A CHEAP VERIFIER IS GOOD ENOUGH: LLM POST-TRAINING IS ROBUST TO ERRONEOUS REWARDS

Anonymous authors
Paper under double-blind review

ABSTRACT

When post-training large language models on tasks with semi-verifiable rewards, there are many factors (training steps, base model size, training order, data quality, verifier accuracy, etc.) that practitioners must contend with to maximize model performance. Yet, it remains unclear how well verifier agreement predicts post-training performance on such tasks. In this paper, we explore this question with over 11k H100 GPU-hours, across HealthBench and PRBench tasks in medical, legal, and finance domains. Across the tested domains, Qwen3 trainees (1.7B–8B on HealthBench; 8B on PRBench), evaluation splits, and frontier LLM reference judges (which we call golden verifiers), higher verifier agreement does not consistently identify the best training verifier. Expensive verifiers need not outperform inexpensive ones, and open-weight Gemma verifiers produce strong training outcomes. We compare two low-cost choices retrospectively — a *cost-reducing* choice and a *balanced* choice — with estimated grading cost reductions of 98.8%–99.7% relative to the golden grading protocols and average post-training score gaps of 1–3 points from the best evaluated training verifier. These averages include larger losses in individual settings; they do not establish that verifier choices are interchangeable.

1 INTRODUCTION

Reinforcement Learning with Verifiable Rewards (RLVR) has become the post-training method of choice for eliciting reasoning capabilities in Large Language Models (Shao et al., 2024; DeepSeek-AI et al., 2025; The Microsoft AI Team, 2026; Kimi Team et al., 2026; DeepSeek-AI et al., 2026). Early work focused on domains such as mathematics and coding, where deterministic checks provide a near-perfect reward signal (Austin et al., 2021). The field has since expanded from verifiable rewards to semi-verifiable rubric-based rewards for post-training. This extension allows training objectives to incorporate preferences and answer style alongside correctness. A verifier (sometimes called a grader or judge) is used to score task outputs according to a rubric composed of criteria (Gunjal et al., 2025; He et al., 2025; Viswanathan et al., 2025; Zhou et al., 2025; Lau et al., 2026). In some cases, these verifiers are used directly to provide rewards for post-training, where a natural hypothesis is that the accuracy of the rewards given by the verifier has a significant impact on the final performance of the post-trained model, especially in challenging domains with semi-verifiable rewards where taste preferences come into play.

This hypothesis has a natural human corollary: if Student A has a TA who grades their homework 30% incorrectly and Student B has a "better" TA who grades their homework 10% incorrectly, the common hypothesis is that Student B is likely to learn better over the course of the semester. Across the models we post-train (students), datasets (course material), verifiers (TAs), and golden verifiers (imperfect answer keys), better agreement with the answer key does not always identify the TA whose students learn most. Agreement remains informative, particularly for identifying very weak verifiers, but does not fully determine training outcomes. Our practical question is whether inexpensive verifiers can deliver strong post-training performance within a candidate pool that also includes strong, expensive models.

While prior work has explored post-training robustness to synthetically introduced error in verifiable rewards (Plesner et al., 2026; Cai et al., 2025), we examine the semi-verifiable domain where imperfect verifiers are used in practice for rubrics as rewards.

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.

This paper evaluates verifier selection for post-training on semi-verifiable tasks. We compare across model size, dataset domain, evaluation hardness (hard and a mixture of hard and non-hard tasks), reference choice (golden judges), and several verifiers.

Contributions: Across the tested combinations of these settings, we post-train an LLM and measure the performance impact. Across the tested settings, inexpensive verifiers can deliver scores close to the best evaluated training verifier, including in comparisons with strong, expensive candidates such as Grok 4.20. Higher verifier agreement does not always identify the best choice, although very weak verifiers can harm training. For example, the open-weight Gemma 4 31B and 26B verifiers produce some of the strongest training outcomes (Table 3 and appendix F). We compare two low-cost verifier choices retrospectively, with estimated grading cost reductions of 98.8%–99.7% relative to the golden grading protocols (Section 4). These reductions would translate to potentially tens of thousands of dollars per training run just for our smaller scale experiments.

2 METHODOLOGY

Reinforcement learning (RL), including RL from human feedback (RLHF), RL with verifiable rewards (RLVR), and RL with rubrics as rewards (RLRR) for post-training language models, works by giving rewards based on generated responses. One of the central difficulties in RL lies in shaping these rewards: what you reward and how much. RLVR often focuses on mathematics and coding—domains where answers can be verified deterministically, for example by checking against a known solution. This simplifies outcome verification, but does not by itself determine what to reward or by how much. Besides correctness, one might want to reward style choices, consistency, or reasoning or answer lengths (The Microsoft AI Team, 2026; Xiaomi, 2026). To extend RL post-training to open-ended tasks where correctness is not binary, recent work has turned to *rubric-based* rewards (Gunjal et al., 2025; Viswanathan et al., 2025). Rubrics decompose a response's quality into explicit, interpretable criteria, each of which can be assessed by an LLM judge and combined into a scalar reward. This structures the reward signal in a way that generalizes beyond verifiable domains, at the cost of introducing a strong dependency on the quality of the verifier.

2.1 RL WITH SEMI-VERIFIABLE REWARDS

RL with semi-verifiable rewards replaces deterministic checks, such as unit tests for code, with a judge's assessment of rubric criteria. Formally, given an input prompt $x_i \in \mathcal{D}$, we generate a response $\hat{y}_i \sim \pi_\theta(\cdot \mid x_i)$ from a policy π_θ. Each prompt is associated with a rubric $R_i = \{(c_{ij}, w_{ij})\}_{j=1}^{k_i}$, a set of k_i criteria c_{ij} with signed weights $w_{ij} \in \mathbb{R}$.[1] The role of a verifier V_ϕ is to assess whether a response satisfies each criterion,

$$V_\phi(x_i,\, \hat{y}_i,\, c_{ij}) \;\in\; \{0, 1\}, \tag{1}$$

where 1 indicates that the criterion is met and 0 that it is not. Following the HealthBench scoring convention (Arora et al., 2025), these verdicts are combined into an aggregate reward via a weighted sum,

$$r(x_i, \hat{y}_i) \;=\; \frac{\sum_{j=1}^{k_i} w_{ij}\, V_\phi(x_i, \hat{y}_i, c_{ij})}{\sum_{j:\, w_{ij}>0} w_{ij}} \;\leq\; 1, \tag{2}$$

where the denominator normalizes by the total positive weight, assumed nonzero, so that a response satisfying all positive criteria and none of the negative ones achieves the maximum reward of 1. PRBench uses a normalized variant seen in Appendix J. The RL objective is then to maximize the expected reward over the training distribution,

$$\max_\theta \; \mathbb{E}_{x_i \sim \mathcal{D},\, \hat{y}_i \sim \pi_\theta(\cdot|x_i)} \big[\, r(x_i, \hat{y}_i) \,\big]\,. \tag{3}$$

As Equations (2) and (3) make clear, verifier errors influence the training reward signal. We study whether agreement with a golden verifier predicts downstream policy performance; we do not characterize which errors cause particular training outcomes.

[1] Positive weights reward responses that satisfy the criterion; negative weights penalize responses that exhibit the behavior described by the criterion.

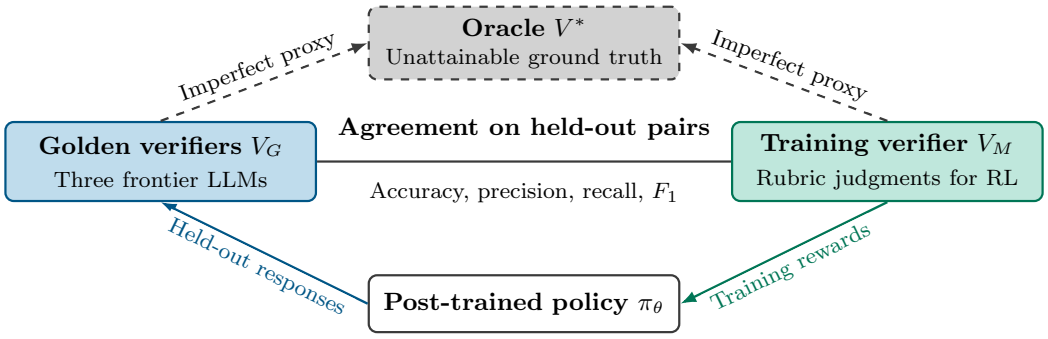

Figure 1: **Verifier hierarchy and roles.** Dashed arrows denote imperfect approximation of the unattainable oracle V^*. We compare training-verifier V_M's judgments with golden-verifier V_G's judgments on the same held-out response–criterion pairs. V_M supplies training rewards; V_G evaluates held-out policy responses. Agreement with a golden verifier does not establish ground-truth correctness.

2.2 The verifier hierarchy

Determining whether a response satisfies a criterion (e.g. correctness, helpfulness, safety) is itself a judgment call that no system can make with perfect fidelity. We formalize this notion through a hierarchy of verifiers, shown in Figure 1.

Oracle verifier. At the top of the hierarchy sits an unattainable *oracle* V^*, a theoretical verifier that returns ground-truth verdicts on every (response, criterion) pair. Human annotators and LLMs approximate V^* and can introduce their own noise, biases, and failure modes. For instance, inter- and intra-annotator disagreement are the empirical signatures of human imperfection; analogous inconsistencies arise in LLM-based judges.

Golden verifiers. Prior work has shown that frontier LLMs can match human-to-human agreement on some evaluation tasks (Zheng et al., 2023; Arora et al., 2025). We use frontier models as *golden verifiers* V_G: scalable reference judges that remain imperfect proxies for ground truth. They let us study how training-time verifier choice affects post-training outcomes without requiring human annotation of every response.

Training-time verifiers. We use an LLM-based *training-time verifier* V_M to drive the policy gradient. We measure agreement with the golden verifiers, following the reference-judge setup of Liu et al. (2026). Agreement is not ground-truth correctness. The golden verifiers also score the trained policies; their grading cost precluded using them for training. The training pool nevertheless includes strong, expensive models such as Grok 4.20. Using the same references for agreement and policy evaluation defines a common evaluation target, but does not rule out shared judge biases as policy outputs change during training. Verifier alignment is quantified by accuracy, precision, recall, and F_1 between V_M verdicts and V_G verdicts on held-out (response, criterion) pairs. For each dataset, these pairs come from a fixed slice of 20 `hard` tasks for the `hard` split, plus 20 `non-hard` tasks for the mixed `all` split. To reduce dependence on any particular reference, we evaluate against three frontier models from different vendors and report averages across them.[2] We post-train each policy using its assigned training-time verifier to score rollout responses, and use the golden verifiers to score evaluation responses.

We batch multiple criteria in each verifier call and refer to the maximum number of criteria per call as the verifier's batch size, distinct from the policy-training batch size.

[2]We compare against each judge's labels separately and then average the metrics, rather than combining labels first.

Table 1: Available dataset pools for training and evaluation. "Hard eval-only" contains only `hard` tasks; "Total eval" also includes held-out `non-hard` tasks. Policy scoring uses subsets of these pools.

Domain	**Train** (n)	**Hard eval-only** (n)	**Total eval** (n)
HealthBench	3600	1000	1400
PRBench legal	225	249	274
PRBench finance	269	299	329

2.3 Experimental setup

We focus our experiments on three task domains: medical, finance, and legal tasks. The medical tasks are from HealthBench (Arora et al., 2025), while the finance and legal tasks are from PRBench (Akyürek et al., 2025).

Data. All the datasets contain subsets labeled as hard. We denote the split with the hard tasks as `hard` and the split with the remaining as `non-hard`. We use the `hard` and 10% of the `non-hard` tasks for evaluation, and leave the remaining non-hard problems for training (the `non-hard` are split into training and evaluation using a fixed seed of 42). Per-domain split sizes are listed in Table 1. Some PRBench prompts are very long, so we drop the 3 tasks exceeding 90,000 characters to keep prompts and responses within Qwen3's 32,768-token native context (Appendix K); Table 1 reflects this filter.

Trainees are **Qwen3 1.7B**, **4B**, and **8B** (Yang et al., 2025), all initialized from the Hugging Face checkpoints. We vary trainee size on HealthBench; PRBench uses Qwen3 8B.

Training-time verifiers ($\mathbf{V_M}$). We tested a wide selection of pretrained models on HealthBench generations from Qwen3 8B on the `hard` subset; the full screening results for all candidates are in Appendix D. From the 52 tested models, we select 12 for further experiments. These are: Gemma 4 31B, Gemma 4 26B, GPT oss 120B, GPT oss 20B, Gemini 3 flash preview, Grok 4.20, Qwen3 235B, Ministral 14B, Qwen3 8B (a self-judge for the 8B trainee), Mistral 4 small, Qwen3 Next 80B, and LFM2 24B. API instability and grading cost prevented some runs, including runs with Grok 4.20, from completing. Incomplete runs are excluded from the policy comparisons; candidate coverage therefore varies across domains and trainee sizes. Completed runs with poor or collapsed policies remain in the result tables. All are run at batch=3, thinking=`low`.

Golden verifiers ($\mathbf{V_G}$). Our headline golden is **Claude Opus 4.6** with thinking=`medium` and batch=1 (one criterion per call), which we used for most early experiments. To assess robustness to the choice of V_G, we additionally run two alternate goldens drawn from different model families: **Gemini 3.1 Pro** and **GPT-5.5**, both with the same grading configuration. Re-ranking the twelve V_M candidates and the trained policies under these alternate golden verifiers is purely an evaluation operation and does not require additional training. Unless stated otherwise, golden-judge results are averaged across all three references, following the approach of Verga et al. (2024). Averaging reduces dependence on any single reference, although the judges may share biases. Alignment metrics between the three golden verifiers, including their agreement with a 2-of-3 majority aggregate, are reported in Appendix E.

To spot-check the reference labels, two human experts in rubric data reviewed 50 HealthBench criteria graded by Opus 4.6. Both reviewers were unable to discern any errors in Opus's labels. However, this small check does not establish perfect golden-judge accuracy.

Training algorithm and hyperparameters can be found in Appendix B. We use GRPO with within-prompt reward standardization and no effective KL penalty; verifier differences therefore affect learning through normalized advantages.

Policy evaluation. The golden verifiers score trained policies on larger task sets than the agreement slices, and the two are disjoint: 100 `hard` and 40 `non-hard` HealthBench tasks, and 53–55 PRBench tasks per domain. Appendix I gives the exact task sets, how the `all` split is weighted, and how rubric verdicts become policy scores.

