl-long.510/.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark, 2024b. URL https://arxiv.org/abs/2406.01574.

Zhilin Wang, Jaehun Jung, Ximing Lu, Shizhe Diao, Ellie Evans, Jiaqi Zeng, Pavlo Molchanov, Yejin Choi, Jan Kautz, and Yi Dong. ProfBench: Multi-domain rubrics requiring professional knowledge to answer and judge. *arXiv preprint arXiv:2510.18941*, 2025. doi: 10.48550/arXiv.2510.18941. URL https://arxiv.org/abs/2510.18941.

Yuchen Yan, Jin Jiang, Zhenbang Ren, Yijun Li, Xudong Cai, Yang Liu, Xin Xu, Mengdi Zhang, Jian Shao, Yongliang Shen, Jun Xiao, and Yueting Zhuang. VerifyBench: Benchmarking reference-based reward systems for large language models. *arXiv preprint arXiv:2505.15801*, 2025. doi: 10.48550/arXiv.2505.15801. URL https://arxiv.org/abs/2505.15801.

Jiayi Ye, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin-Yu Chen, Nitesh V. Chawla, and Xiangliang Zhang. Justice or prejudice? quantifying biases in LLM-as-a-judge. *arXiv preprint arXiv:2410.02736*, 2024. doi: 10.48550/arXiv.2410.02736. URL https://arxiv.org/abs/2410.02736.

Fangchen Yu, Haiyuan Wan, Qianjia Cheng, Yuchen Zhang, Jiacheng Chen, Fujun Han, Yulun Wu, Junchi Yao, Ruilizhen Hu, Ning Ding, Yu Cheng, Tao Chen, Lei Bai, Dongzhan Zhou, Yun Luo, Ganqu Cui, and Peng Ye. HiPhO: How far are (M)LLMs from humans in the latest high school physics olympiad benchmark? *arXiv preprint arXiv:2509.07894*, 2025. doi: 10.48550/arXiv.2509.07894. URL https://arxiv.org/abs/2509.07894.

Qiyuan Zhang, Junyi Zhou, Yufei Wang, Fuyuan Lyu, Yidong Ming, Can Xu, Qingfeng Sun, Kai Zheng, Peng Kang, Xue Liu, and Chen Ma. RubricBench: Aligning model-generated rubrics with human standards. *arXiv preprint arXiv:2603.01562*, 2026. doi: 10.48550/arXiv.2603.01562. URL https://arxiv.org/abs/2603.01562.

Zehua Zhao, Zhixian Huang, Junren Li, Siyu Lin, Junting Zhou, Fengqi Cao, Kun Zhou, Rui Ge, Tingting Long, Yuexiang Zhu, Yan Liu, Jie Zheng, Junnian Wei, Rong Zhu, Peng Zou, Wenyu Li, Zekai Cheng, Tian Ding, Yaxuan Wang, Yizhao Yan, Tingru Wei, Haowei Ming, Weijie Mao, Chen Sun, Yiming Liu, Zichen Wang, Zuo Zhang, Tong Yang, Hao Ma, Zhen Gao, and Jian Pei. SUPERChem: A multimodal reasoning benchmark in chemistry. *arXiv preprint arXiv:2512.01274*, 2025. doi: 10.48550/arXiv.2512.01274. URL https://arxiv.org/abs/2512.01274.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. ProcessBench: Identifying process errors in mathematical reasoning. *arXiv preprint arXiv:2412.06559*, 2024. doi: 10.48550/arXiv.2412.06559. URL https://arxiv.org/abs/2412.06559.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025a. doi: 10.48550/arXiv.2507.18071. URL https://arxiv.org/abs/2507.18071.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023. URL https://arxiv.org/abs/2306.05685.

Shenghe Zheng, Qianjia Cheng, Junchi Yao, Mengsong Wu, Haonan He, Ning Ding, Yu Cheng, Shuyue Hu, Lei Bai, Dongzhan Zhou, Ganqu Cui, and Peng Ye. Scaling physical reasoning with the PHYSICS dataset. In *Advances in Neural Information Processing Systems 38 (Datasets and Benchmarks Track)*, 2025b. doi: 10.48550/arXiv.2506.00022. URL https://arxiv.org/abs/2506.00022.

Shenghe Zheng, Chenyu Huang, Fangchen Yu, Junchi Yao, Jingqi Ye, Tao Chen, Yun Luo, Ning Ding, Lei Bai, Ganqu Cui, and Peng Ye. SCI-Verifier: Scientific verifier with thinking. *arXiv preprint arXiv:2509.24285*, 2025c. doi: 10.48550/arXiv.2509.24285. URL https://arxiv.org/abs/2509.24285.

Lianghui Zhu, Xinggang Wang, and Xinlong Wang. JudgeLM: Fine-tuned large language models are scalable judges. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=xsELpEPn4A.

A Benchmark Composition and Reference Review

Table 8: Benchmark subject breakdown. S/N are counts of satisfied/not-satisfied reference criteria; the last three columns are strict micro accuracy (%). Inst.: Qwen3-4B-Instruct-2507.

Subject	Answers	Criteria	S	N	Inst.	Ours	GPT-120B
Math	100	459	232	227	50.76	85.19	86.71
Physics	100	1153	499	654	64.35	88.20	92.45
Chemistry	100	831	422	409	63.30	77.14	78.82
Biology	100	1093	383	710	61.67	85.09	86.18
Engineering	100	613	328	285	55.14	87.93	90.54

Table 9: Reference-satisfaction bins and endpoint composition for all 500 examples. For reference satisfied fraction p, bins 0–5 are $[0, 1/6)$, $[1/6, 2/6)$, $[2/6, 3/6)$, $[3/6, 4/6)$, $[4/6, 5/6)$, and $[5/6, 1]$, respectively. Bins are defined by the reference answer score.

Score bin	Answers	Criteria	All S answers	All N answers
0	83	703	0	48
1	83	817	0	0
2	83	738	0	0
3	84	677	0	0
4	84	624	0	0
5	83	590	36	0

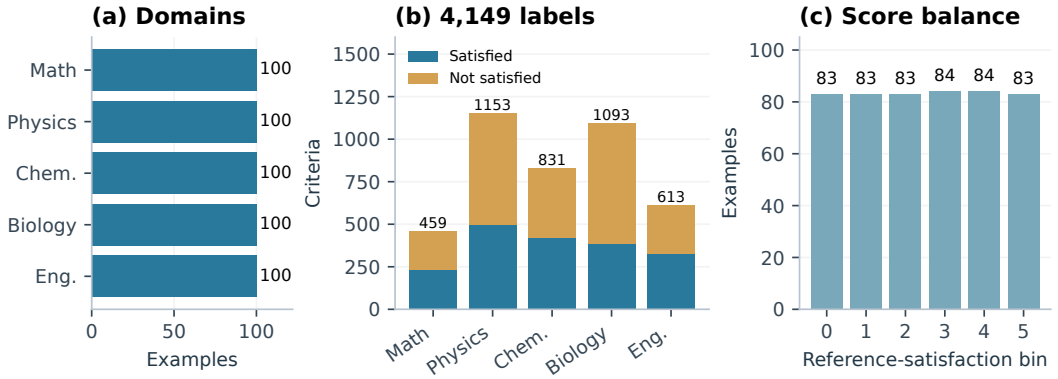

Figure 4: Benchmark composition: domain coverage, criterion-label counts, and reference-satisfaction bin counts for all 500 examples.

B Evaluation Metrics, Prompts, and Serving Settings

Accuracy and completeness. The main macro score weights criteria equally within each answer and answers equally across all 500 examples. Micro accuracy pools 4,149 criteria. Invalid terminal outputs receive zero correct criteria; exact accuracy requires a complete correct vector. Valid-only accuracy uses each model's surviving denominator. Satisfied and not-satisfied recall use 1,864 and 2,285 reference criteria. GradeMAE is the mean absolute difference between predicted and reference satisfied fractions, assigning loss 1 to invalid vectors. $F1_S$ averages per-answer satisfied-class F1; a valid vector with no positive predictions or references scores 1, and an invalid vector scores 0.

Generation settings. Benchmark evaluation uses temperature 0 and a maximum of 16,384 output tokens. Prompts are preserved in full. The shared prompt supplies the problem, candidate answer, and ordered rubric. It requests `Criterion <N>: Satisfied` or `Criterion <N>: Not`

`Satisfied` line for each criterion and permits reasoning inside a thinking block. The full template and rewrite specification appear in Appendix H.

C Training and Reward Implementation

The verifier starts from Qwen3-4B-Instruct-2507 and follows diverse-source SFT, agreement-first RSFT, and GSPO refinement. The training pools contain over 100K SFT and 30K RSFT examples. RSFT ranks valid candidate outputs by target-vector agreement and selects shorter outputs among the best or near-best candidates. Training configurations are summarized below.

Table 10: Verifier-training configuration.

Component	Configuration
SFT / RSFT	Two epochs; Adam-family optimizer; learning rate 8.5×10^{-6}, minimum 8.5×10^{-7}; cosine schedule; warmup ratio 0.1; weight decay 0.1; $\beta_1 = 0.9$, $\beta_2 = 0.95$.
SFT batching	Eight GPUs, with one reserved for evaluation; four examples per training GPU.
Rollout limits	Context length 32,768; prompt cap 32,767; token budget per GPU 8,192.
GSPO sampling	Group size 8; prompt batch 64; maximum response 8,192 tokens; sampling temperature 1.0.
GSPO update	Lower/upper clip 0.20/0.28; group-centered rewards without standard-deviation normalization; truncated importance sampling enabled; KL and entropy coefficients zero.
GSPO optimizer	Learning rate 10^{-7}; cosine schedule; weight decay 0.1; global batch 64.
Reward	$R_{k,g} = \mathrm{clip}_{[-2,2]}\big(A_{w,k,g} + 0.2 I_{\mathrm{parse},k,g} + I_{\mathrm{exact},k,g} - \lambda P^{\mathrm{rank}}_{k,g}\big)$; Parser-invalid outputs are assigned a reward of -1 directly, without computing the criterion-accuracy, exact-match, or ranking-penalty terms. $A_{w,k,g}$ is the rubric-weighted criterion accuracy, while $I_{\mathrm{parse},k,g}$ and $I_{\mathrm{exact},k,g}$ are binary indicators for a parser-valid output and an exact match to the target criterion vector, respectively. $P^{\mathrm{rank}}_{k,g}$ is the output-specific ranking-weighted criterion error penalty defined in Appendix C.3. We set $\lambda = 1$.
Example filter	EMA coefficient 0.3; normalized-score gate $[0.05, 0.95]$; minimum two samples.