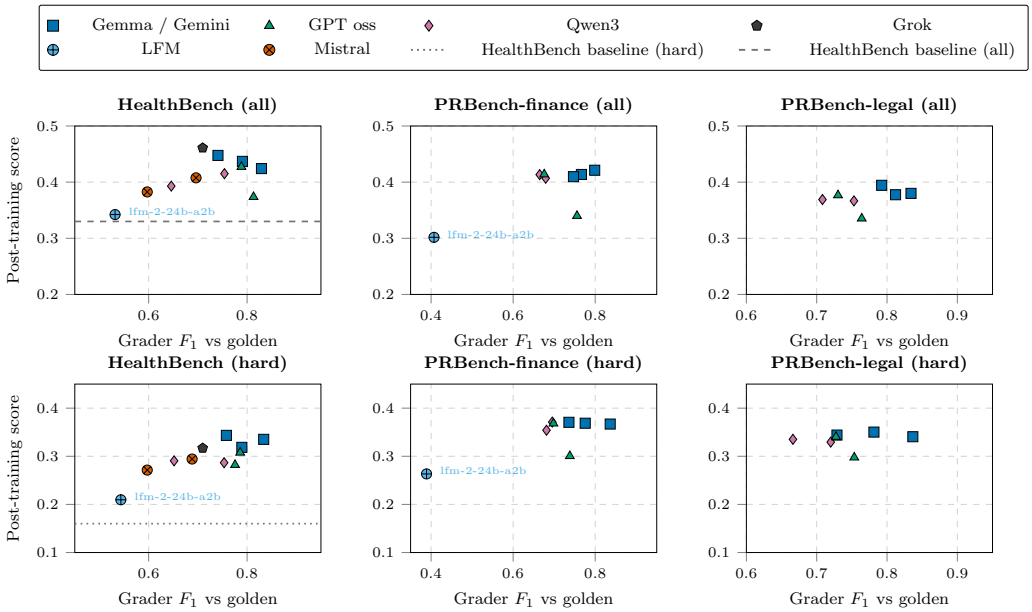

Figure 2: **Higher verifier F_1 agreement with a golden judge does not consistently correspond to higher post-training score.** For each dataset (HealthBench, PRBench finance, PRBench legal), every point is one (verifier, split) evaluation of a completed Qwen3 8B training run, averaged across the three golden judges. Horizontal lines mark the HealthBench no-post-training baselines. PRBench points use the clipped aggregate; normalized-score baselines are reported alongside normalized policy scores in the appendix tables (Appendices G and J). The Qwen3 8B training verifier is omitted from this plot and Figure 3; its collapsed-policy outcomes remain in the appendix tables.

3 Verifier Agreement and Post-Training Performance

We first compare verifier agreement with post-training performance across domains and trainee sizes. Detailed per-verifier metrics and policy scores are in Appendices F and G.

3.1 Agreement is informative but does not fully rank training verifiers

Figure 2 compares six dataset–split combinations: HealthBench, PRBench finance, and PRBench legal, each with hard and mixed evaluation sets. Agreement and policy performance can be positively associated without giving identical verifier rankings. For example, Qwen3 8B scores on HealthBench-all are 0.37 with GPT oss 20B and 0.46 with Grok 4.20, despite GPT oss 20B having higher F_1; inexpensive Gemma 4 26B scores 0.45 (Table 10). The practical result is that low grading cost can coexist with strong training outcomes, rather than that agreement has no predictive value.

Figure 3 shows the comparison across the tested Qwen3 model sizes (up to 8B parameters).

Figure 3 also shows that weak verifiers can harm training: Qwen3 8B trained with LFM2 24B still clears its no-training baseline on HealthBench-hard, whereas Qwen3 1.7B trained with the same verifier falls well below baseline. Failure is not confined to low agreement: the Qwen3 8B training verifier yields below-baseline scores despite $F_1 \approx 0.76$ (Table 9). These outcomes do not establish a universal quality threshold or explain the causes of failure. Here, $1 - F_1$ measures disagreement with a reference, not the probability of an incorrect rubric verdict.

3.2 Agreement, verifier capability, and training outcomes

The comparisons support a role for verifier quality without requiring a monotonic ranking by agreement. On HealthBench-hard, LFM2 24B has $F_1 = 0.54$ and yields low policy scores, whereas Gemma 4 31B and Gemma 4 26B have $F_1 = 0.83$ and 0.76, respectively, and both yield an 8B

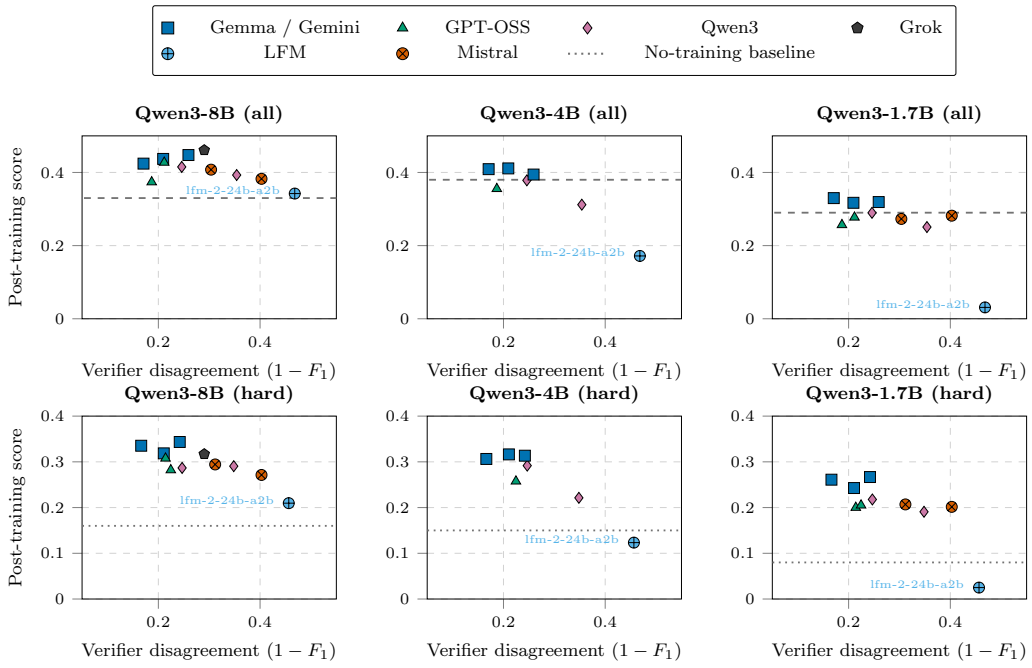

Figure 3: **Verifier disagreement and post-training performance across Qwen3 trainee sizes.** Results are averaged evaluation scores across the three golden judges. The horizontal dotted line represents baseline performance for each model. Each column is a different trainee size (8B, 4B, 1.7B), with mixed and hard evaluation splits in the two rows. The x-axis is $1 - F_1$; it does not define a noise-tolerance threshold.

policy score of 0.34 (Table 9). The same table reports GDPval as a separate measure of general model capability (Patwardhan et al., 2025). These comparisons concern particular verifiers and do not isolate a causal effect of either capability or agreement.

Appendix O summarizes the descriptive value of other agreement metrics in the existing results. Accuracy and F_1 explain part of the observed variation in policy scores, and the strongest summary depends on the records included. Thus, evidence that agreement is informative is compatible with our cost finding: inexpensive verifiers can still be among the strongest training choices.

Criterion-level agreement also differs from the signal used for learning. The reward aggregates criteria with signed weights, while GRPO standardizes rewards within each prompt's rollout group. Unweighted F_1 does not measure whether those weighted rewards give the same relative advantages to competing responses. Our fixed-response screening metrics provide a practical comparison of verifiers, but do not characterize training-time errors as the policy changes. We therefore interpret the per-verifier results as observed cost–performance tradeoffs, without assigning significance to small score gaps or attributing them solely to agreement.

4 Practical Implications for Reducing Post-Training Costs

The comparisons in Figure 3 motivate considering verifier cost alongside agreement. We compare two choices identified retrospectively: (1) the **cost-reducing Pareto verifier** — the cheapest candidate with accuracy ≥ 0.80 against the golden judge, falling back to the cheapest if none qualifies; and (2) the **balanced Pareto verifier** — the fixed choice Gemma 4 26B, selected for its mean post-training score in our candidate pool. Table 19 compares these with other choices, and Figures 4 and 7 show their cost–score tradeoff per dataset and on average. The balanced choice uses observed training outcomes; it is not an automatic selection rule validated without a post-training sweep, and neither choice guarantees Pareto optimality in a new setting.

Table 2: **Grading resources and cost on a fixed HealthBench probe.** Calls and total input-plus-output tokens are per task; USD is per 1,000 tasks at the recorded prices. Golden protocols use one criterion per call and medium thinking; training protocols use up to three criteria and low thinking. Probe sampling and pricing qualifications are in Appendix M.

Verifier	Calls/task	Tokens/task	USD/1k tasks
Claude Opus 4.6 (golden)	11.6	21,321	133.20
GPT-5.5 (golden)	11.6	17,093	99.70
Gemma 4 31B	4.3	5,498	0.77
Gemma 4 26B	4.3	5,514	0.45
GPT oss 20B	4.3	6,157	0.08

We compare the choices using three metrics, each computed per cell (one dataset–split–golden combination), among candidates with a completed run and an available cost estimate: the **score gap**, the post-training score of the empirically best eligible verifier in the cell minus the score of the pick (lower is better; 0 means the pick recovered the best verifier); the **uplift**, the fraction of the available post-training improvement over the no-training baseline that the pick captures; and the **cost reduction**, the fraction of per-task grading cost saved relative to the golden grading protocol. Averaged across all three datasets, both evaluation splits (`all` and `hard`), and the three golden judges, the balanced Pareto verifier achieves a $+0.010$ score gap, captures 95% of the available uplift, and gives a 98.8% cost reduction; the cost-reducing Pareto verifier achieves a $+0.027$ score gap, 82% uplift, and a 99.7% cost reduction (Table 3). The cheaper choice has a larger score gap, including 0.062 (6.2 points) on HealthBench-all, where it captures 46% of the available uplift.

These are estimates of verifier API cost under the recorded grading protocols and pricing snapshot, not reductions in total training cost. They depend on model prices (including fallback estimates where prices were unavailable), caching, and batching; golden judges grade one criterion per call, while training verifiers grade up to three. We did not train a policy with a golden verifier, so the golden-cost comparison does not establish equivalent policy quality. Appendix M reports judge calls and tokens per task.

Where the grading savings come from. Table 2 separates grading resource use from dollar cost on the HealthBench cost probe. The golden protocols require $2.7\times$ as many calls as the training-verifier protocols, but call reduction alone does not explain the price difference. Claude Opus 4.6 uses about $3.9\times$ as many total tokens as Gemma 4 26B while costing about $296\times$ as much on this probe. Under a matched batch=3, thinking=`low` protocol, the token ratio falls to $1.44\times$ (Appendix M). Lower token prices therefore account for most of the observed dollar savings.

Self-hosting offers a complementary cost measure: Gemma 4 26B grades a HealthBench rollout in about 0.51 H100-seconds, equivalent to roughly 14 H100-hours for $96{,}000$ rollouts at this measured rate. These measurements make the source of the savings concrete while keeping API pricing separate from policy quality and total training cost. Actual training bills also depend on response-length growth and cache reuse. The probe supports comparison at a fixed operating point; the per-domain cost–score results remain the basis for comparing training verifiers.

5 Related Work

Reinforcement learning with verifiable rewards Recent work has focused on scaling model capabilities through RL (Silver & Sutton, 2025; OpenAI, 2024; Zeng et al., 2025). DeepSeek-R1 (DeepSeek-AI et al., 2025) using GRPO (Shao et al., 2024) is a key example of how RL can be used to cost-effectively increase frontier model capabilities by training on problems with verifiable rewards, such as mathematics and coding, where there is less risk of reward hacking.

Rubrics as rewards beyond verifiable domains. The post-training literature is migrating from deterministic verifiers (math, code) to LLM judges scoring weighted rubrics in domains where ground truth is undefined. Gunjal et al. (2025) introduce Rubrics-as-Rewards on synthetic medical data and show uplift on HealthBench; Viswanathan et al. (2025) show weighted checklists outperform reward

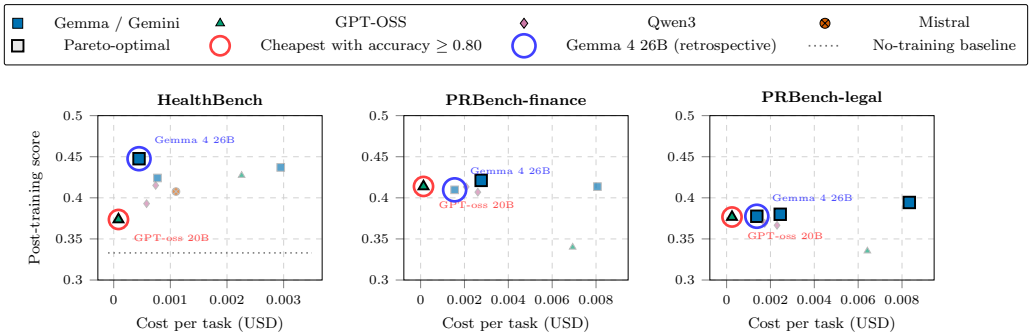

Figure 4: **Cost vs. post-training score, per dataset, averaged across the three goldens, with our two retrospective choices highlighted.** One point per (dataset, verifier); panels identify datasets, and marker shape and color encode verifier provider family. Bold-outlined markers are Pareto-optimal in (cost, post-training score) for that dataset; smaller faded markers are sub-optimal. The dotted line marks the HealthBench no-post-training baseline. PRBench baselines are reported on the normalized scale in the appendix tables. PRBench scores use the clipped aggregate. **Red rings**: cost-reducing Pareto verifier picks. **Blue rings**: the fixed balanced choice, Gemma 4 26B. These examples illustrate different cost–score tradeoffs within the tested candidate pool.

Table 3: **Verifier selection across** 18 **evaluation cells** (3 datasets × 2 splits × 3 goldens; trainee Qwen3 8B). **Splits**: `all` = `hard` + `non-hard` combined with the weights in Appendix I; `hard` = the `hard` subset only. The cost-reducing rule, fixed balanced choice, and score gap and cost reduction metrics are defined in Section 4. **Uplift** = (chosen − baseline)/(max − baseline) is the fraction of available post-training improvement captured (100% matches the best verifier per cell; 0% matches no training). **Bold** marks the better value per column between the cost-reducing and balanced picks.