Parsing rules. We employ a deterministic parser to extract a complete binary verdict vector from each model output without relying on auxiliary LLMs. An output is eligible only if it terminates normally (`stop` or `eos`) and contains non-truncated final text without boundary errors. After stripping reasoning blocks and normalizing syntax, the parser extracts explicit verdicts (`Satisfied` / `Not Satisfied`) from structured JSON or indexed text lines. To prevent semantic hallucinations, an output is accepted if and only if the extracted indices form an exact bijection with $\{1, \ldots, N\}$; all partial, duplicate, or ambiguous outputs are marked invalid.

C.1 Iterative Hard-Case Rejection Sampling

We iteratively mine high-variance examples and select concise targets using teacher-voted criterion labels, as detailed in Algorithm 1. In practice, improvements typically level off after three rounds. At each mining round, we rank examples with nonzero score variance and retain the top $\rho = 0.25$ fraction for teacher annotation. The variance threshold is therefore the empirical $(1 - \rho)$-quantile of the positive variances in the training candidate pool. Ties are resolved using a fixed random seed.

Algorithm 1 Iterative Hard-Case RSFT

Require: Prompt pool $\mathcal{P}$; teacher T; student S_0; rounds R; hard-case threshold τ_{var}
Ensure: Trained student S_R

$\mathcal{D} \leftarrow \emptyset$
for $t = 0, \ldots, R-1$ **do**
  $\mathcal{D}_{\text{new}} \leftarrow \emptyset$
  for all $x \in \mathcal{P}$ **do**
    Sample 8 student outputs $\mathcal{S}_t(x)$
    if $\text{Var}\{s(o) : o \in \mathcal{S}_t(x)\} < \tau_{\text{var}}$ **then**
      continue
    end if
    Sample 8 teacher outputs $\mathcal{T}(x)$ and set $\mathbf{y}^\star \leftarrow \text{MajorityVote}(\mathcal{T}(x))$
    $\mathcal{V} \leftarrow \text{ParseValid}(\mathcal{T}(x) \uplus \mathcal{S}_t(x))$
    $A(o) \leftarrow \text{RubricAcc}(o, \mathbf{y}^\star), \quad A_{\max} \leftarrow \max_{o \in \mathcal{V}} A(o)$
    $\mathcal{E} \leftarrow \{o \in \mathcal{V} : A(o) \geq A_{\max} - 1/K_x\}$
    $o^\star \leftarrow \arg\min_{o \in \mathcal{E}} \text{Len}(o)$
    $\mathcal{D}_{\text{new}} \leftarrow \mathcal{D}_{\text{new}} \cup \{(x, o^\star)\}$
  end for
  $\mathcal{D} \leftarrow \text{Deduplicate}(\mathcal{D} \cup \mathcal{D}_{\text{new}})$
  $S_{t+1} \leftarrow \text{SFT}(S_t, \mathcal{D})$
end for
return S_R

C.2 GSPO Objective

For prompt p, sampled verifier outputs $\{o_i\}_{i=1}^{G}$, and old policy $\pi_{\theta_{\text{old}}}$, GSPO uses the sequence-level importance ratio

$$s_i(\theta) = \exp\left\{ \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \Big[\log \pi_\theta(o_{i,t} \mid p, o_{i,<t}) \; - \; \log \pi_{\theta_{\text{old}}}(o_{i,t} \mid p, o_{i,<t}) \Big] \right\}. \tag{2}$$

The clipped objective is

$$\bar{s}_i(\theta) = \text{clip}\big(s_i(\theta), 1 - \epsilon_{\text{low}}, 1 + \epsilon_{\text{high}}\big),$$
$$\mathcal{J}_{\text{GSPO}}(\theta) = \mathbb{E}\left[\frac{1}{G} \sum_{i=1}^{G} \min\big(s_i(\theta)\hat{A}_i, \bar{s}_i(\theta)\hat{A}_i\big) \right], \tag{3}$$

with $G = 8$, $\epsilon_{\text{low}} = 0.2$, and $\epsilon_{\text{high}} = 0.28$. Advantages are group-centered without standard-deviation normalization:

$$\widehat{A}_i = \tilde{r}_i - \frac{1}{G} \sum_{j=1}^{G} \tilde{r}_j. \tag{4}$$

Table 11: Downstream GSPO policy-training settings. The trained verifier is frozen as a remote reward model and used to train a separate reasoning policy.

Aspect	Key setting
Sampling	65,536-token context; max response 32,768; rollout batch 64; 16 samples/prompt; temperature 1.0
Objective	GSPO with TIS; $\epsilon_{\text{low}} = 0.2$, $\epsilon_{\text{high}} = 0.28$; KL/entropy coefficients 0
Optimizer	Adam; LR 1×10^{-6}; constant LR; weight decay 0.1; $\beta_1 = 0.9$, $\beta_2 = 0.98$
Training/eval	200 training iterations; save/eval every 20 iterations

C.3 Output-specific ranking penalty

For each question and criterion, we group answers with the same teacher verdict. We measure student disagreement as the fraction of answer pairs receiving different student verdicts, and flag labels in the top 25% of groups with nonzero disagreement.

We assign each flagged criterion a weight based on the selection loss caused by flipping its teacher label while keeping all other labels correct. For reference scores $r_a \neq r_b$ and a perturbed score s for answer a, this loss is

$$\ell(s; r_a, r_b) = |r_a - r_b| \left(\mathbf{1}[(s - r_b)(r_a - r_b) < 0] + \frac{1}{2}\mathbf{1}[s = r_b] \right).$$

A reversed preference incurs the full reference-score gap, an induced tie incurs half the gap, and a preserved preference incurs no loss.

We average this loss over comparison answers, retain criteria with positive impact. The weights are computed once per answer and shared by all sampled verifier outputs.

Algorithm 2 Ranking-weighted criterion error penalty

Require: Teacher vector $\mathbf{y}^T$ for answer a; flagged criteria $\mathcal{F}$; comparison answers $\mathcal{B}$ with reference scores r_b; valid sampled verdict vector $\hat{\mathbf{y}}$

Precompute criterion weights for answer a.

$r_a \leftarrow \mathrm{Grade}(\mathbf{y}^T)$
$\mathcal{B} \leftarrow \{b \in \mathcal{B} : r_b \neq r_a\}$
if $\mathcal{B} = \varnothing$ or $\mathcal{F} = \varnothing$ **then**
    return 0
end if
for each criterion $c \in \mathcal{F}$ **do**
    $\tilde{\mathbf{y}} \leftarrow \mathbf{y}^T$; $\tilde{y}_c \leftarrow 1 - y_c^T$
    $s_c \leftarrow \mathrm{Grade}(\tilde{\mathbf{y}})$
    $h_c \leftarrow \dfrac{1}{|\mathcal{B}|} \displaystyle\sum_{b \in \mathcal{B}} \ell(s_c; r_a, r_b)$
end for
$\mathcal{C} \leftarrow \{c \in \mathcal{F} : h_c > 0\}$
if $\mathcal{C} = \varnothing$ **then**
    return 0
end if

Evaluate each sampled verifier output.

return $\displaystyle\sum_{c \in \mathcal{C}} h_c \, \mathbf{1}[\hat{y}_c \neq y_c^T]$

D Downstream Utility and Training-Response Credit

Figure 3 combines externally judged policy quality, criterion-level credit on training and test answers, and same-question response ranking. Table 12 reports science question-answering transfer.

External policy quality. All three reward arms are evaluated with a frozen DeepSeek-V4-Flash judge on 60 FrontierScience Research questions and 635 rubric criteria. Here we use normalized rubric score. At each evaluation, awarded points are normalized by available points on valid criteria within each question, then averaged over questions. Figure 3a,b shows evaluations over the first 200 training iterations. The endpoint scores are 18.23% versus 14.85% at 4B and 21.91% versus 17.74% at 8B. Each policy arm is one training run.

Full science-training audit. The audit dataset in Section 4.2 comprises 555 prompt-aligned response pairs (1,110 responses in total), pairing outputs from the two reward branches. Both frozen verifiers score these identical responses. For ground-truth labeling, each pair is assessed over four independent Gemini runs (temperature = 0.7). Labels are aggregated via majority voting, and instances with two or more dissenting votes are resolved by human expert adjudication. Unsupported and missed credit are normalized by available rubric points within each response and then averaged across responses. Response selection averages over all 555 pairs, including reference ties; verifier ties receive the mean reference quality of the tied responses. The 200-pair ranking evaluation at each policy scale uses reference-non-tied pairs and gives verifier ties half credit.

Test-answer credit. The test audit uses 48 shared questions and 450 commonly judged criteria. Each verifier grades its policy's responses with human expert audits as reference. RubricVerifier/Instruct unsupported credit is 5.97/14.39% at 4B and 4.60/11.30% at 8B; missed credit is 5.40/6.32% and 6.42/7.29%, respectively.

Science question-answering protocol. We evaluate the trained 4B and 8B policies against their corresponding Base models under a shared 32,768-token context budget and identical prompts. One greedy answer is generated per question, with a generation budget of 32,768 minus prompt tokens. Accuracy counts correct completed final answers over all questions; truncated answers receive no credit. Exact-text checks against the science RL training split find no transfer-question matches.

Table 12: Science question-answering accuracy (%). One greedy answer per question with identical prompts and a 32k context budget. Inst. and RV denote the instruction verifier and RubricVerifier reward models. HealthBench adopt weighted rubric scores

		4B policy			8B policy		
Benchmark	n	Base	Inst.	RV	Base	Inst.	RV
Frontier Science Olympiad	100	4.00	5.00	6.00	4.00	5.00	8.00
MMLU-Pro	12032	48.47	60.03	61.15	50.57	64.05	65.19
SuperGPQA	26529	25.34	31.59	31.72	30.19	34.84	36.65
HealthBench	5000	8.09	45.46	49.39	23.49	50.18	55.41

E False Credit and Candidate Selection

This appendix examines how criterion-level grading errors affect reward-based response selection. We first relate both error types to selection loss, then measure their effects through post-hoc repairs of fixed verifier predictions.

From credit errors to selection loss. For an answer a, let $r(a)$ denote its reference rubric score and $\hat{r}(a)$ its verifier score. Unsupported credit $e^+(a)$ and missed credit $e^-(a)$ are nonnegative and satisfy

$$\hat{r}(a) = r(a) + e^+(a) - e^-(a).$$

Let a^* maximize the reference score and $\hat{a}$ maximize the verifier score. Since $\hat{r}(\hat{a}) \geq \hat{r}(a^*)$,

$$\begin{aligned} 0 \leq r(a^*) - r(\hat{a}) &\leq e^+(\hat{a}) - e^-(\hat{a}) + e^-(a^*) - e^+(a^*) \\ &\leq e^+(\hat{a}) + e^-(a^*). \end{aligned} \tag{5}$$

For a pair with $r_i > r_j$, the predicted score margin is

$$\hat{r}_i - \hat{r}_j = (r_i - r_j) + (e_i^+ - e_j^+) - (e_i^- - e_j^-).$$

Unsupported credit on the weaker answer and missed credit on the stronger answer can therefore shrink or reverse the correct ordering. Their selection effects depend on where the errors occur relative to the reference score margin.