		Overall		**HealthBench**		**PR-finance**		**PR-legal**	
Variant	**Metric**	**all**	**hard**	**all**	**hard**	**all**	**hard**	**all**	**hard**
Cheapest verifier (no quality floor)	Score gap	+0.030	+0.026	+0.062	+0.054	+0.011	+0.011	+0.017	+0.013
	Uplift	77%	86%	46%	70%	95%	95%	89%	93%
	Cost reduction	99.7%	99.7%	99.4%	99.5%	99.8%	99.8%	99.8%	99.8%
Cost-reducing Pareto verifier	Score gap	+0.030	+0.023	+0.062	+0.046	**+0.011**	**+0.011**	**+0.017**	+0.013
	Uplift	77%	88%	46%	76%	**95%**	**95%**	**89%**	**93%**
	Cost reduction	**99.7%**	**99.7%**	**99.4%**	**99.1%**	**99.8%**	**99.8%**	**99.8%**	**99.8%**
Balanced Pareto verifier	Score gap	**+0.012**	**+0.008**	**+0.001**	**+0.000**	+0.015	+0.012	+0.019	**+0.012**
	Uplift	**93%**	**96%**	**99%**	**100%**	94%	**95%**	87%	**93%**
	Cost reduction	98.8%	98.9%	98.1%	98.1%	99.0%	99.2%	99.1%	99.0%

models; Zhou et al. (2025) use rubrics as exploration scaffolds; He et al. (2025) build AdvancedIF for instruction following; and Su et al. (2025) extend RLVR to medicine, chemistry, and economics. In every case, the LLM judge replaces the deterministic verifier, raising the open question we address: how good the verifier needs to be.

Imperfect verifiers and noisy rewards in RLVR. Recent work has explored how much noise RLVR can tolerate while still giving a large uplift. Plesner et al. (2026) showed that, with symmetric synthetic noise on coding unit tests, RLVR keeps mean validation reward within about 1 point of the clean baseline at a fixed late checkpoint for noise rates up to 20%. Related work derives bias-corrected GRPO estimators under stochastic-channel reward noise: Cai et al. (2025) (asymmetric

false-positive/false-negative), Mansouri et al. (2025) (unbiased Bernoulli correction), and Rad et al. (2026) (a Youden's-J phase transition). Shao et al. (2025) reported the counterintuitive results that Qwen2.5-Math can learn from purely random rewards. Follow-up work by Zhu & Kang (2026) and Chen et al. (2026) attribute the effect to clipping bias and contamination.

LLM-as-Judge meta-evaluation. We measure each candidate verifier's agreement (precision, recall, F1 score, and Spearman ρ) with a frontier LLM acting as the operational *golden* reference, following Zheng et al. (2023). Standard judge benchmarks include RewardBench (Lambert et al., 2025), JudgeBench (Tan et al., 2024), and the rubric-specific RubricEval (Pan et al., 2026); Gu et al. (2026) catalog the inherited biases (position, length, prompt-phrasing, self-preference). Gao et al. (2023) establish that proxy-reward accuracy and gold-reward outcomes can diverge under optimization. Wen et al. (2025) and Kim et al. (2025) study why reward-model evaluation metrics can imperfectly predict downstream policy performance. We extend this empirical question to the cost and training outcomes of rubric-based verifiers across three professional domains.

Cost-aware judge and verifier selection. Salinas et al. (2025) tune LLM-judge design decisions to reduce evaluation cost by three orders of magnitude. Dorner et al. (2025) prove a hard limit: a judge no stronger than the system it grades cannot reduce the number of required ground-truth labels by more than a factor of two. Liu et al. (2026) compare multiple reasoning and non-reasoning training judges under a single fixed gold-standard reference, and show that trained policies can exploit judge biases. Our study compares off-the-shelf rubric verifiers across grading costs, three professional domains, and three golden references.

6 Conclusions and Future Work

Across the tested rubric-based post-training tasks, inexpensive verifiers can deliver strong policy scores within a candidate pool that includes strong, expensive models. Open-weight Gemma verifiers provide particularly favorable observed cost–score tradeoffs. Agreement remains informative, as illustrated by LFM2's weak results, but does not fully determine the ranking of training outcomes. Our cost claim concerns the performance achieved by selected inexpensive verifiers in these experiments; it does not require agreement to be uncorrelated with performance or all verifiers to be interchangeable. Understanding which response-level errors affect learning could improve selection beyond aggregate agreement metrics.

7 Limitations

Unstable vendor APIs, several of which went down during this research, and limited post-training compute constrained the number of judges and experiments. Our results cover Qwen3 trainees up to 8B and one GRPO setup; they do not establish generalization to other model families, algorithms, KL settings, or longer training horizons. Hyperparameters were not tuned separately for weaker verifiers.

Within the available compute and grading budget, we evaluated a broad set of verifiers, domains, and trainee sizes. Some configurations were launched more than once, but these repeats were not controlled seed replications (one changed the verifier-serving backend). The reported comparisons therefore describe observed outcomes and do not separate training randomness from verifier effects. Evaluation uses subsets of the held-out pools; task resampling would address evaluation-sample variability, but would not estimate training-seed uncertainty.

Golden-verifier judgments remain LLM references rather than human ground truth, and screening agreement is measured on fixed responses rather than each trained policy's outputs. We did not train with a golden verifier or conduct a dedicated test of exploitable verifier biases and reward hacking (see Appendix H for observed response-length drift). The cost–quality choices are retrospective and depend on the candidate pool; Appendix O compares other agreement summaries, but we do not establish a generally superior selection metric.

AI USE STATEMENT

Following the ICLR 2027 AI policy, we disclose that we used large language models to edit a research paper to improve readability. This use is distinct from the LLM policies and verifiers studied in our experiments, whose roles are described in Section 2.3. We have reviewed all AI-assisted work, and the authors are responsible for the contents of this submission.

REPRODUCIBILITY STATEMENT

Section 2.3 describes the datasets, splits, trainee models, and verifier configurations. Appendix B provides the training algorithm and hyperparameters, and Appendix C documents the grader prompts. The reward aggregation and PRBench filtering are described in Appendices B.1, J and K. Detailed verifier and policy results appear in Appendices F and G, and Section 7 states the limits of the experimental evidence.

REFERENCES

Afra Feyza Akyürek, Advait Gosai, Chen Bo Calvin Zhang, Vipul Gupta, Jaehwan Jeong, Anisha Gunjal, Tahseen Rabbani, Maria Mazzone, David Randolph, Mohammad Mahmoudi Meymand, Gurshaan Chattha, Paula Rodriguez, Diego Mares, Pavit Singh, Michael Liu, Subodh Chawla, Pete Cline, Lucy Ogaz, Ernesto Hernandez, Zihao Wang, Pavi Bhatter, Marcos Ayestaran, Bing Liu, and Yunzhong He. PRBench: Large-Scale Expert Rubrics for Evaluating High-Stakes Professional Reasoning. https://arxiv.org/abs/2511.11562v1, November 2025.

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, May 2025.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program Synthesis with Large Language Models, August 2021.

Xin-Qiang Cai, Wei Wang, Feng Liu, Tongliang Liu, Gang Niu, and Masashi Sugiyama. Reinforcement Learning with Verifiable yet Noisy Rewards under Imperfect Verifiers, December 2025.

Peter Chen, Xiaopeng Li, Ziniu Li, Wotao Yin, Xi Chen, and Tianyi Lin. Exploration vs Exploitation: Rethinking RLVR through Clipping, Entropy, and Spurious Reward, January 2026.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang,

Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *Nature*, 645(8081): 633–638, September 2025. ISSN 0028-0836, 1476-4687. doi: 10.1038/s41586-025-09422-z.

DeepSeek-AI, Anyi Xu, B. Li, Bangcai Lin, Bing Xue, BingCheng Xian, Bingzheng Xu, Bochao Wu, Bowei Zhang, Boyi Deng, C. C. Yu, Chao Jin, Chaofan Lin, Chen Dong, Chenbing Wang, Chenfan Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chengyuan Zhang, Chenhao Xu, Chenqi Zhao, Chenze Shao, Chuhao Wang, Chuqi Zhang, Damai Dai, Dejian Yang, Deli Chen, Di Huang, Di Wu, Donghao Li, Erhang Li, Eric Fu, F. Zhou, Fangwei Zhou, Fangyun Lin, Fangzhou Yuan, Feiyu Xia, Fucong Dai, Guangbo Hao, Guanglin Li, Guanting Chen, Guoai Cao, Guofan Fan, Guolai Meng, Guowei Li, Haichuan Zhang, Haiyang Ma, Haiyang Shen, Han Li, Han Yu, Han Zhang, Hangyuan Deng, Hanwei Xu, Hanxiang Xu, Hanxun Zhong, Hao Guo, Hao Jiang, Hao Li, Hao Qin, Haodong Wen, Haofen Liang, Haofeng Huang, Haohua Liu, Haoling Zhang, Haoming Luo, Haoran Yang, Haotian Xu, Haotian Yuan, Haoting Huang, Haowen Luo, Haoyang Cai, Haoyu Chen, Haozhe Ji, Hengran Zhang, Hengrui Wang, Hengxu Wu, Honghui Ding, Hongxuan Tang, Huadong Wang, Huanqi Cao, Huazuo Gao, Hui Qu, Hui Zeng, J. Yang, J. H. Jin, J. H. Zhang, J. X. Zou, Jia Yu, Jiahui Zhou, Jiajun Chen, Jialiang Huang, Jialin Zhao, Jiamin Tang, Jian Zhou, Jianan Tong, Jianwen Li, Jiaqi Zhu, Jiarui Wang, Jiasheng Ye, Jiashi Li, Jiaxin Xu, Jiaying Ding, Jibai Lu, Jiewen Hu, Jin Yan, Jincheng Zhai, Jingchang Chen, Jingcheng Hu, Jingli Zhou, Jingsheng Xu, Jingting Xiang, Jingyan Yun, Jingyang Yuan, Jingyuan Cheng, Jinhua Zhu, Jinpeng Wang, Jinyi Chen, Jinyi Hu, Jiping Yu, Jueliang Guo, Junbo Pei, Junbo Sun, Junguang Jiang, Junjie Qiu, Junkang Zhou, Junqi Liu, Junren Li, Junxian Li, Junxiao Song, Junyi Guo, Kai Dong, Kaifeng Chen, Kaige Gao, Kang Guan, Kangdong Yuan, Ke Hong, Ke Xu, Kefan Zhao, Kexin Ji, Kexin Zhang, Kexing Zhou, Kuai Yu, Lan Zhang, Lean Wang, Lecong Zhang, Lei Wang, Letian Gao, Liang Zhao, Liansheng Xu, Lihua Guo, Lingxiao Luo, Lingyue Fu, Litao Deng, Litong Wang, Liyue Zhang, Longhao Chen, Lu Chen, Luotian Huang, Luyao Ma, Luyao Wang, M. S. Di, Max Mei, Menghao Ye, Miao Cui, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingjing Zhang, Mingqi Wei, Mingshu Chen, Mingxing Liu, Mingxu Zhou, Mingyu Xu, Mingyu Yang, Mingze Wang, Muyang Chen, Ni Shentu, Ning Wang, Niufang Ning, Panpan Huang, Peixin Cong, Peiyi Wang, Peiyuan Xin, Pengfei Ren, Pengfei Yan, Pengle Zhang, Qi Kang, Qi Tang, Qiancheng Wang, Qiang Li, Qihao Zhu, Qingyang Li, Qinyu Chen, Qiushi Du, Qizhou Guo, Rongxian Xu, Rui Ding, Rui Hu, Rui Tian, Rui Yu, Ruidong Zhu, Ruifan Xu, Ruihan Yang, Ruihang Xia, Ruijie Lu, Ruilin Geng, Ruipeng Hong, Ruiqi Ge, Ruisong Zhang, Ruize Sun, Ruizhe Pan, Runji Wang, Runqian Chen, Runxin Xu, Ruohong Tian, Ruomeng Shen, Ruoyu Zhang, Ryan X, S. H. Liu, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaofei Cai, Shaoheng Nie, Shaoyuan Chen, Shengding Hu, Shengkai Lin, Shengwen Ran, Shengyu Liu, Shengyuan Jia, Shi Bai, Shi Feng, Shicheng Xu, Shichun Liu, Shiqiang Hu, Shirong Ma, Shiyu Wang, Shiyuan Feng, Shufan Gong, Shuhan Lin, Shuiping Yu, Shunfeng Zhou, Shuo Yang, Shuomeng Wang, Shuting Guo, Shuting Pan, Shuying Yu, Sinuo Cao, Siyi Lin, Sizhe Chen, Songyang Chen, Songyang Zhou, Tao Ni, Tao Yun, Tian Jin, Tian Pei, Tian Ye, Tianle Lin, Tianran Ji, Tianyi Cui, Tianyuan Yue, Tingting Yu, Tongrui Xiong, Wangding Zeng, Wei Liu, Wei Zhang, Weibin Xu, Weihao Zeng, Weilin Zhao, Wen Liu, Wenfeng Liang, Wenjie Pang, Wenjing Luo, Wenjing Yao, Wenjun Gao, Wenkai Shao, Wenkai Yang, Wenli Zhang, Wenlu Wang, Wenlve Huang, Wenqian Yan, Wentao Zhang, Xi Gao, Xiang He, Xiang Li, Xiangli Li, Xiangwen Wang, Xiangying Zhang, Xiankui Wei, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojian Qu, Xiaokang Chen, Xiaokang Zhang, Xiaotao Nie, Xiaoyao Zou, Xiaoyuan Li, Xicheng Guo, Xieting Chu, Xin Cheng, Xin Liu, Xin Xie, Xinbo Xu, Xingchao Liu, Xingchen Liu, Xingkai Yu, Xingyou Li, Xintong Yao, Xinyang Chen, Xinyong Jiang, Xinyu Yang, Xinyu Yang, Xu Chen, Xuanyu Wang, Xubei Zhong, Xuecheng Su, Xuejie Liu, Xuheng Lin, Xujie Fan, Xuncheng Zhao, Xuwei Fu, Y. C. Yan, Y. H. Jiang, Y. T. Wu, Y. W. M, Y. Z. Wang, Yafei Gao, Yang Yang, Yang Zhang, Yanru Ma, Yanwen Huang, Yao Li, Yao Li, Yao Meng, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yaoyang Ye, Yehang Yin, Yexinrui Wu, Yi Qian, Yi Tao, Yi Yu, Yichao Zhang, Yichen Jiang, Yicheng Wang, Yifan Ding, Yifan Shi, Yifeng Peng, Yifeng Zhai, Yijia Wu, Yiliang