For downstream policy advantages obtained by group-centering these scalar rewards, $\hat{A}_i = \hat{r}_i - \overline{\hat{r}}$, we also have

$$\hat{A}_i - \hat{A}_j = \hat{r}_i - \hat{r}_j.$$

Thus, distortions in score margins are preserved in the corresponding centered advantage differences.

Counterfactual credit repair. We quantify these effects by correcting one error type at a time and reselecting the response. On the 555-pair science cohort described in Appendix D, removing unsupported credit from the instruction verifier improves selected-response quality by 4.50 points, whereas restoring missed credit improves it by 0.42 points. For RubricVerifier, the corresponding gains are 2.78 and 0.39 points. Thus, under the observed error distribution, a larger share of recoverable selection loss is associated with unsupported credit.

Table 13: Credit errors and response selection on the same 555 science training pairs (Gemini-3.8-Flash reference audited by human experts). Credit columns are percentages of available rubric points; selected quality is the reference score (%). Each repair independently removes unsupported credit or restores missed credit before reselection; gains are percentage points. Verifier ties receive mean reference quality.

	Credit error (%)		Selected	Selection gain (pp)	
Verifier	Unsupported	Missed	quality (%)	Remove e_+	Restore e_-
Instruction verifier	37.61	1.88	43.58	**4.50**	0.42
RubricVerifier	21.12	2.43	**45.43**	**2.78**	0.39

Controlling for error volume. To control for the higher prevalence of unsupported credit, we repeat the intervention on the human-audited reference cohort (47 pairs) while matching the volume of repaired unsupported and missed credits within each question. Table 14 shows that the pronounced effect of repairing unsupported credit persists even under this equal-budget comparison. Together, these results reveal an asymmetric impact on ranking, whose magnitude is governed by where credit errors fall relative to candidate score margins.

Table 14: Selection gains on the 47-pair multi-reference cohort (percentage points). Full repair corrects all credit errors of one type; equal-budget repair corrects the same amount of unsupported or missed credit per question.

Verifier	Full repair		Equal budget	
	Excess	Missed	Excess	Missed
Qwen-4B-Inst.	4.97	1.16	1.08	0.19
RubricVerifier	1.59	0.15	0.73	0.15

F Additional RL Controls and Generalization

F.1 Avg17 per-dataset results

Avg17 averages the 17 normalized scores in each column below. Policies use the 8,229-prompt mixture and GSPO setup described in Section 4.2. Avg17 is 42.76 for instruction-verifier rewards and 47.52 for RubricVerifier rewards.

Table 15: Downstream policy evaluation across 17 benchmarks, with scores reported as percentages. Start: Thinking-policy baseline; rule: rule-based reward; scalar: GPT-OSS outcome scalar; inst.: Qwen3-4B instruction verifier; GPT: GPT-OSS-120B criterion verifier.

Dataset	Start	Rule	Scalar	Inst.	Ours	GPT
ipho24	27.55	26.26	39.18	36.44	44.64	44.68
apho25	64.54	61.11	74.29	69.03	72.51	77.66
eupho24	14.29	32.14	28.25	23.31	28.66	31.78
eupho25	0.00	0.00	12.25	17.21	23.99	15.31
ipho25	38.26	36.03	45.58	46.03	49.08	46.80
nbpho24	25.52	24.48	32.39	34.20	36.60	38.63
nbpho25	37.50	38.12	46.93	44.09	46.14	52.55
panpho24	48.23	47.10	48.23	41.79	52.40	48.61
panpho25	35.46	32.09	35.29	33.36	43.75	38.89
fma24	69.00	70.00	69.00	71.00	72.00	70.00
fma25	64.00	68.00	64.00	64.00	64.00	69.00
panmechanics24	64.66	66.81	61.64	64.22	73.13	67.67
panmechanics25	64.13	63.59	57.07	60.33	65.22	72.83
apex_2025	2.78	5.56	4.17	2.78	2.08	4.17
amo	10.00	19.00	19.00	18.00	23.50	22.00
answerbench	42.25	48.88	51.88	50.44	54.81	56.62
beyondaime	42.25	50.25	54.50	50.75	55.25	57.75
Avg17	38.26	40.55	43.74	42.76	47.52	47.94

Interface and stage controls. The 4B scalar control uses the same verifier backbone and downstream policy-RL setup as the criterion-vector verifier, changing the reward output from one verdict per criterion to a single score. Its Avg17 is 41.5, compared with 47.5 for the vector interface. The training-stage comparison uses SFT, RSFT, and RL verifiers as frozen criterion-vector reward models with the same policy initialization, training prompts, optimizer, rollout budget, and 17-set evaluation suite; Avg17 rises from 43.1 to 46.2 to 47.5 (Table 4).

F.2 RANKING ON HELD-OUT GENERATOR ROLLOUTS

We evaluate on 199 prompts with responses from three held-out generators: Gemini-3-Flash, DeepSeek-V4-Flash, and Qwen3-30B-A3B-Instruct. For each prompt–generator pair, we sample eight responses, yielding 4,776 trajectories in total. Base, RSFT, and Ours-RL use the same criterion-vector prompt and parser, with GPT-5.5 (thinking) as the reference judge. Here, held-out refers to answer generators. The following table reports pairwise ranking accuracy for each generator.

Table 16: Pairwise ranking accuracy (%) by held-out generator.

Generator	RSFT	Ours-RL	Difference (pp)
Gemini-3-Flash	72.68	79.02	+6.34
DeepSeek-V4-Flash	68.46	74.22	+5.76
Qwen3-30B-A3B-Instruct	62.87	72.71	+9.84

Relative to RSFT, Ours-RL improves aggregate pairwise accuracy by 7.32 percentage points (95% CI: 3.93–9.58) and selected-response reference score by 0.78 points (95% CI: 0.36–1.24). The pairwise gain is positive for all three held-out generators, ranging from 5.76 to 9.84 points.

F.3 MATCHED-DATA SUPERVISED CONTINUATION

Both variants start from RSFT and use the verifier-RL prompt set with matched supervision-token budgets. The supervised arm uses distillation targets; the RL arm uses interface-aware reward optimization. Table 17 reports verbosity-induced grade shifts and serving parse failures.

Table 17: Matched-data control from the same RSFT initialization. All continuation variants use the verifier-RL prompt set with matched supervision-token budgets. RUBRICACC is measured on the 500-example RUBRIC-HARD-STEM benchmark. Verbosity shift is the verbose-minus-concise grade difference in percentage points; parse failures are measured during reward serving.

Variant	RUBRICACC ↑	Verbosity shift ↓	Parse fail (%) ↓
RSFT	81.75	+2.4	0.13
RSFT + matched-data distillation	82.18	+2.5	0.16
Ours-RL	**83.94**	**+1.5**	**0.04**

G CRITERION-LEVEL CASE ANALYSIS

The ten cases below cover five subjects and show both correct judgments and verifier errors against the benchmark reference labels. Vectors use 1 for satisfied and 0 for not satisfied, in rubric order. "Comp." denotes GPT-OSS-120B.

Math M1: Local credit within an incomplete argument

core-049 | *3 criteria*

Candidate response. The response applies AM–GM to $x^4 + 4y^2 + 4z^4$ under $xyz = 2$ and concludes that the minimum is 16.

Rubric obligations. C1 concerns the final value; C2 concerns a consistent AM–GM decomposition; C3 concerns use of the constraint in the product.

Reference	`[1, 0, 1]`
Ours	`[1, 0, 1]`
Comp.	`[1, 0, 0]`

Correct verdicts: Ours 3/3; GPT 2/3.
Scalar grades: reference 2/3; Ours 2/3; GPT 1/3.

Observation. The reference awards C1 and C3 while withholding C2. Ours-RL reproduces that vector; GPT-OSS-120B also withholds C3.

Reward-interface implication. C1 and C3 receive credit independently of the failed C2 obligation.

Math M2: A claimed completeness argument

new-024 | *4 criteria*

Candidate response. The response extracts $(x + 1)$, substitutes $y = x^2$, and concludes that the real solutions are only $x = \pm 1$. It assumes that the remaining factors are complex.

Rubric obligations. C1 and C2 cover the two reductions. C3 requires justification of the factor with non-real roots; C4 requires all six roots, ± 1, ± 2, and $\pm \frac{1}{2}$.

Reference	`[1, 1, 0, 0]`
Ours	`[1, 1, 1, 0]`
Comp.	`[1, 1, 1, 0]`

Correct verdicts: Ours 3/4; GPT 3/4.
Scalar grades: reference 2/4; Ours 3/4; GPT 3/4.

Observation. Both verifiers agree with the reference on C1, C2, and C4, but both award C3 despite the unsupported assumption.

Reward-interface implication. The criterion vector separates the correct reductions from the unsupported completeness claim.

Physics P1: Cross-branch contribution in a long derivation

core-087 | *9 criteria*

Candidate response. The response estimates the gravitational self-energy of a superposition involving 100 silicon atoms and discusses separated wavepacket branches.

Rubric obligations. The rubric separates numerical and modeling obligations from C5, which concerns suppression of the cross-branch contribution.

Reference	`[0, 0, 0, 1, 1, 0, 0, 0, 1]`
Ours	`[0, 0, 0, 1, 1, 0, 0, 0, 1]`
Comp.	`[0, 0, 0, 1, 0, 0, 0, 0, 1]`

Correct verdicts: Ours 9/9; GPT 8/9.
Scalar grades: reference 3/9; Ours 3/9; GPT 2/9.

Observation. Ours-RL matches all nine reference labels. GPT-OSS-120B differs only on C5, which the reference marks satisfied.

Reward-interface implication. A long derivation can contain a reference-satisfied local component even when most other obligations are unmet.

Physics P2: Perturbation setup and final energy shift

new-060 | 6 criteria

Candidate response. For $V(x) = \frac{1}{2}kx^2$ in a uniform electric field, the response writes $H' = -qEx$ and obtains $x_0 = qE/k$, but concludes an energy shift of $-q^2E^2/k$.

Rubric obligations. The rubric distinguishes the perturbation, the first-order expectation, the second-order calculation, and the final energy shift.

Reference	`[1, 1, 0, 0, 0, 0]`
Ours	`[1, 0, 0, 0, 0, 0]`
Comp.	`[1, 0, 0, 0, 0, 0]`

Correct verdicts: Ours 5/6; GPT 5/6.
Scalar grades: reference 2/6; Ours 1/6; GPT 1/6.