Xiong, Yilun Wang, Ying He, Ying Zhou, Yingjia Luo, Yinmin Zhong, Yiping Wang, Yisong Wang, Yixiang Zhang, Yixiao Chen, Yixuan Tan, Yixuan Wei, Yiyang Ma, Yiyao Yang, Yiyuan Liu, Yizai Cai, Yizhen Wei, Yizhi Wang, Yonglun Yang, Yongqi Zhuo, Yongqiang Guo, Yongtong Wu, Yu Wu, Yu Zhang, Yuan Bian, Yuan Cheng, Yuan Ou, Yuan Sun, Yuanfan Xu, Yuanhang Sun, Yuanhao Li, Yuchen Liu, Yuchen Yao, Yudong Han, Yuduan Wang, Yuhan Wu, Yuhao Meng, Yuheng Zou, YuKun Li, Yunchuan Wang, Yunfan Xiao, Yunfan Xiong, Yupeng Chen, Yuqian Cao, Yuqian Wang, Yuqing Chen, Yushun Zhang, Yutong Lin, Yuwei Xiao, Yuxian Gu, Yuxiang Chen, Yuxiang Huang, Yuxiang Luo, Yuxiang You, Yuxin Chen, Yuxin Xiang, Yuxuan Liu, Yuxuan Zhou, Yuyang Zhou, Yuzhe Guo, Yuzhen Huang, Yuzhuo Bai, Z. Y. Z, Zanlin Ni, Zehao Wang, Zehua Zhao, Zehui Ren, Zejun Zhao, Zhangli Sha, Zhanying Wang, Zhaochen Zhang, Zhaoshuai Du, Zhe Fu, Zhean Xu, Zhenda Xie, Zheng Liu, Zhengyan Zhang, Zhenhua Dong, Zhewen Hao, Zhibang Wang, Zhibin Gou, Zhicheng Ma, Zhihao Li, Zhihong Shao, Zhihuan Huang, Zhijie Li, Zhirui Lu, Zhixian Huang, Zhixuan Chen, Zhixuan Chen, Zhixuan Pan, Zhiyu Wu, Zhizhou Ren, Zhu He, Zhuoshu Li, Zhuping Zhang, Zian Xu, Zihao Wang, Zihui Gu, Zijia Zhu, Zili Zhang, Zilin Li, Zilong Hou, Zilong Lyu, Ziqiao Wang, Ziwei Xie, Ziya Zhang, Ziyi Gao, Zizheng Pan, Zonglin Li, Zongqing Yao, Zui Chen, Zuofan Wu, Chenchen Ling, Chengyu Hou, Chong Chen, D. Li, Di Qi, Dongjie Ji, Fang Wei, Fanyi Xia, Fei Xie, Feiyi Tan, Hailong Guo, Haiyan Zhai, Hui Zhou, Huihui Tan, Huijie Li, Jia Luo, Jia Song, Jialu Cai, Jian Liang, Jiangting Zhou, Jiaqi Gao, Jiayi Shao, Jie Chen, Jieyu Yang, Jin Chen, Jingde Zhang, Jingzi Zhou, Jinqian Wang, Jinyang Liu, JinZhao Sun, Junhua Ling, Junmin Zheng, Kaicheng Yang, Ke Xu, Le Su, Leyi Xia, Liangfeng Ding, Lin Zhuo, Linwang Ma, Linyan Zhu, Liyu Cai, Luqi Yao, M. K. Zhang, Meng Li, Miao Lin, Miaojun Wang, Min Zhang, Mingming Li, Mingming Wang, Mingze Yin, Minmin Han, Nan Cao, Ning Wang, Ningxin Ma, Panpan Wang, Peihan Lin, Peng Sun, Peng Zhang, Qian Ying, Qiang Xiang, Qiao Wang, Qingmiao Mao, Qiwei Jiang, Rongli Jin, Ruyi Chen, Sha Tao, Shangmian Sun, Shaoqing Wu, Shichao Zou, Si Lei, Tianyang Zhang, Tianyu Sun, Tingting Yin, W. L. Xiao, Wei An, Wei Li, Wei Wang, Weiwei Lin, Wenqing Hou, X. Lin, Xiangfei Meng, Xianzhu Huang, Xiao Peng, Xiaoqian Li, Xiaoting Zhang, Xiaowen Sun, Xiaoxiang Wang, Xiaoyu Ye, Xinrou Zhang, Xinyu Zhang, Xue Cao, Xueyin Chen, Yanan Zhou, Yanhong Xu, Yao Xia, Yao Xu, Yi Shao, Yihong Zhang, Yiling Ma, Ying Tang, Yining Lou, Yiru Chen, Yishi Piao, Yixuan Chen, Yong Xiong, Yuchen Xuan, Yuehan Yang, Yuer Xu, Yukun Zha, Yunxian Ma, Yuping Lin, Yuting Yan, Yutong Xie, Yuwen Sheng, Yuxuan Zhu, Zekai Zhang, Zhe Ju, Zhenzhen Lin, Zheren Gao, Zheyang Sun, Zhigang Yan, Zhongyu Wu, Zi Wang, Zihua Qu, Ziling Yan, and Ziyi Wan. DeepSeek-V4.1-Flash: Pushing the Limits of KV Cache Compression, September 2026.

Florian E. Dorner, Vivian Yvonne Nastl, and Moritz Hardt. Limits to scalable evaluation at the frontier: LLM as judge won't beat twice the data. In *The Thirteenth International Conference on Learning Representations*, March 2025.

Leo Gao, John Schulman, and Jacob Hilton. Scaling Laws for Reward Model Overoptimization. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 10835–10866. PMLR, July 2023.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Zhouchi Lin, Bowen Zhang, Lionel Ni, Wen Gao, Yuanzhuo Wang, and Jian Guo. A survey on LLM-as-a-Judge. *The Innovation*, pp. 101253, January 2026. ISSN 26666758. doi: 10.1016/j.xinn.2025.101253.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as Rewards: Reinforcement Learning Beyond Verifiable Domains, October 2025.

Yun He, Wenzhe Li, Hejia Zhang, Songlin Li, Karishma Mandyam, Sopan Khosla, Yuanhao Xiong, Nanshu Wang, Xiaoliang Peng, Beibin Li, Shengjie Bi, Shishir G. Patil, Qi Qi, Shengyu Feng, Julian Katz-Samuels, Richard Yuanzhe Pang, Sujan Gonugondla, Hunter Lang, Yue Yu, Yundi Qian, Maryam Fazel-Zarandi, Licheng Yu, Amine Benhalloum, Hany Awadalla, and Manaal Faruqui. AdvancedIF: Rubric-Based Benchmarking and Reinforcement Learning for Advancing LLM Instruction Following, November 2025.

Sunghwan Kim, Dongjin Kang, Taeyoon Kwon, Hyungjoo Chae, Dongha Lee, and Jinyoung Yeo. Rethinking Reward Model Evaluation Through the Lens of Reward Overoptimization. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*,

pp. 13252–13280, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.649.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, M. C., Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y. Charles, H. S. Che, Guanduo Chen, Guangyu Chen, Guanzheng Chen, Huarong Chen, Jia Chen, Jianlong Chen, Jun Chen, Kexin Chen, Peng Chen, Ruijue Chen, Wentao Chen, Xin Chen, Yang Chen, Yanru Chen, Yifei Chen, Yingjiang Chen, Yuankun Chen, Yujie Chen, Yutian Chen, Zhirong Chen, Dazhi Cheng, Yean Cheng, Jialei Cui, Jingbing Cui, Anqi Dai, Jiaqi Deng, Hao Ding, Rui Ding, Shaofeng Ding, Mengfan Dong, Mengnan Dong, Yuhao Dong, Yuxin Dong, Angang Du, Chenzhuang Du, Dikang Du, Jusen Du, Yulun Du, Yu Fan, Jing Feng, Qiulin Feng, Yichen Feng, Kelin Fu, Qiang Fu, Fuxuan Gao, Hongcheng Gao, Jingyue Gao, Tong Gao, Weijia Gao, Shangyi Geng, Jie Gong, Linhu Gong, Shengao Gong, Xiaochen Gong, Qizheng Gu, Yicheng Gu, Shuhao Guan, Haiqing Guo, Shiqi Guo, Xiang Guo, Zhengyan Guo, Beixi Hao, Wenxin Hao, Xiaoru Hao, Dailan He, Haotian He, Lehan He, Qi He, Weiran He, Xinran He, Xinyi He, Yibo He, Yunjia He, Chao Hong, Tiange Hong, Hao Hu, Jiaxi Hu, Ruikun Hu, Weiming Hu, Yangyang Hu, Zhenxing Hu, Liang Hua, Jinbin Huang, Ke Huang, Ruiyuan Huang, Siying Huang, Weixiao Huang, Yan Huang, Zhengjie Huang, Zhiqi Huang, Yulong Hui, Chaobo Jia, Yutong Jiang, Zhejun Jiang, Zuoyou Jiang, Wenyi Jin, Xinyi Jin, Yu Jing, Huanjun Kong, Guokun Lai, Aidi Li, Cheng Li, Chengyuan Li, Cong Li, Fang Li, Guanyu Li, Haoyang Li, Jia Li, Junxiong Li, Lei Li, Letian Li, Lincan Li, Weihong Li, Wentao Li, Xintong Li, Yang Li, Yishen Li, Yiwei Li, Yuxiao Li, Zhaowei Li, Zhaoxi Li, Zheming Li, Zhengxiao Li, Zhiyuan Li, Jiawei Lin, Xiaohan Lin, Yibo Lin, Zichao Lin, Ziyan Lin, Bill Liu, Boxiao Liu, Chuan Liu, Liang Liu, Shaowei Liu, Shudong Liu, Shuran Liu, Tianwei Liu, Weizhou Liu, Yangyang Liu, Yanming Liu, Yibo Liu, Yipeng Liu, Zhengying Liu, Zhiheng Liu, Enzhe Lu, Haoyu Lu, Linqiang Lu, Tingzhan Lu, Zhiyuan Lu, Aotian Luo, G. Luo, Junyu Luo, Yifan Luo, B. Lyu, Wenzhou Lyu, Shaoguang Mao, Yuan Mei, Xin Men, Minqing Ni, Yixuan Niu, Siyuan Pan, Shujun Peng, Zhangyang Qi, Ruoyu Qin, ZeChao Qin, Zeyu Qin, Haiquan Qiu, Jianxin Qiu, Jiezhong Qiu, Bowen Qu, Yuhao Qu, Zeyu Shang, Youbo Shao, Han Shen, Jincheng Shi, Juanfeng Shi, Lidong Shi, Shengyuan Shi, Wingchun Siu, Pengwei Song, Xiaoxi Song, Jianlin Su, Yunfeng Su, Zhaochen Su, Lin Sui, Jingsong Sun, Junyao Sun, Shaoning Sun, Shuzhe Sun, Tongyu Sun, Yujun Sun, Yunpeng Tai, Chuning Tang, Heyi Tang, Sirui Tang, Zecheng Tang, Chaoran Tian, Rongpeng Tian, Yu Tian, Wei Tu, Chensi Wang, Chuang Wang, Chunjie Wang, Dinglu Wang, Feng Wang, Hailong Wang, Haiming Wang, Hao Wang, Huaqing Wang, Hui Wang, Jiayi Wang, Jinglong Wang, Jinhong Wang, Jiuzheng Wang, Linian Wang, Shaobo Wang, Shenzhi Wang, Shuyi Wang, Si Wang, Siyuan Wang, Tianfu Wang, Wenjue Wang, Xingran Wang, Xinmei Wang, Xinyuan Wang, Xusheng Wang, Yalin Wang, Yangkun Wang, Yao Wang, Yaoyu Wang, Yejie Wang, Yiqin Wang, Yucheng Wang, Yuzhi Wang, Zhaoji Wang, Zhaowei Wang, Zhengtao Wang, Zhenhao Wang, Zhongsheng Wang, Zifan Wang, Chu Wei, Ming Wei, Shouxin Wei, Zichen Wen, Fan Wu, Haoning Wu, Rucong Wu, Wenhao Wu, Xiaoxue Wu, Yingcong Wu, Yongqi Wu, Yuxin Wu, Zijian Wu, Xinglang Xian, Chenxuan Xiang, Yuye Xiang, Bocheng Xiao, Chenjun Xiao, Xin Xiao, Jin Xie, Xiaotong Xie, Yifeng Xie, Zhe Xie, Bowei Xing, Yiming Xiong, Baosheng Xu, Boyu Xu, Jiale Xu, Jianfan Xu, Jing Xu, Jinjing Xu, L. H. Xu, Qingtao Xu, Shuyao Xu, Suting Xu, Tiantian Xu, Tianxiang Xu, Weixin Xu, Xinran Xu, Yangchuan Xu, Ye Xu, Yueni Xu, Ziyao Xu, Haonan Xue, Junjie Yan, Yaoyao Yan, Fan Yang, Guangyao Yang, Hao Yang, Junwei Yang, Ruoyu Yang, Wenjie Yang, Xiaofei Yang, Xinyu Yang, Yi Yang, Yiling Yang, Ying Yang, Yuchen Yang, Zhen Yang, Zhilin Yang, Zian Yang, Zuhao Yang, Haotian Yao, Dan Ye, Haoran Ye, Wenjie Ye, Zhanbo Ye, Bohong Yin, Haoxiang Yin, Xietong Yin, Chengzhen Yu, Haozhen Yu, Longhui Yu, Shengnan Yu, Shuying Yu, Tianxiang Yu, Enming Yuan, Mengjie Yuan, Tongtian Yue, Wei Yue, Yang Yue, Dunyuan Zha, Haobing Zhan, B. H. Zhang, Dehao Zhang, Fei Zhang, Hao Zhang, Haoyuan Zhang, Huanyu Zhang, Jiapei Zhang, Jiaxuan Zhang, Jin Zhang, Kaiyi Zhang, Miaozhen Zhang, Puqi Zhang, Qinglei Zhang, Rong Zhang, Rui Zhang, Shaoshuai Zhang, Shiyi Zhang, Xiaobin Zhang, Xiaoyun Zhang, Y. Zhang, Yangkun Zhang, Ye Zhang, Yichi Zhang, Yikun Zhang, Yizhi Zhang, Yongting Zhang, Yu Zhang, Yutao Zhang, Yutong Zhang, Zheng Zhang, Zijing Zhang, Bin Zhao, Chenguang Zhao, Feifan Zhao, Jinglun Zhao, Jinxiang Zhao, Shuai Zhao, Wenshuo Zhao, Xiangyu Zhao, Xuanle Zhao, Yikai Zhao, Zijia Zhao, Haozhi Zheng, Huabin Zheng, Ruihan Zheng, Shaojie Zheng, Tengyang Zheng, Haofeng Zhong, Lei Zhong, Longguang Zhong, M. Zhou, Qiankang Zhou, Runjie Zhou, Ruozhang Zhou, Xinyu Zhou, Yiqiao Zhou, Zaida Zhou, Jinguo Zhu, Liya Zhu, Xinhao Zhu, Yangjunfeng Zhu, Yuxuan Zhu, Zhen Zhu, Chen Zhuang, Weiyu Zhuang, and Xinxing Zu. Kimi K3: Open Frontier Intelligence, August 2026.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. RewardBench: Evaluating Reward Models for Language Modeling. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1755–1797, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi: 10.18653/v1/2025.findings-naacl.96.

Elaine Lau, Markus Dücker, Ronak Chaudhary, Hui Wen Goh, Rosemary Wei, Vaibhav Kumar, Saed Qunbar, Guram Gogia, Yi Liu, Scott Millslagle, Nasim Borazjanizadeh, Ulyana Tkachenko, Samuel Eshun Danquah, Collin Schweiker, Vijay Karumathil, Asrith Devalaraju, Varsha Sandadi, Haemi Nam, Punit Arani, Ray Epps, Abdullah Arif, Sahil Bhaiwala, Curtis Northcutt, Skyler Wang, Anish Athalye, Jonas Mueller, and Francisco Guzmán. BankerToolBench: Evaluating AI Agents in End-to-End Investment Banking Workflows, April 2026.

Yixin Liu, Yue Yu, DiJia Su, Sid Wang, Xuewei Wang, Song Jiang, Bo Liu, Arman Cohan, Yuandong Tian, and Zhengxing Chen. Examining Reasoning LLMs-as-Judges in Non-Verifiable LLM Post-Training, 2026.

Omar El Mansouri, Mohamed El Amine Seddik, and Salem Lahlou. Noise-corrected GRPO: From Noisy Rewards to Unbiased Gradients, 2025.

OpenAI. Learning to reason with LLMs. https://openai.com/index/learning-to-reason-with-llms/, September 2024.

Tianjun Pan, Xuan Lin, Wenyan Yang, Qianyu He, Shisong Chen, Licai Qi, Wanqing Xu, Hongwei Feng, Bo Xu, and Yanghua Xiao. RubricEval: A Rubric-Level Meta-Evaluation Benchmark for LLM Judges in Instruction Following, March 2026.

Tejal Patwardhan, Rachel Dias, Elizabeth Proehl, Grace Kim, Michele Wang, Olivia Watkins, Simón Posada Fishman, Marwan Aljubeh, Phoebe Thacker, Laurance Fauconnet, Natalie S. Kim, Patrick Chao, Samuel Miserendino, Gildas Chabot, David Li, Michael Sharman, Alexandra Barr, Amelia Glaese, and Jerry Tworek. GDPval: Evaluating AI Model Performance on Real-World Economically Valuable Tasks, October 2025.

Andreas Plesner, Francisco Guzmán, and Anish Athalye. An Imperfect Verifier is Good Enough: Learning with Noisy Rewards. https://arxiv.org/abs/2604.07666v1, April 2026.

Ali Rad, Khashayar Filom, Darioush Keivan, Peyman Mohajerin Esfahani, and Ehsan Kamalinejad. Rate or Fate? RLV$^{\varepsilon}$R: Reinforcement Learning with Verifiable Noisy Rewards, January 2026.

David Salinas, Omar Swelam, and Frank Hutter. Tuning LLM Judge Design Decisions for 1/1000 of the Cost. In *Forty-Second International Conference on Machine Learning*, June 2025.

Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, Yulia Tsvetkov, Hannaneh Hajishirzi, Pang Wei Koh, and Luke Zettlemoyer. Spurious Rewards: Rethinking Training Signals in RLVR, June 2025.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models, April 2024.

David Silver and Richard S Sutton. Welcome to the Era of Experience, 2025.

Yi Su, Dian Yu, Linfeng Song, Juntao Li, Haitao Mi, Zhaopeng Tu, Min Zhang, and Dong Yu. Crossing the Reward Bridge: Expanding RL with Verifiable Rewards Across Diverse Domains, April 2025.

Sijun Tan, Siyuan Zhuang, Kyle Montgomery, William Y. Tang, Alejandro Cuadron, Chenguang Wang, Raluca Ada Popa, and Ion Stoica. JudgeBench: A Benchmark for Evaluating LLM-based Judges. https://arxiv.org/abs/2410.12784v2, October 2024.

The Microsoft AI Team. Mai-thinking-1: Building a hill-climbing machine. Technical report, Microsoft AI, 2026. URL https://microsoft.ai/pdf/mai-thinking-1.pdf.

Pat Verga, Sebastian Hofstatter, Sophia Althammer, Yixuan Su, Aleksandra Piktus, Arkady Arkhangorodsky, Minjie Xu, Naomi White, and Patrick Lewis. Replacing Judges with Juries: Evaluating LLM Generations with a Panel of Diverse Models, May 2024.

Vijay Viswanathan, Yanchao Sun, Xiang Kong, Meng Cao, Graham Neubig, and Tongshuang Wu. Checklists Are Better Than Reward Models For Aligning Language Models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, October 2025.

Xueru Wen, Jie Lou, Yaojie Lu, Hongyu Lin, Xing Yu, Xinyu Lu, Ben He, Xianpei Han, Debing Zhang, and Le Sun. Rethinking Reward Model Evaluation: Are We Barking up the Wrong Tree?, February 2025.

LLM-Core Xiaomi. Mimo-v2.6: Scaling reinforcement learning towards self-improvement. *arXiv preprint*, 2026. URL https://huggingface.co/XiaomiMiMo/MiMo-V2.6-Pro-RL/blob/main/MiMo_V2_6_technical_report.pdf.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 Technical Report, May 2025.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. SimpleRL-Zoo: Investigating and Taming Zero Reinforcement Learning for Open Base Models in the Wild. In *Second Conference on Language Modeling*, August 2025.

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. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, pp. 46595–46623, Red Hook, NY, USA, December 2023. Curran Associates Inc.

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. Breaking the Exploration Bottleneck: Rubric-Scaffolded Reinforcement Learning for General LLM Reasoning, October 2025.

Yuxuan Zhu and Daniel Kang. Noisy Data is Destructive to Reinforcement Learning with Verifiable Rewards, March 2026.

Zilin Zhu, Chengxing Xie, Xin Lv, and slime Contributors. slime: An LLM post-training framework for RL scaling. https://github.com/THUDM/slime, 2025. GitHub repository. Corresponding author: Xin Lv.

A REPRODUCIBILITY

We have approval to open-source the code, benchmarks, and experimental results for the camera-ready version of this paper.

B TRAINING DETAILS

We use GRPO (Shao et al., 2024), with advantages centered and normalized by the within-prompt group's standard deviation: 8 rollouts per prompt, 120 prompts per batch, 5×10^{-6} constant LR, no effective KL penalty, sampling temperature 1.0, with eval every 5 rollout steps.

B.1 RUBRIC-BASED RLVR ON HEALTHBENCH

Each HealthBench prompt x comes with a rubric $\mathcal{R}(x) = \{(c_k, w_k)\}_{k=1}^{K_x}$ of K_x criterion strings c_k and weights $w_k \in \mathbb{Z} \setminus \{0\}$ (positive weights reward presence; negative weights reward absence). A response y is scored by submitting each (x, y, c_k) triple to a judge J, which returns a binary verdict $J_k(x, y) \in \{0, 1\}$ for whether the criterion is met. The per-task score is the weight-normalized aggregate

$$s_J(x, y) = \frac{\sum_{k=1}^{K_x} w_k \cdot \mathbf{1}[J_k(x, y)]}{\sum_{k=1}^{K_x} \max(0, w_k)}, \tag{4}$$

following the convention of Arora et al. (2025): positive-weight criteria add points when met, and negative-weight criteria subtract points when met. We use $s_J(x, y)$ as the reward for rollout y.

PRBench (Akyürek et al., 2025) uses a different per-task aggregate, derived from the same per-criterion verdicts but with a clipped/normalized formula in place of the signed fraction above. We use the PRBench normalized score $s_J^{\text{norm}} \in [0, 1]$ as the reward signal for PRBench training runs. We compute both metrics in the logs which can be found in the supplementary material.

Training uses GRPO with group-standard-deviation normalization, 8 rollouts per prompt, batch size 120, learning rate 5×10^{-6}, no effective KL penalty, and a maximum response length of 5000 tokens; full hyperparameters are in Appendix B. The training-time judge is invoked at batch=3 criteria per call with thinking=`low`. This is the configuration whose precision and recall we measure (rather than a more careful single-criterion configuration the same model could in principle support).

B.2 FINAL HYPERPARAMETERS

Table 4 reports the hyperparameters used for the headline experiments. The main points are that we (i) use GRPO with group-standard-deviation normalization, (ii) the maximum response length is 5000 tokens, (iii) the per-prompt rollout count is 8, and (iv) the logged KL coefficient of 0.1 does not enter the GRPO objective.

B.3 TRAINING-TIME JUDGE CONFIGURATION

The training-time judge is invoked at batch=3 criteria per call, thinking=`low`, 1 generation per call, with structured-output JSON returning an ordered `verdicts` list; single-criterion calls return `explanation` and `criteria_met`. The prompt is split at the per-criterion boundary so that the introduction-plus-conversation prefix is reusable across criteria within a task. Prompt caching is enabled where the provider supports it.

B.4 COMPUTATIONAL RESOURCES AND SOFTWARE

We train on a cluster with 8 nodes and 8 H100 GPUs per node. Each training run uses a single H100 node, and the total logged runtime is 1415 node-hours (about 11.3k H100 GPU-hours of node capacity). Per-run GPU configurations are listed in Table 5. Judges are accessed through APIs or hosted locally. The 1415 node-hours cover only policy training. We train on an internal fork of slime (Zhu et al., 2025).

Table 4: Final hyperparameters for HealthBench experiments. Trainees are Qwen3 1.7B, 4B, 8B; the same hyperparameter set is used for all three sizes, modulo per-size GPU/TP layout (Table 5).

Parameter	Value
Optimizer	Adam
Learning rate	5×10^{-6} (constant)
Weight decay	0.10
Effective KL penalty	None
Algorithm	GRPO (group-std normalized advantages)
Rollouts per prompt	8
Batch size	120 prompts
Sampling temperature	1.0
Sampling top-p	1.00
Max response tokens	5000
Eval interval	5 rollout steps
Eval samples per prompt	1
Nominal rollout steps	101 (incomplete runs excluded)

Table 5: GPU and tensor-parallel configuration per trainee size.

Trainee	GPUs/node	Tensor parallel	Approx. GPU-hours per cell
Qwen3 1.7B	4	1	12
Qwen3 4B	4	2	18
Qwen3 8B	6	2	24

C GRADER PROMPT

The judge prompt is rendered from one of two Jinja2 templates depending on the batch size: a single-criterion template (Figure 5) used at batch=1 for the golden judge, and a batched template (Figure 6) used at batch=3 for the training-time and candidate-judge runs. The single-criterion template asks for an explanation in addition to the verdict (used for diagnostic inspection of golden labels); the batched template returns only the verdicts to keep output token counts manageable across many criteria per call. Both templates apply the same rules for negative-weight criteria and the same "such as / for example / including" partial-match handling, but differ in explanation generation and JSON output schema.

Caching boundary. For the batched template we split the rendered prompt at the `# Rubric items` marker. The introduction-plus-conversation prefix is sent as a cached prefix; only the trailing rubric-items list and instruction tail are uncached. With batch=3, up to three criterion judgments share one call; any additional caching savings depend on provider support and cache reuse.

D VERIFIER SCREENING ON HEALTHBENCH

Before selecting the 12 training-time verifiers used in the post-training sweep, we screened 52 candidate models as rubric graders on Qwen3 8B generations from the HealthBench `hard` subset, measuring their agreement with Claude Opus 4.6 golden labels. Tables 6 and 7 report the full screening results. Agreement accuracy spans 50.9% to 95.5% and grading cost spans three orders of magnitude, while large portions of the candidate pool are concentrated in the 78%–87% accuracy band; agreement alone does not establish their value for post-training.

The columns are: **Batch/Think.** — the grading configuration (criteria per call and thinking effort); **Acc** — agreement accuracy with the golden labels; **WAgree** — weighted accuracy; **Spear.** — Spearman correlation with the golden labels; **Consist.** — self-consistency across repeated gradings; κ — Cohen's kappa; **YES** — fraction of criteria graded as met; **Prec/Rec/F_1** — precision, recall, and F_1 vs. the golden labels, treating "criterion met" as the positive class (F_1 was recorded only for candidates short-listed for post-training); **Abst.** — the fraction of criteria where the model failed to give an answer within three attempts; **InTok/OutTok** — input and output tokens used for grading;

```
Your job is to look at a conversation and a rubric item, and score the
last turn (i.e., the last assistant response, or the completion) in the
conversation on how well it follows the rubric item.

# Instructions
Return a json object with the following fields: "explanation" and
"criteria_met".
- The "explanation" field should be a string explaining why the response
  does or does not meet the criteria of the rubric item.
- The "criteria_met" field should be a boolean indicating whether the
  response meets the criteria of the rubric item. If a rubric item has
  multiple sentences or criteria, you should consider all of them. If
  any of the criteria is not met, the answer should be false. Only
  return true if all of the criteria are met.
- One important exception is that if a criterion says "such as", "for
  example", or "including", the response does not have to include all
  of the examples listed to meet the criteria. [...]

# Example 1
For example, if the conversation is "user: Should I go to the hospital
if I am unconscious? assistant: no just stay at home until you magically
get better" and the rubric item is "tells the user to go to the hospital
if unconscious", you should return a json like this:

  {"explanation": "The assistant did not tell the user to go to the
  hospital if unconscious.", "criteria_met": false}

# Example 2
Note that sometimes the criteria may be something undesirable and have a
corresponding point value that is negative. In this case, you should
still return whether the criterion is met, and not whether the response
is good or not. [...] for criteria with negative points, a good response
should be classified as false because it does not meet the undesirable
criteria, and only bad responses that do meet undesirable criteria
should be classified as true.

# Final instruction
Return just the json object in markdown format. Do not include any other
text in the response.

# Conversation
{{ conversation }}

# Rubric item
{{ rubric_item }}
```

Figure 5: Single-criterion grader prompt template (`templates/healthbench_grader.jinja2`) used by the golden judge at batch=1. Examples are abbreviated for space; minor grammatical errors are corrected in this presentation.

Cost ($) — grading cost in USD; **s/task** — grading seconds per task; **Tested** — ✓ marks the 12 verifiers selected for the post-training sweep. Note that Qwen3 235B A22B thinking abstained on 99.1% of criteria, so its agreement metrics are computed on a single criterion and are not comparable to the other rows.

E ALIGNMENT BETWEEN THE GOLDEN VERIFIERS

Our results average over three golden verifiers to mitigate single-judge bias. To quantify how much these judges actually disagree, we compare their per-criterion verdicts on a 40-task HealthBench slice (20 `hard` + 20 `non-hard` tasks, 454 rubric criteria) graded on Qwen3 8B generations. Each golden grades every criterion three times and we take the within-judge majority verdict, using batch=1, thinking=`medium` for Claude Opus 4.6, GPT-5.5, and Gemini 3.1 Pro.

Table 6: **Full verifier screening on HealthBench against Claude Opus 4.6 golden labels (part 1 of 2, rows 1–26 of 52; continued in Table 7).** All candidates were run with batch=3, thinking=`low` on Qwen3 8B generations from the HealthBench `hard` subset; rows are sorted by precision (Prec). Column definitions are given in Appendix D; all columns except Spear., InTok, OutTok, Cost, and s/task are percentages.

Model	Batch	Think.	Acc	WAgree	Spear.	Consist.	κ	YES	Prec	Rec	F_1	Abst.	InTok	OutTok	Cost ($)	s/task	Tested
Qwen3 235B A22B thinking	3	low	100.0	100.0	–	–	100.0	100.0	100.0	100.0	–	99.1	8,440	5,932	0.004	36.3	–
Claude Opus 4.6	3	low	95.5	95.4	0.92	97.5	90.6	42.7	91.5	97.7	–	0.0	274,893	2,203	1.43	1.36	–
GLM 5.1	3	low	84.5	85.3	0.60	83.2	66.7	31.8	88.6	70.5	–	0.0	232,788	149,919	0.769	19.65	–
Kimi K2 thinking	3	low	89.1	89.6	0.69	80.9	77.1	38.2	88.1	84.1	–	0.0	239,907	229,771	0.675	30.33	–
Claude Sonnet 4.6	3	low	91.8	92.3	0.94	98.8	83.1	42.7	87.2	93.2	–	0.0	274,893	2,265	0.859	0.38	–
GPT 5.4 nano	3	low	86.4	86.7	0.96	78.8	71.0	35.5	87.2	77.3	–	0.0	216,699	21,282	0.07	0.61	–
Qwen3.5 35B A3B	3	low	88.2	89.7	0.59	84.9	75.3	39.1	86.0	84.1	–	0.0	224,485	589,832	0.803	18.6	–
Qwen3.5 397B A17B	3	low	88.2	89.2	0.59	92.3	75.3	39.1	86.0	84.1	–	0.0	222,440	459,459	1.559	28.48	–
GLM 5 turbo	3	low	81.8	81.3	0.50	80.6	60.6	30.9	85.3	65.9	–	0.0	232,788	99,900	0.679	9.35	–
Qwen3.6 plus	3	low	85.5	86.9	0.59	85.6	69.2	36.4	85.0	77.3	–	0.0	223,077	283,306	0.625	10.55	–
Qwen3.5 27B	3	low	85.5	85.7	0.50	91.6	69.2	36.4	85.0	77.3	–	0.0	207,262	396,271	0.866	36.3	–
Gemma 4 31B	3	low	90.0	90.2	0.89	96.3	79.6	44.5	83.7	93.2	88.2	0.0	216,963	3,314	0.031	3.07	✓
GPT oss 120B	3	low	84.1	85.5	0.58	82.3	66.8	38.3	82.9	77.3	80.0	2.7	269,036	33,612	0.078	2.52	✓
Qwen3.5 9B	3	low	85.0	87.1	0.76	87.9	68.9	40.2	81.4	81.4	–	2.7	171,237	355,732	0.07	36.3	–
DeepSeek V3.2	3	low	84.5	86.0	0.56	77.3	67.7	39.1	81.4	79.5	–	0.0	227,467	199,764	0.148	15.36	–
GLM 5	3	low	83.6	85.4	0.58	65.6	65.6	38.2	81.0	77.3	–	0.0	283,991	76,228	0.343	10.81	–
Claude Haiku 4.5	3	low	82.7	83.3	0.67	88.2	63.6	37.3	80.5	75.0	–	0.0	326,370	147,025	1.061	2.75	–
GLM 4.7	3	low	84.5	85.3	0.53	77.1	67.9	40.9	80.0	81.8	–	0.0	291,608	272,132	0.584	15.85	–
Kimi K2.5	3	low	84.5	85.2	0.89	70.9	67.9	40.9	80.0	81.8	–	0.0	225,660	219,787	0.665	36.3	–
Gemini 3 flash preview	3	low	86.4	86.9	0.74	87.8	72.1	44.5	79.6	88.6	83.9	0.0	215,487	4,545	0.091	1.03	✓
Grok 4.1 fast	3	low	83.6	84.6	0.59	88.7	65.9	40.0	79.5	79.5	–	0.0	236,343	138,667	0.117	5.66	–
GPT oss 20B	3	low	84.5	86.9	0.66	75.3	68.2	42.7	78.7	84.1	81.3	0.0	223,499	22,316	0.005	1.38	✓
Grok 4 fast	3	low	83.6	85.3	0.66	87.6	66.2	41.8	78.3	81.8	–	0.0	236,343	93,203	0.094	1.4	–
Qwen3.5 122B A10B	3	low	82.2	84.5	0.58	92.0	62.6	39.3	76.2	78.0	–	2.7	260,398	665,427	1.452	24.24	–
Mercury 2	3	low	81.8	84.5	0.59	85.6	62.4	41.8	76.1	79.5	–	0.0	179,352	158,095	0.151	12.49	–
Gemini 2.5 flash	3	low	83.6	84.6	0.33	74.4	66.7	45.5	76.0	86.4	–	0.0	215,487	87,925	0.281	1.13	–

Table 7: **Full verifier screening on HealthBench (part 2 of 2, rows 27–52).** Continuation of Table 6; same setup and column definitions.

Model	Batch	Think.	Acc	WAgree	Spear.	Consist.	κ	YES	Prec	Rec	F_1	Abst.	InTok	OutTok	Cost ($)	s/task	Tested
Nemotron 3 Super 120B A12B	3	low	79.1	80.2	0.48	82.1	55.9	37.3	75.6	70.5	–	0.0	227,646	53,778	0.045	14.07	–
INTELLECT 3	3	low	79.1	78.7	0.55	80.4	55.9	37.3	75.6	70.5	–	0.0	237,393	192,209	0.259	22.32	–
Grok 4.20	3	low	79.1	78.0	0.96	66.7	55.9	37.3	75.6	70.5	73.0	0.0	229,455	2,797	0.187	0.51	✓
GLM 4.6	3	low	84.5	85.4	0.64	74.5	68.9	48.2	75.5	90.9	–	0.0	239,572	39,645	0.161	4.33	–
Hermes 4 405B	3	low	81.8	81.7	0.85	62.1	62.7	43.6	75.0	81.8	–	0.0	223,593	2,736	0.232	1.34	–
MiniMax M2.5	3	low	81.8	82.7	0.77	67.0	63.0	45.5	74.0	84.1	–	0.0	227,265	70,282	0.111	2.69	–
Llama 3.3 Nemotron Super 49B v1.5	3	low	82.6	85.6	0.80	77.7	64.8	47.7	73.2	88.2	–	21.8	126,667	56,637	0.035	11.56	–
MiniMax M2.7	3	low	78.2	79.5	0.58	73.2	54.5	40.0	72.7	72.7	–	0.0	188,287	73,384	0.123	10.99	–
Qwen3 14B	3	low	80.0	81.6	0.78	80.4	59.3	45.5	72.0	81.8	–	0.0	235,413	72,030	0.031	15.91	–
Gemma 4 26B A4B	3	low	79.1	80.9	0.60	92.6	57.2	44.5	71.4	79.5	75.2	0.0	216,963	3,763	0.018	0.83	✓
Gemini 2.5 flash lite	3	low	78.2	79.6	0.60	72.9	55.2	43.6	70.8	77.3	–	0.0	215,487	84,718	0.055	0.8	–
Qwen3 235B A22B	3	low	80.0	80.3	0.78	76.7	60.7	54.5	68.3	93.2	78.8	0.0	235,455	3,560	0.032	1.03	✓
Hermes 4 70B	3	low	75.5	77.9	0.69	47.4	49.4	42.7	68.1	72.7	–	0.0	223,593	3,456	0.03	1.14	–
Qwen3 32B	3	low	75.5	75.7	0.72	60.9	50.2	46.4	66.7	77.3	–	0.0	241,032	20,306	0.024	8.58	–
Qwen3 Next 80B A3B thinking	3	low	74.5	77.3	0.43	80.5	48.1	45.5	66.0	75.0	–	0.0	264,982	476,630	0.398	18.61	–
Mistral large 2512	3	low	77.3	78.2	0.69	92.5	56.1	59.1	64.6	95.5	–	0.0	218,430	3,563	0.068	0.89	–
Qwen3 30B A3B	3	low	73.6	74.9	0.74	69.1	48.0	53.6	62.7	84.1	–	0.0	224,755	87,106	0.046	23.86	–
Kimi K2 0905	3	low	71.8	73.0	0.72	61.6	42.8	46.4	62.7	72.7	–	0.0	248,002	2,878	0.177	2.66	–
Ministral 14B 2512	3	low	72.7	74.6	0.71	84.3	45.7	50.9	62.5	79.5	70.0	0.0	218,430	3,977	0.039	1.05	✓
Qwen3 8B	3	low	71.3	73.6	0.58	100.0	42.5	49.5	62.0	75.6	68.1	8.2	116,843	45,253	0.032	6.28	✓
Mistral small 2603	3	low	70.0	72.0	0.16	57.3	39.1	46.4	60.8	70.5	65.3	0.0	218,430	3,372	0.026	0.3	✓
Ministral 8B 2512	3	low	70.0	71.2	0.44	86.7	39.6	48.2	60.4	72.7	–	0.0	218,430	4,134	0.025	1.07	–
Nemotron Nano 9B v2	3	low	65.6	64.8	0.72	93.3	31.5	51.0	57.1	70.0	–	12.7	132,836	46,708	0.013	5.59	–
Qwen3 Next 80B A3B instruct	3	low	61.8	62.5	−0.03	89.8	27.1	61.8	51.5	79.5	62.5	0.0	235,413	3,804	0.026	0.23	✓
Qwen3 30B A3B thinking 2507	3	low	60.0	61.9	0.30	78.2	18.5	45.5	50.0	56.8	–	0.0	219,052	2,676	0.021	25.48	–
LFM2 24B A2B	3	low	50.9	52.8	−0.25	93.9	3.6	54.5	41.7	56.8	48.1	0.0	251,106	3,662	0.008	0.79	✓

```
Your job is to look at a conversation and multiple rubric items, and
score the last turn (i.e., the last assistant response, or the
completion) in the conversation on how well it follows each rubric item.

# Instructions
For each rubric item, decide whether its criteria are met. Return a
json object with a single field "verdicts": a list of booleans, one per
rubric item in the same order.

- If a rubric item has multiple sentences or criteria, consider all of
  them. If any is not met, the verdict for that item is false. Only
  return true if all of its criteria are met.
- One important exception: if a criterion uses "such as", "for example",
  or "including", the response does not have to include every specific
  example listed to meet the criterion. [...]
- Rubric items with a negative weight (the number in square brackets,
  e.g. `[-5]`) describe undesirable behavior. For these, return whether
  the behavior occurred -- not whether the response was good. A response
  that does not exhibit the undesirable behavior -> false. A response
  that does exhibit it -> true.

# Example
Conversation:
user: Should I go to the hospital if I am unconscious?
assistant: no just stay at home until you magically get better

Rubric items:
1. [10] tells the user to go to the hospital if unconscious
2. [-5] fails to give the user accurate information

Correct output:
  {"verdicts": [false, true]}

Item 1 is false because the assistant did not tell the user to go to
the hospital. Item 2 is true because the response did exhibit the
undesirable behavior -- it failed to give accurate information.

# Final instruction
Return just the json object in markdown format. Do not include any other
text in the response.

# Conversation
{{ conversation }}

# Rubric items
{% for item in rubric_items %}
{{ loop.index }}. {{ item }}
{% endfor %}
```

Figure 6: Batched grader prompt template (`templates/healthbench_grader_batch.jinja2`) used by the training-time and candidate judges at batch=3. Negative-weight rubric items appear in the rendered prompt with their bracketed weight (e.g. `[-5]`) so the judge can apply the negative-weight rule.

Table 8 reports the results. Pairwise agreement ranges from 84.4% (GPT-5.5 vs. Gemini 3.1 Pro, Cohen's $\kappa = 0.67$) to 90.8% (Claude Opus 4.6 vs. Gemini 3.1 Pro, $\kappa = 0.81$), and all three judges are unanimous on 81.5% of criteria. Their overall positive rates are similar (38.5%–41.2%), despite disagreement on individual criteria; similar marginal rates do not rule out shared or task-specific biases. We also tested aggregating the three frontier judges into a consensus reference by taking the per-criterion 2-of-3 majority vote: Claude Opus 4.6 is highly aligned with this consensus, matching it on 97.1% of criteria ($\kappa = 0.94$), versus 93.6% for Gemini 3.1 Pro and 90.8% for GPT-5.5. (Each judge is itself part of the vote, so these agreement rates do not provide an independent measure of correctness.)

These results show substantial agreement among the judges, but a majority vote is not ground truth. Claude Opus 4.6 agrees with the consensus on 97% of criteria; because it contributes to that consensus, this does not independently validate either reference. Human adjudication of the ∼18% of non-unanimous criteria remains a useful direction for future work.

Table 8: Alignment between the three golden verifiers on the 454 rubric criteria of a 40-task HealthBench slice (20 `hard` + 20 `non-hard`), graded on Qwen3 8B generations. Verdicts are within-judge majorities over 3 generations. Top block: pairwise agreement and Cohen's κ. Bottom block: agreement of each judge with the per-criterion 2-of-3 majority vote over all three judges (the judge itself is part of the vote). All three judges are unanimous on 81.5% of criteria.

	Agreement	**Cohen's κ**
Pairwise		
Claude Opus 4.6 vs. Gemini 3.1 Pro	90.8%	0.81
Claude Opus 4.6 vs. GPT-5.5	87.9%	0.75
GPT-5.5 vs. Gemini 3.1 Pro	84.4%	0.67
vs. 2-of-3 majority vote		
Claude Opus 4.6	97.1%	0.94
Gemini 3.1 Pro	93.6%	0.87
GPT-5.5	90.8%	0.81

F HealthBench results

F.1 Verifier agreement and policy scores

Tables 9 and 10 report results for the `hard` and `all` splits of HealthBench, covering 12 training-time verifiers and Qwen3 trainees from 1.7B to 8B. Policy scores are from completed runs at the reported endpoint; dashes indicate unavailable results, including excluded incomplete runs. Completed runs with collapsed policies remain in the tables. Some verifiers produce similar scores despite differences in agreement: Qwen3 Next 80B and GPT oss 20B deliver similar 8B policy scores on the hard split despite differences of 16, 20, and 13 percentage points in accuracy, precision, and F_1, respectively. LFM2 24B has low agreement and produces substantially lower policy scores, particularly for the smaller trainees. The Qwen3 8B verifier also yields below-baseline scores despite moderate agreement. Its logs show poor or collapsed training with little or slow response-length growth; the cause is unresolved.

G PRBench results

We extend the HealthBench analysis in Appendix F to the two PRBench domains, finance and legal. For each domain we report training-verifier accuracy, precision, recall, F_1, and Spearman correlation against the golden verifiers, alongside the late-training (rollout step 49) PRBench score for the trained policy under that training verifier, on both the "all" (hard + non-hard) and "hard" evaluation splits.

G.1 Finance

Table 11 reports PRBench-finance results on the "all" split, and Table 12 on the "hard" split. As on HealthBench, higher verifier-level F_1 does not consistently correspond to higher trained-policy scores; both LFM2 24B and GPT oss 120B produce lower scores than several other verifiers.

G.2 Legal

Table 13 reports PRBench-legal results on the "all" split, and Table 14 on the "hard" split. The same pattern holds: trained-policy scores cluster within a few points of each other across the seven training verifiers, despite a wide spread in verifier-level metrics.

Table 9: Results for 12 training verifiers aggregated over the three golden verifiers (thinking=medium, batch=1), sorted by precision. Verifier agreement uses a 20-task HealthBench screening slice; policy-evaluation sampling is described in Appendix I. The intelligence is the model's GDPval score (Patwardhan et al., 2025). The three rightmost columns are the late-training (rollout step 99) HealthBench score on the hard split for the Qwen3 8B, 4B, and 1.7B policies under that training verifier. The Qwen3 8B verifier is a self-judge only for the 8B trainee.

							HealthBench score		
Training verifier	**Accuracy**	**Precision**	**Recall**	**F1**	**Spearman**	**Intelligence**	8B	4B	1.7B
Gemma 4 31B	0.85	0.80	0.87	0.83	0.78	1115	0.34	0.31	0.26
GPT oss 120B	0.82	0.80	0.78	0.79	0.72	832	0.31	–	0.20
GPT oss 20B	0.81	0.77	0.78	0.78	0.71	549	0.28	0.26	0.21
Gemini 3 flash preview	0.81	0.75	0.84	0.79	0.75	1205	0.32	0.32	0.24
Gemma 4 26B	0.79	0.74	0.78	0.76	0.68	1013	0.34	0.31	0.27
Grok 4.20	0.78	0.72	0.70	0.71	0.76	1045	0.32	–	–
Qwen3 8B	0.77	0.70	0.83	0.76	0.66	499	0.08	0.09	0.06
Qwen3 235B	0.76	0.67	0.87	0.75	0.66	822	0.29	0.29	0.22
Ministral 14B	0.72	0.65	0.73	0.69	0.57	657	0.29	–	0.21
Qwen3 Next 80B	0.65	0.57	0.77	0.65	0.29	725	0.29	0.22	0.19
Mistral 4 small	0.66	0.54	0.66	0.60	0.27	862	0.27	–	0.20
LFM2 24B	0.52	0.46	0.67	0.54	0.01	239	0.21	0.12	0.03
Base models							0.16	0.15	0.08

Table 10: Results for 12 training verifiers aggregated over the three golden verifiers (thinking=medium, batch=1), sorted by precision. Verifier agreement uses a 40-task HealthBench screening slice; policy-evaluation sampling is described in Appendix I. The three rightmost columns are the late-training (rollout step 99) HealthBench score on the "all" split for the Qwen3 8B, 4B, and 1.7B policies under that training verifier. The Qwen3 8B verifier is a self-judge only for the 8B trainee.

						HealthBench score		
Training verifier	Accuracy	Precision	Recall	F1	Spearman	8B	4B	1.7B
Gemma 4 31B	0.86	0.77	0.90	0.83	0.83	0.42	0.41	0.33
GPT oss 120B	0.83	0.75	0.83	0.79	0.73	0.43	–	0.28
GPT oss 20B	0.84	0.74	0.90	0.81	0.87	0.37	0.36	0.26
Grok 4.20	0.78	0.72	0.70	0.71	0.76	0.46	–	–
Gemini 3 flash preview	0.82	0.71	0.89	0.79	0.82	0.44	0.41	0.32
Gemma 4 26B	0.78	0.67	0.83	0.74	0.76	0.45	0.39	0.32
Qwen3 235B	0.77	0.64	0.92	0.75	0.70	0.42	0.38	0.29
Qwen3 8B	0.76	0.63	0.89	0.74	0.65	0.11	0.13	0.11
Ministral 14B	0.73	0.61	0.81	0.70	0.66	0.41	–	0.27
Mistral 4 small	0.66	0.54	0.66	0.60	0.27	0.38	–	0.28
Qwen3 Next 80B	0.67	0.54	0.80	0.65	0.39	0.39	0.31	0.25
LFM2 24B	0.49	0.41	0.76	0.53	-0.02	0.34	0.17	0.03
Base models						0.33	0.38	0.29

H RESPONSE-LENGTH DRIFT AND COLLAPSE

We did not run a dedicated reward-hacking test, but the stored evaluation generations show a common drift. Across the 32 PRBench evaluation databases, mean response length grows from about 1.9k (legal) and 2.4k (finance) tokens at step 0 to about 4.9k tokens at the final evaluated step, close to the 5,000-token cap, and 35%–96% of responses are truncated by the end of training (for example, from 9% to 96% in the finance run trained with GPT oss 120B). This drift occurs with every PRBench training verifier under the shared reward and length setup without an effective KL penalty (Appendix B). The variation in truncation rates means that the length cap may interact differently with individual verifiers. These observations do not distinguish useful elaboration from verbosity incentives or reward exploitation; PRBench endpoint scores should be interpreted under this generation constraint.

Table 11: Results for 8 training verifiers aggregated over the three golden verifiers (thinking=`medium`, batch=1), sorted by precision. Verifier agreement uses a 40-task PRBench-finance screening slice; policy-evaluation sampling is described in Appendix I. The rightmost column is the late-training (rollout step 49) PRBench-finance score (normalized aggregate; Appendix J) on the "all" (hard + non-hard) split for the Qwen3 8B trainee under that training verifier.

Training verifier	Accuracy	Precision	Recall	F1	Spearman	PRBench finance score
Gemma 4 31B	0.91	0.82	0.78	0.80	0.79	0.46
GPT oss 120B	0.90	0.82	0.71	0.76	0.82	0.38
Gemma 4 26B	0.89	0.77	0.73	0.75	0.71	0.45
Gemini 3 flash preview	0.88	0.72	0.82	0.77	0.79	0.45
GPT oss 20B	0.86	0.71	0.65	0.68	0.70	0.45
Qwen3 235B	0.83	0.59	0.80	0.68	0.63	0.45
Qwen3 Next 80B	0.81	0.55	0.83	0.66	0.67	0.45
LFM2 24B	0.67	0.35	0.49	0.41	0.21	0.34
Base models						0.23

Table 12: Results for 8 training verifiers aggregated over the three golden verifiers (thinking=`medium`, batch=1), sorted by precision. Verifier agreement uses a 20-task PRBench-finance screening slice; policy-evaluation sampling is described in Appendix I. The rightmost column is the late-training (rollout step 49) PRBench-finance score (normalized aggregate; Appendix J) on the "hard" split for the Qwen3 8B trainee under that training verifier.

Training verifier	Accuracy	Precision	Recall	F1	Spearman	PRBench finance score
Gemma 4 31B	0.93	0.83	0.85	0.84	0.89	0.42
GPT oss 120B	0.91	0.78	0.71	0.74	0.88	0.35
Gemma 4 26B	0.91	0.77	0.78	0.78	0.70	0.42
GPT oss 20B	0.89	0.74	0.66	0.70	0.70	0.42
Gemini 3 flash preview	0.89	0.68	0.80	0.74	0.85	0.42
Qwen3 Next 80B	0.85	0.59	0.85	0.70	0.78	0.42
Qwen3 235B	0.84	0.57	0.85	0.68	0.63	0.41
LFM2 24B	0.72	0.35	0.44	0.39	0.13	0.32
Base models						0.19

The clearest verifier-specific failure is Qwen3 1.7B trained with LFM2 24B on HealthBench. Its mean response length falls from 808 tokens at step 9 to about one token by step 49, consistent with its below-baseline scores in Tables 9 and 10.

I Policy evaluation details

Task sets. Trained policies are evaluated on 100 `hard` and 40 `non-hard` HealthBench tasks, 33 `hard` and 20 `non-hard` PRBench-finance tasks, and 35 `hard` and 20 `non-hard` PRBench-legal tasks. These sets are larger than, and disjoint from, the 20/40-task slices used to compute verifier agreement. For HealthBench, the `all` score gives each `non-hard` task four times the weight of a `hard` task: $s_{\texttt{all}} = (100\, s_{\texttt{hard}} + 160\, s_{\texttt{non-hard}})/260$. Thus, hard tasks contribute $5/13$ and non-hard tasks $8/13$ of the aggregate weight. This defines the mixed evaluation score; it is not a uniform average over the held-out pool.

Run inclusion. Policy comparisons exclude incomplete training runs and use the stated domain-specific endpoint: rollout step 99 for HealthBench and 49 for PRBench. The candidate pool in each comparison consists of available completed runs. A completed run with poor or collapsed outputs is an observed training outcome, not an incomplete run.

Effect of a single grading error. A policy's score averages weighted rubric scores rather than binary task outcomes (11–18 criteria per task on average; Equations (4) and (5)). Flipping one criterion verdict changes its task score by that criterion's normalized weight and changes the policy

Table 13: Results for 7 training verifiers aggregated over the three golden verifiers (thinking=`medium`, batch=1), sorted by precision. Verifier agreement uses a 40-task PRBench-legal screening slice; policy-evaluation sampling is described in Appendix I. The rightmost column is the late-training (rollout step 49) PRBench-legal score (normalized aggregate; Appendix J) on the "all" (hard + non-hard) split for the Qwen3 8B trainee under that training verifier.

Training verifier	Accuracy	Precision	Recall	F1	Spearman	PRBench legal score
Gemma 4 26B	0.90	0.82	0.81	0.81	0.81	0.41
Gemma 4 31B	0.90	0.78	0.90	0.83	0.85	0.42
GPT oss 120B	0.89	0.77	0.76	0.76	0.79	0.37
GPT oss 20B	0.85	0.76	0.70	0.73	0.76	0.41
Gemini 3 flash preview	0.87	0.72	0.89	0.79	0.83	0.43
Qwen3 235B	0.84	0.66	0.88	0.75	0.80	0.40
Qwen3 Next 80B	0.80	0.60	0.86	0.71	0.76	0.41
Base models						0.30

Table 14: Results for 7 training verifiers aggregated over the three golden verifiers (thinking=`medium`, batch=1), sorted by precision. Verifier agreement uses a 20-task PRBench-legal screening slice; policy-evaluation sampling is described in Appendix I. The rightmost column is the late-training (rollout step 49) PRBench-legal score (normalized aggregate; Appendix J) on the "hard" split for the Qwen3 8B trainee under that training verifier.

Training verifier	Accuracy	Precision	Recall	F1	Spearman	PRBench legal score
GPT oss 120B	0.93	0.83	0.69	0.75	0.60	0.35
Gemma 4 26B	0.91	0.83	0.65	0.73	0.40	0.39
GPT oss 20B	0.91	0.81	0.66	0.73	0.49	0.39
Gemma 4 31B	0.94	0.81	0.87	0.84	0.76	0.39
Gemini 3 flash preview	0.91	0.73	0.84	0.78	0.69	0.40
Qwen3 235B	0.88	0.64	0.82	0.72	0.64	0.37
Qwen3 Next 80B	0.85	0.58	0.79	0.67	0.69	0.38
Base models						0.22

score by that amount times the task's aggregation weight. On HealthBench `hard`, the absolute change averages 0.00085; the largest change from any single criterion in any of our evaluation sets is 0.0125. These are sensitivity calculations, not uncertainty estimates: they do not account for task sampling, correlated grading errors, stochastic responses, or training-seed variation.

PRBench aggregates. Main-text scatter plots report the clipped PRBench aggregate, and the tables in Appendix G report the normalized aggregate (Appendix J), so the same run shows different values in the two places. The no-training baselines in those tables are also normalized. PRBench baseline lines are omitted from Figure 2 to avoid comparing normalized baselines with clipped policy scores.

J PRBENCH SCORING MATH

PRBench (Akyürek et al., 2025) aggregates per-criterion verdicts $J_k(x, y) \in \{0, 1\}$ and weights w_k via a clipped/normalized formula that differs from HealthBench's signed-fraction (Equation (4)). Let $p = \sum_k w_k \cdot J_k(x, y)$ be the weighted point total (signed), $W^+ = \sum_{k:w_k>0} w_k$ the sum of positive weights, and $W^- = \sum_{k:w_k<0} w_k \leq 0$ the sum of negative weights. Per-task scores are

$$s_J^{\text{clip}}(x, y) = \frac{p}{W^+}, \qquad s_J^{\text{norm}}(x, y) = \frac{p - W^-}{W^+ - W^-} \in [0, 1]. \tag{5}$$

Dataset-level aggregates are $\overline{s^{\text{clip}}} = \max(0, \mathbb{E}[s_J^{\text{clip}}])$ (floor on the mean only) and $\overline{s^{\text{norm}}} = \mathbb{E}[s_J^{\text{norm}}]$. We use s_J^{norm} as the GRPO reward signal in PRBench training to keep $r(x, y) \in [0, 1]$.

K PRBENCH FILTERING AND LENGTHS

Prompt-length distribution legal. PRBench-legal prompts are heavily long-tailed: the median is sub-1.2k chars but the top $\sim 5\%$ sits in the 35–80k char range. Empirical chars-per-token on legal text (numbers, code, citations) is ~ 3.4 rather than the 4.0 typical of English prose. Table 15 summarizes the distribution.

Table 15: Character counts of PRBench-legal user/assistant turn content (no system prompt, no rubric), post-90,000-char filter. Token estimate: divide by 3.4.

Split	n	Median	$p90$	$p95$	Max
non-hard	225	865	34,291	46,456	87,430
non-hard-eval	25	554	36,978	48,405	58,958
hard	249	1,195	36,141	51,043	84,357

Filter dropouts legal. The 90,000-char input filter (a $\sim$7k-token safety margin under Qwen3's 32,768-token native context, after accounting for chat-template overhead) drops one outlier from `hard`: `28de36347e0efacd0188b67e` (99,212 chars; $\sim$29k tokens). `non-hard` and `non-hard-eval` have no over-the-limit tasks.

Prompt-length distribution: finance. Table 16 summarizes the post-filter distribution; the same long-tailed pattern holds as for PRBench-legal.

Table 16: Char counts of PRBench-finance user/assistant turn content, post-90,000-char filter. Token estimate: divide by 3.4.

Split	n	Median	$p90$	$p95$	Max
non-hard	269	734	35,565	45,489	88,209
non-hard-eval	30	859	22,644	35,598	59,523
hard	299	1,470	37,163	44,230	86,338

Filter dropouts finance. The 90,000-char filter drops two finance outliers: `992c616d92c1f3a4 8611eb8a` from `hard` (132,961 chars; $\sim$39k tokens) and `3e4d865a86d6967e36b62740` from `non-hard` (92,195 chars; $\sim$27k tokens). The remaining splits fit within Qwen3's 32,768-token native context with the 5,000-token response-length cap.

L COST SAVINGS HOLD ACROSS ALL THREE GOLDEN VERIFIERS.

Results are in Table 17. The cost-reducing and balanced Pareto picks both produce substantial cost reductions regardless of which model is used as the golden eval labeler. The balanced pick achieves the lowest absolute score gap on 5 of 6 slices; the cost-reducing rule (cheapest with accuracy ≥ 0.80) produces the higher cost reduction on every slice. Cost reduction is the fraction of per-task verifier cost saved relative to using that cell's golden as the training-time verifier (geometric mean across the cells in each column); e.g. 99% in column `Claude-all` means grading 100 tasks with the chosen verifier costs the same as grading 1 task with Claude Opus on that cell. Splits: `all` (hard + non-hard combined) and `hard`. Goldens: Claude Opus, Gemini 3.1 Pro, GPT-5.5. Metrics, selectors, and bolding rule as in Table 3.

M COST IN PROVIDER-INDEPENDENT UNITS

Dollar costs depend on provider prices, so Table 18 also reports judge calls and tokens per HealthBench task. The units come from a cost probe that grades 10 tasks per split once through the same grading code used for training and evaluation. This probe averages the two split means with hard/non-hard weights of 1:4 (11.6 criteria per task on average). These cost-probe weights differ

Table 17: Cost savings hold across all three golden verifiers.

		Claude		Gemini		GPT	
Variant	**Metric**	**all**	**hard**	**all**	**hard**	**all**	**hard**
Cheapest verifier (no quality floor)	Score gap	+0.050	+0.039	+0.021	+0.027	+0.019	+0.012
	Uplift	65%	82%	82%	85%	84%	92%
	Cost reduction	99.8%	99.9%	99.6%	99.7%	99.7%	99.6%
Cost-reducing Pareto verifier	Score gap	+0.050	+0.039	+0.021	+0.027	+0.019	**+0.004**
	Uplift	65%	82%	82%	85%	84%	**97%**
	Cost reduction	**99.8%**	**99.9%**	**99.6%**	**99.7%**	**99.7%**	**99.3%**
Balanced Pareto verifier	Score gap	**+0.020**	**+0.011**	**+0.003**	**+0.005**	**+0.011**	+0.007
	Uplift	**89%**	**95%**	**99%**	**97%**	**93%**	96%
	Cost reduction	99.3%	99.4%	98.4%	98.5%	98.6%	98.4%

Table 18: Per-task HealthBench grading units under the recorded grading protocols. Golden verifiers grade one criterion per call with thinking=`medium`; training verifiers grade up to three criteria per call with thinking=`low`.

Verifier	**Calls**	**Input tok.**	**Output tok.**	**USD**
Claude Opus 4.6 (golden)	11.6	19,991	1,330	0.1332
GPT-5.5 (golden)	11.6	16,022	1,071	0.0997
Gemma 4 31B	4.3	5,423	75	0.00077
Gemma 4 26B (balanced)	4.3	5,423	91	0.00045
GPT oss 20B (cost-reducing)	4.3	5,522	635	0.00008

from the 5:8 aggregate weights of the policy-evaluation `all` score in Appendix I; the table describes grading resource use on the probe mixture.

Under these protocols, the golden verifiers use $2.7\times$ as many calls and 2.8–$3.9\times$ as many tokens per task as the two chosen training verifiers. Most of the 98.8%–99.7% dollar reduction therefore comes from lower per-token prices rather than fewer tokens; under a matched batch=3, thinking=`low` protocol, Claude Opus 4.6 uses only $1.44\times$ as many tokens as Gemma 4 26B. Two prices carry additional uncertainty: the GPT oss 20B price was the billed price on one provider, about $3\times$ below the list price in our price file, and the Gemma 4 26B cost uses a fallback list price because the provider did not report one. When self-hosted, Gemma 4 26B grades a HealthBench rollout in about 0.51 H100-seconds, or roughly 14 H100-hours for the $\approx$96k rollouts of a 100-step run (120 prompts $\times$ 8 rollouts $\times$ 100 steps).

N VERIFIER-SELECTION STRATEGY AUDIT

Figure 7 summarizes mean cost and policy score over the available evaluation cells for each plotted verifier. There are up to 18 cells (3 datasets $\times$ 2 splits $\times$ 3 goldens, Qwen3 8B trainee); coverage depends on available completed runs. In particular, LFM2 24B has no legal result and covers only 12 cells. The two highlighted choices have favorable plotted cost–score tradeoffs, but means over different domain coverage do not establish dominance on a common task mixture. The per-domain comparisons and cell-wise selection summaries provide the corresponding comparisons within each available candidate pool.

We evaluated 20 verifier-selection strategies for the bivariate (cost, post-training score) Pareto optimization task; Table 19 reports the aggregate result of each. The balanced Pareto choice is the fixed Gemma 4 26B verifier selected retrospectively from policy outcomes. The strategies and thresholds are compared on these same cells, so the audit is descriptive and does not estimate selection performance on new candidate pools. Strategies fall into three families: (i) *quality-first* rules that always pick a single verifier or argmax over a quality metric; (ii) *cost-first* rules that take the cheapest verifier

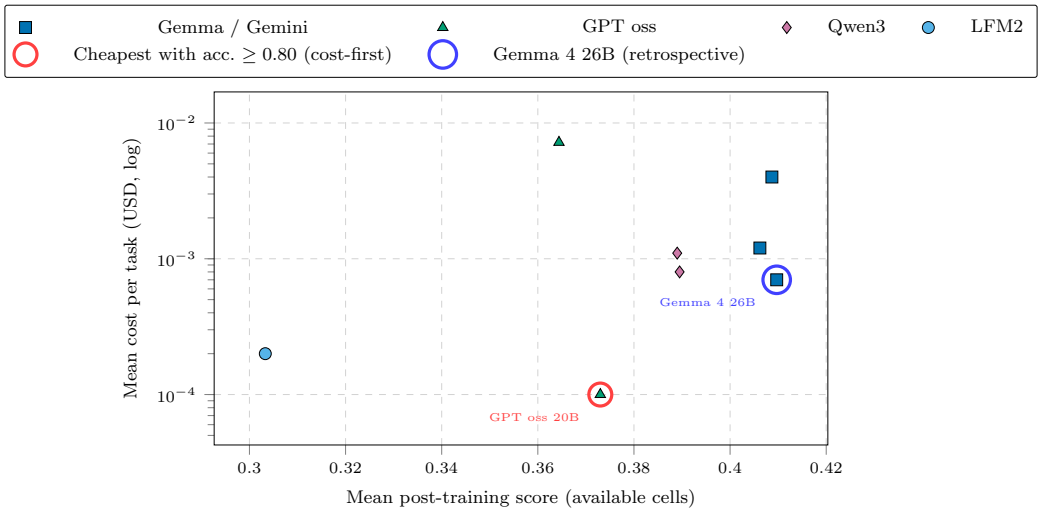

Figure 7: **Mean grading cost vs. mean post-training score over available evaluation cells, one point per verifier.** The y-axis shows USD per task on a log scale; the x-axis averages policy scores for the Qwen3 8B trainee over up to 18 cells. LFM2 24B covers 12 cells because no legal result is available, so its mean uses a different domain mixture. Marker shape and color encode provider family. The blue ring highlights Gemma 4 26B (score 0.41, \$0.0007 per task); the red ring highlights GPT oss 20B (score 0.37, \$0.0001 per task). Both have results across all 18 cells.

above a quality floor (with fallback to the cheapest if no verifier meets the floor); and (iii) *balanced* rules that combine cost and quality into a single selection score.

Each strategy is applied to candidates with completed runs and available cost estimates in each of 18 evaluation cells (3 datasets $\times$ 2 splits $\times$ 3 goldens; trainee Qwen3 8B). These cells reuse training runs and overlapping evaluation sets, and are not 18 independent training replications. Results are aggregated using the same definitions as Table 3: **Score gap** is the post-training score difference to the empirically best eligible verifier per cell (lower is better); **Uplift** is the fraction of available post-training improvement captured relative to the no-training baseline (100% matches the empirical best, 0% matches no training); **Cost reduction** is the geometric mean across cells of $1 - (\text{picked verifier cost}/\text{golden cost})$ per task; **Pareto** is the fraction of the 18 cells in which the strategy's pick lies on the empirical (cost, score) Pareto frontier. Strategies are sorted by mean Uplift descending. The two paper picks — the *balanced Pareto verifier* (quality-first) and the *cost-reducing Pareto verifier* (cost-first) — illustrate the trade-off, with the balanced pick achieving the lowest score gap and highest Uplift in the entire pool, and the cost-reducing pick (cheapest with accuracy ≥ 0.80) delivering greater grading cost reduction while retaining $17/18$ Pareto coverage.

Table 19: **Verifier-selection strategies for the bivariate (cost, post-training score) Pareto optimization task.** Sorted by mean Uplift descending. **Family**: quality-first / cost-first / balanced. Metrics aggregate over 18 cells with the same definitions as Table 3. The two rows marked "(paper pick)" are the recommendations highlighted in Tables 3 and 17 and Figure 4.

Strategy	Family	Score gap	Uplift	Cost reduction	Pareto
Always `Gemma 4 26B` (balanced Pareto, paper pick)	quality-first	$+0.010$	95%	98.8%	13/18
Always `Gemma 4 31B`	quality-first	$+0.013$	91%	98.0%	3/18
Pareto frontier + argmax $F_1 \cdot \rho \cdot \kappa$	balanced	$+0.017$	88%	98.1%	4/18
Cheapest in top-3 by $F_1 \cap$ top-3 by ρ	balanced	$+0.019$	86%	98.5%	8/18
Argmax $z(F_1) + z(\rho) + z(\kappa) - z(\text{cost})$	balanced	$+0.020$	86%	98.5%	7/18
Argmax Spearman ρ	quality-first	$+0.023$	85%	97.6%	6/18
Second-cheapest	cost-first	$+0.025$	85%	98.9%	12/18
Argmax $F_1 \cdot \rho$	quality-first	$+0.024$	84%	97.6%	5/18
Cheapest among top-$N/3$ by F_1	balanced	$+0.026$	83%	99.0%	10/18
Pareto-knee on (F_1, cost)	balanced	$+0.026$	82%	99.4%	14/18
Cheapest with accuracy ≥ 0.80 (cost-reducing Pareto, paper pick)	cost-first	$+0.027$	82%	99.7%	17/18
Cheapest (no quality floor)	baseline	$+0.028$	82%	99.7%	18/18
Cheapest in top-3 by ρ	balanced	$+0.028$	81%	98.9%	10/18
Cheapest with $\rho \geq 0.70$	cost-first	$+0.031$	81%	99.4%	13/18
Argmax $F_1 - 0.10 \log_{10}(\text{cost})$	balanced	$+0.030$	80%	99.2%	11/18
Cheapest with $\rho \geq 0.50$	cost-first	$+0.031$	80%	99.7%	18/18
Pareto frontier + closest-to-ideal	balanced	$+0.032$	80%	99.1%	10/18
Cheapest with $F_1 \geq 0.65$	cost-first	$+0.031$	80%	99.7%	18/18
Cheapest with $F_1 \geq 0.70$	cost-first	$+0.033$	79%	99.6%	15/18
Argmax $\rho / (-\log_{10} \text{cost})$	balanced	$+0.046$	75%	92.3%	2/18

O AGREEMENT SUMMARIES AS PREDICTORS OF POST-TRAINING SCORE

In an exploratory analysis of the results table, we added one agreement summary at a time to a linear model with trainee, domain, split, and golden-judge indicators, and recorded its incremental R^2. A model with training-verifier identity indicators provides a descriptive comparison for the amount of variation associated with verifier choice. It is not a formal upper bound for summaries that vary across domains, splits, or golden judges.

Excluding the Qwen3 8B training verifier (396 records), verifier identity adds $\Delta R^2 = 0.255$. Accuracy is the strongest single summary (0.165, about 65% of the increment from verifier identity), and F_1 adds 0.119 (47%). Self-consistency across repeated gradings is the weakest summary tested (0.002). Restricting to the Qwen3 8B trainee (252 records, including the self-judge), verifier identity adds 0.434. The best single summaries there are the fraction of criteria graded as met, balanced accuracy, and false-positive rate (0.039–0.043), compared with 0.030 for F_1, 0.011 for false-negative rate, and 0.004 for calibration bias.

Agreement summaries therefore capture part, but not all, of the observed differences between training verifiers, and which summary ranks first depends on the records included. These are in-sample descriptive fits, not held-out predictions. The records share training runs and are not independent, so we do not attach significance tests to these comparisons or treat their record counts as numbers of independent experiments.