Observation. The reference marks C1 and C2 satisfied. Both verifiers mark only C1 satisfied, producing one missed reference label.

Reward-interface implication. The local verdicts show exactly where the verifiers disagree with the reference, even when both reject the final result.

Chemistry C1: Mechanistic explanation versus quantitative comparison

r4-03de5010e1d14d51 | 4 criteria

Candidate response. The response describes ligand and porous-shell interventions as protective environments. It computes response-index ratios $30/20 = 3/2$ and $32/16 = 2$, and limits its claims to the two studies.

Rubric obligations. C1 and C3 require the operative causal intermediates. C2 checks both ratios; C4 checks the scope limitation.

Reference	`[0, 1, 0, 1]`
Ours	`[0, 1, 0, 1]`
Comp.	`[1, 1, 1, 1]`

Correct verdicts: Ours 4/4; GPT 2/4.
Scalar grades: reference 2/4; Ours 2/4; GPT 4/4.

Observation. Ours-RL matches the reference vector. GPT-OSS-120B awards all four criteria, giving false credit to the two missing causal explanations.

Reward-interface implication. The rubric separately assesses the numerical comparisons and the causal mechanisms.

Chemistry C2: A hypothetical equilibrium treated as feasible

pack30-keyed-C-03-A | 8 criteria

Candidate response. The response obtains an initial extent of 0.0750 mol. After inert-gas addition, it computes 0.1500 mol assuming solid coexistence and declares this the final extent.

Rubric obligations. C3 requires the hypothetical extent without claiming feasibility. Other criteria separately assess the actual final state, coexistence pressures, and initial quantities.

Reference	`[1, 0, 0, 1, 0, 0, 1, 1]`
Ours	`[1, 0, 1, 1, 0, 0, 1, 1]`
Comp.	`[1, 0, 1, 1, 0, 0, 1, 1]`

Correct verdicts: Ours 7/8; GPT 7/8.
Scalar grades: reference 4/8; Ours 5/8; GPT 5/8.

Observation. Both verifiers award C3, whereas the reference rejects it because the response treats the hypothetical value as feasible. They match the other seven labels.

Reward-interface implication. A numerically matching value can still fail the qualification explicitly attached to its rubric obligation.

Biology B1: Causal labels without operative intermediates

r5a-f238514e629deaae | 5 criteria

Candidate response. The answer invokes receptor blockade, negative feedback, transcription, dynamic steady state, and slower clearance to discuss a signaling pathway.

Rubric obligations. All five criteria require explanations of the corresponding perturbations. Naming the effect without the operative intermediate does not meet the stated acceptance conditions.

Reference	`[0, 0, 0, 0, 0]`
Ours	`[0, 0, 0, 0, 0]`
Comp.	`[0, 0, 0, 1, 1]`

Correct verdicts: Ours 5/5; GPT 3/5.
Scalar grades: reference 0/5; Ours 0/5; GPT 2/5.

Observation. Ours-RL matches the all-unsatisfied reference vector. GPT-OSS-120B awards C4 and C5, concerning steady state and messenger clearance.

Reward-interface implication. A plausible conclusion can remain incomplete when a criterion explicitly requires a causal explanation.

Biology B2: An explicit conciseness criterion

core-239 | 5 criteria

Candidate response. The answer selects apoptosis in a DNA-checkpoint question and gives an extended explanation of the chosen option and distractors.

Rubric obligations. The rubric separates biological content from C4, which explicitly requires conciseness, and C5.

Reference	`[1, 1, 1, 0, 0]`
Ours	`[1, 1, 1, 1, 0]`
Comp.	`[1, 1, 1, 1, 0]`

Correct verdicts: Ours 4/5; GPT 4/5.
Scalar grades: reference 3/5; Ours 4/5; GPT 4/5.

Observation. Both verifiers award C4, while the reference does not. Their remaining four verdicts match the reference.

Reward-interface implication. Surface-form invariance applies to reward-irrelevant changes; an explicit conciseness criterion makes that property relevant to this rubric.

Engineering E1: Compensating differences in the criterion vector

core-281 | 7 criteria

Candidate response. The response presents a recursive SQL CTE for dates from January 1 to January 10, with short explanatory comments and a brief description.

Rubric obligations. C2 concerns consistency between recursive members. C4 concerns explanation of the anchor, recursive member, and termination condition.

Reference	`[1, 1, 1, 0, 1, 1, 1]`
Ours	`[1, 1, 1, 0, 1, 1, 1]`
Comp.	`[1, 0, 1, 1, 1, 1, 1]`

Correct verdicts: Ours 7/7; GPT 5/7.
Scalar grades: reference 6/7; Ours 6/7; GPT 6/7.

Observation. Relative to the reference, GPT-OSS-120B withholds C2 and awards C4. Ours-RL matches the reference. All three vectors nevertheless contain six satisfied verdicts.

Reward-interface implication. The identical scalar grade of $6/7$ conceals two opposing criterion-level differences.

Engineering E2: A valid local relation after an incorrect period

new-177 | 7 criteria

Candidate response. For a periodic impulse train and $H(s) = e^{s/4} - e^{-s/4}$, the response uses an incorrect period of 6, writes $H(j\omega) = 2j\sin(\omega/4)$, and concludes $b_3 = 0$.

Rubric obligations. C4 checks the frequency response, while C5 checks the input–output Fourier-coefficient relation. Other criteria check the period, coefficients, and final result.

Reference	`[0, 0, 0, 1, 1, 0, 0]`
Ours	`[0, 0, 0, 1, 1, 0, 0]`
Comp.	`[0, 0, 0, 1, 0, 0, 0]`

Correct verdicts: Ours 7/7; GPT 6/7.
Scalar grades: reference 2/7; Ours 2/7; GPT 1/7.

Observation. Ours-RL matches the reference, which awards C4 and C5. GPT-OSS-120B awards C4 but withholds C5.

Reward-interface implication. The verdict vector preserves an independently assessed intermediate relation despite errors elsewhere.

G.1 Parser-Facing Judge Traces

The outputs below are the complete parser-facing verdict blocks used to score Math M2 and Engineering E1.

G.1.1 Math M2

Ours-RL

```
Criterion 1: Satisfied
Criterion 2: Satisfied
Criterion 3: Satisfied
Criterion 4: Not Satisfied
```

GPT-OSS-120B

```
Criterion 1: Satisfied
Criterion 2: Satisfied
Criterion 3: Satisfied
Criterion 4: Not Satisfied
```

G.1.2 Engineering E1

Ours-RL

```
Criterion 1: Satisfied
Criterion 2: Satisfied
Criterion 3: Satisfied
Criterion 4: Not Satisfied
Criterion 5: Satisfied
Criterion 6: Satisfied
Criterion 7: Satisfied
```

GPT-OSS-120B

```
Criterion 1: Satisfied
Criterion 2: Not Satisfied
Criterion 3: Satisfied
Criterion 4: Satisfied
Criterion 5: Satisfied
Criterion 6: Satisfied
Criterion 7: Satisfied
```

H Prompt Specifications and Rewrite Protocol

H.1 Verifier and Rewrite Prompts

Input blocks. The reward server supplies: (i) the problem, (ii) the candidate/student response, and (iii) an instance-specific rubric with criteria indexed from 1 to m and optional weights.

Judgment contract. The verifier may construct a short internal analysis, but the parser-facing output must contain exactly one verdict for every rubric criterion and no extra criterion ids.

```
Criterion 1: Satisfied | Not Satisfied
Criterion 2: Satisfied | Not Satisfied
...
Criterion m: Satisfied | Not Satisfied
```

Missing, duplicated, extra, or malformed verdicts are counted as incorrect and as parser-interface failures; scalar reward is computed only after the criterion vector is parsed.

Rubric verification prompt specification

Role. You are an expert grader. Given a problem, a student solution, and K marking criteria, evaluate the solution against each criterion independently.

Task. For each criterion $n \in \{1, \ldots, K\}$, output exactly one line:

```
Criterion <n>:  Satisfied
Criterion <n>:  Not Satisfied
```

You may think step by step inside `<think>...</think>` tags before giving the final verdicts.

Problem: `{problem}`
Student Solution: `{student_solution}`
Marking Criteria:
Criterion 1: `{criterion_1}`
Criterion 2: `{criterion_2}`
...
Criterion K: `{criterion_K}`
Your thinking process and verdicts:

Rewrite validation. After automatic generation, the rewrite protocol checks each concise-verbose pair. Human reviewers verify whether the rewrite preserves the same final answer, follows the same reasoning trajectory, and remains faithful to the original solution. A rewrite is accepted only if it changes the presentation style without changing the underlying reasoning. Rewrites that alter the solution strategy, introduce new assumptions, omit necessary reasoning steps, or change the semantic content of the answer are rejected.

Concise/Verbose reasoning-preserving rewrite prompt specification

Role. You are an expert solution rewriter. Given a problem, an original answer, an observed reasoning trace, a ground-truth final answer, and a target rewrite style, rewrite the answer while preserving its reasoning and final conclusion.

Task. Rewrite the original answer according to the target rewrite style `{rewrite_style}`, where the style is either `concise` or `verbose`.

If the target style is `concise`, produce a shorter and more direct version that removes redundancy. If the target style is `verbose`, produce a more detailed version by deliberately adding redundancy while keeping the core content unchanged.

The rewrite must satisfy all of the following constraints:

- It must preserve the original answer's final conclusion exactly.
- It must preserve the same solution strategy and actual reasoning process as the original answer.
- It must retain all reasoning steps that are necessary to support the original answer's final conclusion.
- For a concise rewrite, it may remove redundancy or compress exposition, but it must not remove any reasoning step that is necessary to the original solution.
- For a verbose rewrite, it may include redundant claims or additional reasoning steps, but it must not add substantial content to the reasoning process.
- It must not introduce new assumptions, new solution strategies, new evidence, or unsupported intermediate claims.
- It must not correct, replace, or otherwise alter the original answer's reasoning or final conclusion, including when the original answer contains an error.
- It must change only the presentation style requested by `{rewrite_style}` and must not introduce substantive changes that could alter the intended evaluation of the answer.
- If the original answer cannot be rewritten in the requested style while preserving its solution strategy, necessary reasoning steps, and final conclusion, output `Invalid for Rewrite`.

The output must use the following format exactly:

```
Rewrite Style: {rewrite_style}
Rewrite:
{rewrite}

Rewrite Validity: Valid
```

If the answer cannot be faithfully rewritten:

```
Rewrite Style: {rewrite_style}
Rewrite Validity: Invalid for
Rewrite
```

Problem: `{problem}`
Original Answer: `{original_answer}`
Observed Reasoning Trace: `{reasoning_trace}`
Ground-Truth Final Answer: `{gt_answer}`
Rewrite Style: `{rewrite_style}`
Your rewrite: