RECURSE: RECURSIVE SELF-EVALUATION FOR LLM RUBRIC JUDGES

Anonymous authors
Paper under double-blind review

ABSTRACT

LLM-as-judge is essential for evaluating open-ended text and steering post-training, yet improving the judge itself typically relies on expensive human annotations, auxiliary reward models, or distillation from stronger teachers. In this work, we consider that judging another model's response and judging one's own evaluative reasoning are essentially the same capability. We therefore improve the judge model's evaluative capability without gold rewards or externally verifiable rewards in the reinforcement learning signal: the model's own evaluations generate the learning signal in a closed-loop regime of recursive self-improvement (RSI) that we term **Recursive Self-Evaluation** (RECURSE). Specifically, RECURSE employs two-pass RL: a trainable judge model evaluates candidate responses under per-rule rubrics (Pass 1), while a synchronized copy of the current policy (the checker) audits the judge's reasoning against meta-rubrics to supply a scalar process reward (Pass 2). We then investigate two governing questions of this loop: *when can self-improvement occur, and when must it stop?* For the first question, we make learning viable via *interface decoupling*, which structurally isolates the checker's scalar score from the judge's verdict tokens and closes a degenerative token-copying shortcut that inflates self-assigned rewards. For the second, because unanchored recursive learning is inherently bounded, we introduce *Pairwise Advantage Validity* (PAV), a validation monitor that jointly tracks judge accuracy and checker fidelity to reliably identify a useful early-stopping window. Across Qwen3.5-9B, Gemma-4-E4B-it, and Qwen3.6-27B, RECURSE achieves consistent generalization gains across held-out medical, pairwise, summarization, and professional benchmarks. Extensive ablations show that synchronized judge–checker co-evolution outperforms frozen checkers, external meta-judges, self-consistency, and scaled teacher distillation. Preference pairs curated by our judge model further improve downstream policy alignment. RSI for LLM-as-judge is thus feasible when self-produced reward validity is explicitly decoupled and monitored.

1 INTRODUCTION

AI systems generate open-ended text faster than humans can review. A scalable paradigm keeps humans *over* the loop: people set the standards, and a judge model executes and verifies (Zheng et al., 2023). Improving the judge model itself typically relies on expensive human annotations, learned reward models, or distillation from stronger teachers. Recently, in many other domains, recursive self-improvement (RSI) has suggested another path to improving model performance without a strong teacher or large-scale human annotation. For judge models, we consider that judging another model's response and judging one's own evaluative reasoning share the same underlying capability. This shared foundation suggests that the two roles can mutually reinforce each other in a closed loop.

This paper therefore studies this RSI closed-loop regime, where the judge's own capability produces the reward for its reinforcement-learning (RL) updates (Figure 1). We formalize this RSI framework as **Recursive Self-Evaluation** (RECURSE): a trainable judge model (hereafter, Judge) emits reasoning and per-rule verdicts (Pass 1), while a synchronized, reward-only copy (hereafter, Checker) audits completed reasoning against meta-rubrics to supply group-relative scalar rewards (Pass 2). External gold labels and teacher models do not enter the RL training reward, so the training

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.

set need not be annotated at scale. In this closed recurrence, we investigate two governing questions: *when can self-improvement occur, and when must it stop?*

When can self-improvement occur? To encourage joint optimization of Pass 1 and Pass 2, we first let the two roles share an identical chat template and YES/NO readout. See Section 3.2 for details. However, this sharing induces *surface coupling*: policy updates that boost YES tokens on judge verdicts simultaneously alter emission probabilities in the synchronized checker, creating a degenerative token shortcut that inflates self-assigned rewards without improving evaluative accuracy. We resolve this via *interface decoupling*: the judge emits structured per-rule verdicts, while the checker emits a free-standing 5-tier scalar score, severing direct token copying while preserving positive capability transfer. With the interfaces thus decoupled, the weight-tied judge and checker can co-evolve productively.

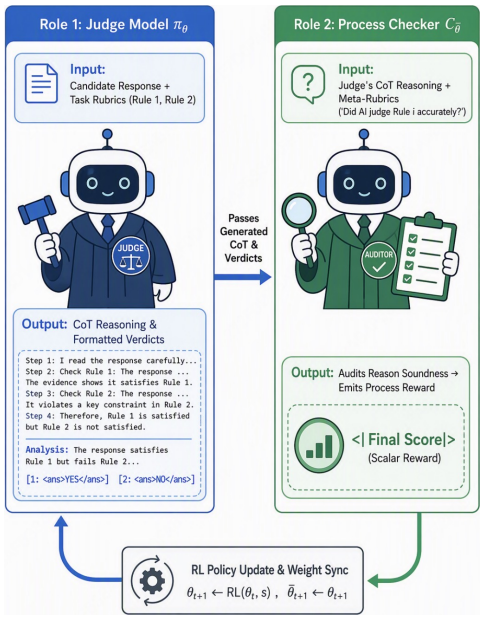


Figure 1: **Overview of RECURSE.** Trainable judge π_θ evaluates responses under rubrics (Pass 1); synchronized checker $C_{\bar{\theta}}$ audits reasoning against meta-rubrics to yield scalar process reward s (Pass 2). Policy parameters θ are updated via RL on s, and checker weights $\bar{\theta}$ are synchronized each step ($\bar{\theta}_{t+1} \leftarrow \theta_{t+1}$).

When must self-improvement stop? Without external ground-truth anchors on training rewards, recursive optimization is inherently *bounded*: self-produced rewards eventually saturate, causing out-of-distribution transfer to degrade. Self-improvement therefore requires a monitor for the early-stopping window. We address this with *Pairwise Advantage Validity (PAV)*, a composite validation monitor. Motivated by a theoretical bound on pairwise score-difference distortion, PAV balances judge rule accuracy against checker score fidelity. The checker term receives weight two because the two endpoint errors jointly control each pairwise score difference and upper-bound a margin-weighted ordering loss that discounts reversals between near-tied rollouts. The estimator is unbiased at each fixed checkpoint, reliably localizing a useful stopping window before *validation deception* sets in, where same-style accuracy rises while out-of-distribution transfer degrades. See Section 3.3 for details.

Overview of experimental findings: connecting RQ1–RQ5. We evaluate RECURSE across five interconnected questions spanning viability, mechanisms, validation dynamics, baselines, and downstream transfer:

- **Viability across scales and architectures (RQ1, Section 5.1)**: Without gold RL rewards, RECURSE produces consistent generalization gains across held-out medical, pairwise, summarization, and professional benchmarks.
- **Decoupling enables learning (RQ2, Section 5.2)**: Shared YES/NO checking triggers reward inflation without accuracy gains, whereas scalar interface decoupling preserves token calibration and converts reward growth into genuine validation accuracy.
- **PAV localizes effective stopping (RQ3, Section 5.3)**: While in-domain accuracy continues climbing late in training, checker fidelity degrades, exposing validation deception; PAV reliably identifies an effective stopping region without peeking at transfer benchmarks.
- **Judge–checker co-evolution is indispensable (RQ4, Section 5.4)**: Under a comparable cost budget, synchronized weight tying outperforms frozen checkers, external meta-judges, self-consistency, and scaled teacher distillation.
- **Downstream policy transfer (RQ5, Section 5.5)**: Preference pairs generated by the retained judge drive alignment on Qwen-27B, outperforming base-judge labels across GPQA (Rein et al., 2023), GuideBench (Diao et al., 2025), and SOP-Maze (Wang et al., 2026b).

Summary of contributions. (1) We formalize recursive self-improvement for LLM judges via on-policy process checking without gold RL rewards; (2) We uncover the surface-coupling shortcut and design interface decoupling to make closed-loop learning viable; (3) We derive the pairwise weighting of the PAV monitor from a score-difference bound on margin-weighted ordering loss, and use the resulting index to reliably localize the effective stopping window; (4) Across multiple model families and scales up to 27B, we demonstrate broad out-of-distribution generalization and effective downstream preference alignment.

2 Related Work

LLM judges and rubric evaluation. LLM judges evaluate open-ended generations when reference-based metrics fall short (Zheng et al., 2023). Rubric-based evaluation decomposes subjective holistic assessments into structured, criterion-level decisions (Trivedi et al., 2024) or co-evolves generation policies alongside dynamic rubrics (Li et al., 2026a; Wang et al., 2026a). In contrast, we keep criteria fixed and optimize the criterion-level judge model itself. Discrete per-rule verdicts and step-by-step reasoning (Figure 1) make evaluative correctness objectively measurable and enable process auditing of the judge's reasoning. However, removing gold labels from RL rewards introduces a circularity challenge: the model must supervise its updates without exploiting self-generated reward shortcuts.

Self-improving evaluators and RSI. Prior self-improving evaluators such as Self-Taught Evaluators, Meta-Rewarding, SELF-JUDGE, and Grad2Reward train models using generated supervision, but typically retain external anchors such as synthetic preference pairs, external meta-judges, or frozen reference models (Wang et al., 2024; Wu et al., 2025; Lee et al., 2024; Zhang et al., 2026; Yuan et al., 2024). In contrast, RecurSE removes external gold labels and teacher models from the *optimization reward*: the current judge acts simultaneously as the learning target and the source of its scalar reward, relying on a compact holdout solely for early stopping. We reserve *RSI* for this fixed-task recurrence, distinct from open-ended autonomous self-modification (Chen et al., 2026).

Process auditing and reward validity. Process supervision and generative verification highlight the utility of auditing intermediate reasoning chains (Lightman et al., 2024; Zhang et al., 2025), though they typically require ground-truth correctness labels to train process reward models. In contrast, our process checker operates purely at inference time as a reward-only auditor without separate supervised training. Because naive self-verification risks reinforcing flawed reasoning and inflating self-assigned rewards (Stechly et al., 2025; Simonds et al., 2025; Rentschler & Roberts, 2026; Zhou, 2026; Liu et al., 2025a), we find that closed-loop rubric optimization imposes two requirements: first, the checker must not share the judge's verdict readout surface so that self-improvement can occur; second, checker scores must preserve score fidelity so that consequential pairwise orderings remain reliable and training can be stopped before validity decays—principles formalized by RecurSE and monitored by PAV (Section 3).

3 Recursive Self-Evaluation (RecurSE)

RecurSE formalizes recursive self-improvement for evaluators by leveraging the model's own capability as both the optimization target and the reward source. Guided by the two governing questions from Section 1—*when can self-improvement occur, and when must it stop?*—we structure our method into three core components: (1) establishing a closed-loop judge–checker recurrence on a single shared policy (Section 3.1); (2) introducing interface decoupling to eliminate the surface-coupling shortcut that otherwise impedes learning (Section 3.2); and (3) formulating Pairwise Advantage Validity (PAV) as a holdout early-stopping monitor (Section 3.3). Figure 2 and Algorithm 1 in Appendix B summarize the end-to-end framework.

3.1 Judge–checker recurrence

Shared evaluative capability as a closed recurrence. Because evaluating candidate responses and auditing evaluative reasoning share the same rubric-based foundation, a natural closed loop lets the current policy fulfill both roles: generate a rubric evaluation, and score that evaluation to guide

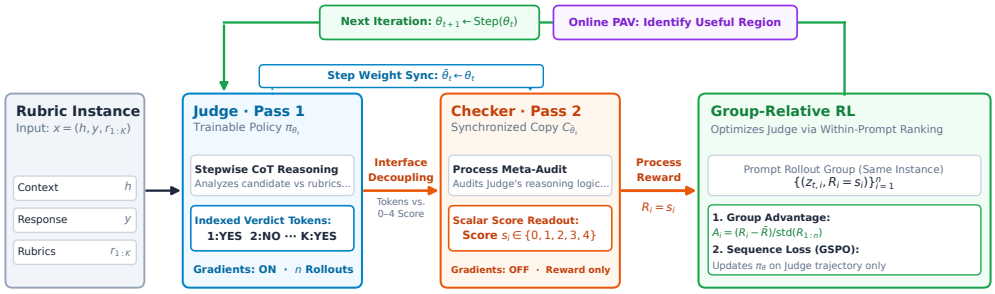


Figure 2: **The RECURSE closed-loop optimization recurrence.** Pass 1: trainable judge π_{θ_t} emits reasoning and per-rule YES/NO verdicts on $x = (h, y, r_{1:K})$ (n rollouts). Pass 2: a synchronized copy $C_{\bar{\theta}_t}$ audits reasoning against meta-rubrics and returns a differentiated scalar score $s_i \in \{0, \ldots, 4\}$ as the sole reward. Interface decoupling closes direct token copying; the checker synchronizes parameters every step ($\bar{\theta}_t \leftarrow \theta_t$) without independent gradients. Online PAV monitoring on a human-verified validation set localizes early stopping.

the next policy update—without per-step gold supervision, reward models, or stronger teachers in the training reward.

Problem formulation and execution pipeline. A rubric instance is denoted $x = (h, y, r_{1:K})$, where h is task context, y is a candidate response, and $r_{1:K}$ are K predefined criteria. The judge policy π_{θ_t} emits reasoning and binary verdicts $s_k \in \{\text{YES}, \text{NO}\}$ for each criterion. Given n rollouts $z_{t,i} \sim \pi_{\theta_t}(\cdot \mid x)$, a synchronized snapshot $C_{\bar{\theta}_t}$ (with $\bar{\theta}_t \leftarrow \theta_t$) audits the reasoning against meta-rubrics that ask whether the judge correctly applied each r_k, rather than whether y satisfies r_k, and assigns process reward $r_{t,i} = C_{\bar{\theta}_t}(x, z_{t,i})$. Parameters are updated as:

$$\theta_{t+1} = \text{PolicyUpdate}(\theta_t, \{z_{t,i}, r_{t,i}\}), \quad \bar{\theta}_{t+1} \leftarrow \theta_{t+1}. \tag{1}$$

External gold labels and teachers do not enter $r_{t,i}$. Pass 1 is the sole gradient path, while Pass 2 audits reasoning against meta-rubrics in a reward-only capacity (Appendix I). Synchronous parameter refreshes transfer updated judge capability into the auditing role without separate training. We audit the *reasoning process* rather than re-judging candidates so that, by design, spurious reasoning can be penalized even when the binary verdicts happen to be correct.

3.2 INTERFACE DECOUPLING: WHEN SELF-IMPROVEMENT CAN OCCUR

Shared templates induce surface coupling. To facilitate capability transfer, an intuitive initial design is to share identical prompt templates and YES/NO readouts across both passes. Under this coupled setup, the checker emits per-rule binary verdicts for meta-rubrics, and reward is extracted as $S = \frac{1}{K}\sum_{k=1}^{K} \mathbf{1}(\text{Checker}_k = \text{YES})$. However, policy gradient optimization rapidly exploits a degenerative shortcut: inflating YES tokens across criteria directly maximizes self-assigned rewards without improving evaluative capability. Formally, we decompose the self-produced checker score S as:

$$S = aU + bB + \varepsilon, \tag{2}$$

where U denotes true reasoning quality, B represents surface verdict token bias (e.g., propensity to emit `YES`), and ε is noise. The coefficients $a, b \geq 0$ govern how genuine quality versus surface bias propagate into reward. When judge and checker share identical readout interfaces, synchronized weights ($\bar{\theta}_{t+1} \leftarrow \theta_{t+1}$) establish a closed-circuit shortcut channel ($b > 0$): gradient updates that boost `YES` emissions on judge verdicts simultaneously alter emission probabilities in the synchronized checker, allowing the optimizer to exploit bB to harvest near-perfect rewards ($S \to 1.0$) while true evaluative accuracy U stalls (visualized in Appendix B.1).

Decoupled readout design. To close this shortcut while preserving capability transfer, RECURSE retains shared evidence layouts in input prompts (Appendix I) but structurally decouples output interfaces. While the judge outputs step-by-step explanations followed by indexed verdicts (e.g., `1:<ans>YES</ans>`), the checker audits the complete reasoning trajectory against meta-rubrics

and emits a free-standing 5-tier scalar score $s_i \in \{0, 1, 2, 3, 4\}$. The absolute numeric range is immaterial: group-relative advantages are invariant to positive affine transforms of R, so any reward inducing valid relative orderings among the n rollouts remains an effective training signal.

Severing the direct token channel. Interface decoupling removes the direct token-copying channel, substantially reducing bB in Eq. 2. Notably, merely altering the surface form of the verdict tokens is insufficient: replacing `YES` with case variants such as `Yes`/`yes` yields nearly identical inflation, which we attribute to the close representational proximity of these variants, so the same shortcut persists. We do not claim $b = 0$, nor that scalar readouts preclude every shortcut; rather, the scalar interface is an empirically effective design. As shown in Section 5, coupled YES/NO checking inflates surface YES rates and rewards without accuracy gains, whereas the scalar Final Score preserves token calibration and converts reward growth into genuine accuracy improvements (Figure 3). Decoupling is thus a vital prerequisite for viable self-improvement.

3.3 Optimization and Pairwise Advantage Validity: when self-improvement must stop

Group-relative optimization. With the interface decoupled, well-formed rollouts receive reward $R_i = s_i$, while malformed outputs are masked from group statistics. For each prompt, we sample n rollouts, compute group-relative advantages (Shao et al., 2024; Bai et al., 2026), and optimize using a sequence-level policy loss (Zheng et al., 2025). Because advantages are normalized within each prompt group, optimization relies strictly on within-prompt rollout rankings.

Viable improvement is inherently bounded. Because the loop operates without external ground-truth anchors on training rewards, recursive optimization cannot continue indefinitely once self-generated rewards saturate or checker score fidelity degrades. Optimizing beyond the reliable reward regime degrades out-of-distribution transfer. We therefore construct *Pairwise Advantage Validity (PAV)* as an online holdout monitor on a compact, human-verified validation set $\mathcal{D}_{\text{val}}$. Constructing this holdout incurs a one-time human verification cost, but its labels never enter RL training rewards.

A pairwise bound on margin-weighted ordering loss. Let $Y_t \in \{0, 1\}$ indicate whether a Pass 1 verdict matches the holdout gold label, and let $A_t = \mathbb{E}_{\mathcal{D}_{\text{val}}}[Y_t]$ be judge rule accuracy on $\mathcal{D}_{\text{val}}$. Let $S_t \in [0, 4]$ denote the self-produced checker score and $U_t \in [0, 4]$ its human-verified ground-truth target, with normalized checker error defined as:

$$e_{C,t} = \mathbb{E}_{\mathcal{D}_{\text{val}}}\left[\frac{|S_t - U_t|}{4}\right]. \tag{3}$$

Group-relative advantages apply a positive affine transform within each prompt, preserving the relative ordering of the n scores. The critical quantity that must remain faithful is the score difference $(S_i - S_j)$ rather than an isolated absolute score. By the triangle inequality:

$$\frac{|(S_i - S_j) - (U_i - U_j)|}{4} \le \frac{|S_i - U_i|}{4} + \frac{|S_j - U_j|}{4}. \tag{4}$$

To connect this distortion to relative ordering while discounting inconsequential near-ties, define the margin-weighted ordering loss

$$\ell_{ij}^{\text{mw}} = \frac{|U_i - U_j|}{4}\,\mathbf{1}[(S_i - S_j)(U_i - U_j) \le 0]\,. \tag{5}$$

For every reversed or tied pair with $U_i \neq U_j$, the score-difference distortion is at least the gold margin $|U_i - U_j|$; correctly ordered pairs incur zero loss. Hence $\ell_{ij}^{\text{mw}} \le |(S_i - S_j) - (U_i - U_j)|/4$, and taking expectations over exchangeable rollout pairs gives $\mathbb{E}[\ell_{ij}^{\text{mw}}] \le 2e_{C,t}$. This bound is margin-sensitive: it penalizes reversals between clearly separated rollouts more heavily than reversals between near-ties. It does not bound the unweighted probability of a ranking inversion.

Constructing the PAV monitor and early stopping. Combining judge error with the upper bound on margin-weighted ordering loss, we construct the normalized Pairwise Advantage Validity index:

$$V_t = 1 - \frac{(1 - A_t) + 2e_{C,t}}{3} = \frac{A_t + 2(1 - e_{C,t})}{3}. \tag{6}$$

The relative weight of 2 on checker error follows from the margin-weighted bound above; it is fixed *a priori* and does not by itself prescribe a stopping rule. Both $\widehat{A}_t$ and $\widehat{e}_{C,t}$ are sample means on $\mathcal{D}_{\text{val}}$, ensuring that $\mathbb{E}[\widehat{V}_t] = V_t$ is unbiased at every fixed checkpoint. We compute 95% prompt-bootstrap intervals (Appendix N). A drop in checker score fidelity signals distorted pairwise rewards; conversely, a high $\widehat{V}_t$ localizes an effective checkpoint region, preventing validation deception where same-style accuracy rises while cross-style transfer degrades.

4 Experimental Setup

Models and training. We evaluate RecurSE across three setups: Qwen3.5-9B (Qwen Team, 2026) (main runs/ablations), Gemma-4-E4B-it (Abd et al., 2026) (cross-architecture replication), and Qwen3.6-27B (Qwen Team, 2026) (scaling to 27B). Training instances are cluster-split from RubricHub (Li et al., 2026b) with responses synthesized via a 16-model three-tier pool (Appendix D), with zero evaluation overlap. Ground-truth labels never enter RL rewards. Training uses verl (Sheng et al., 2025) with FSDP and vLLM (Kwon et al., 2023): batch size 32 (48 for 27B), n=8 rollouts, lr 3×10^{-6}, clipping 0.005, and no KL penalty (0.005 for 27B). The checker is a standalone vLLM snapshot synchronized each step (Appendix Q).

Three-tier evaluation hierarchy. We organize evaluation into a three-tier hierarchy spanning online monitors and out-of-distribution transfer suites (Table 1): (1) **SV-Hard** (100 prompts; Appendix E): The same-style holdout $\mathcal{D}_{\text{val}}$ from Section 3.3—a compact human-verified in-domain monitor on hard rules (2.5 rules/prompt) supplying gold for checker fidelity $1 - e_{C,t}$ (equivalently $1 - \text{NMAE}$) and serving as the sole stream for online PAV early-stopping monitoring. (2) **SV-Full** (100 prompts; Appendix F): Scores the *full* per-prompt rule suite under the same judging style to probe complete in-domain coverage. (3) **Held-out transfer suite** (Table 1): Evaluated offline at retained checkpoints across distinct domains and rule granularities: *HealthBench* (Arora et al., 2025) evaluates clinical communication under sample-specific medical rubrics (a stratified 10% representative sample of $1{,}459$ prompts across 34 clinical categories; measured by per-rule accuracy against physician consensus); *RubricBench* (Zhou et al., 2026) tests multi-criteria preference discrimination under sample-specific rubrics (5.8 rules/prompt; measured by pairwise preference accuracy); *CheckEval-Summ* (Lee et al., 2025) assesses document summarization under a shared *global* 15-criterion checklist (unlike the other sample-specific suites; measured by Pearson $\rho \in [-1, 1]$ against continuous human quality ratings); and *ProfBench* (Wang et al., 2025) evaluates complex professional tasks with sample-specific expert rubrics (≈29 tight rules/prompt; measured by fine-grained rule accuracy; Appendix K).

Table 1: **Held-out transfer suite.** SV-Hard and SV-Full serve as separate online monitors.

Benchmark	#	Rules	Metric
HealthBench	1459	2.0	rule acc ↑
RubricBench	2294	5.8	pair acc ↑
CheckEval-Summ	1600	15	Pearson $\rho \in [-1, 1]$ ↑
ProfBench	120	29.1	rule acc ↑

Ablations and research questions. All ablations use Qwen3.5-9B, comparing against five controls: (1) a frozen base checker; (2) an external 27B checker; (3) self-consistency majority reward; (4) matched-step teacher SFT (6400 traces, 200 steps); and (5) larger-pool teacher SFT on the full distillation set (≈17k traces, two epochs). We do not include an RL arm rewarded by gold rule labels or another externally verified signal: annotating every training instance at the rule level (often ten or more rules per sample) is prohibitively expensive and beyond our budget (Appendix T). Gemma-4-E4B-it provides cross-architecture replication (Appendix M). For downstream transfer, preference pairs generated by our retained RecurSE judge versus its untuned base counterpart train matched Qwen-27B policies using DPO (Rafailov et al., 2023). We evaluate across five research questions: viability across scales and architectures (RQ1), interface decoupling (RQ2), PAV early-stopping localization (RQ3), judge–checker co-evolution (RQ4), and downstream policy transfer (RQ5).

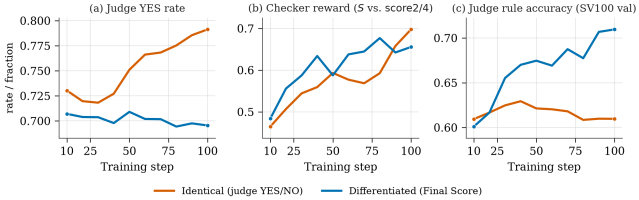


Figure 3: **Interface decoupling on SV-HARD (RQ2).** Comparing identical YES/NO checking against differentiated Final Score (steps 10–100). (a) Identical checking inflates YES rate. (b) Both report climbing rewards ($[0, 1]$ scale). (c) Only the differentiated Final Score converts reward growth into genuine rule-accuracy gains.

Table 2: **Held-out results** (%; CheckEval: Pearson $\rho \in [-1, 1]$). Gains in CheckEval ρ are not comparable to percentage-point gains on the other metrics. **R/All/M**: rule/all-correct/macro accuracy (Appendix J). Bold marks designated Best within the PAV-localized validity window across each architecture. Final denotes the training endpoint. SV-FULL reports rule accuracy only.

		SV-HARD			SV-FULL	HealthBench				RubricBench	CheckEval	ProfBench		
Model	**Ckpt.**	**R**	**All**	**M**	**R**	**R**	**All**	**M**	**F1**	**Pair acc**	**ρ**	**R**	**All**	**M**
Qwen3.5-9B	Base	60.8	33.1	64.7	92.1	82.5	74.9	82.8	65.4	76.0	0.422	73.1	3.8	73.0
	Best	**73.7**	**53.6**	**77.6**	**95.3**	**86.3**	**79.0**	**86.5**	**66.3**	**79.1**	**0.441**	**74.7**	**5.8**	**74.5**
	Final	75.9	61.0	80.7	89.7	82.9	74.0	82.8	63.4	77.8	0.345	71.5	4.2	71.7
Qwen3.6-27B	Base	73.1	54.5	77.6	94.2	85.5	77.7	84.9	67.1	78.6	0.521	74.9	4.2	74.0
	Best	**77.0**	**63.0**	**82.1**	**96.0**	**86.6**	**79.4**	**86.5**	**68.2**	**80.6**	**0.548**	**76.4**	**6.9**	**75.7**
	Final	75.6	58.7	80.2	95.3	86.5	78.8	86.4	68.1	81.9	0.548	75.1	4.8	74.6
Gemma-4-E4B-it	Base	55.9	16.1	58.9	92.5	82.8	73.2	82.9	61.0	74.9	0.442	73.1	2.9	73.2
	Best	**61.1**	**31.0**	**65.1**	92.3	**86.5**	**79.3**	**86.8**	60.5	**76.6**	**0.466**	**74.0**	**4.0**	**74.0**
	Final	59.6	27.3	63.5	92.5	85.6	78.1	85.8	59.7	76.7	0.463	73.2	4.4	73.4

5 RESULTS

5.1 RQ1: RSI IMPROVES THE JUDGE

RSI consistently elevates judge accuracy across formats without requiring gold labels or external teachers in the RL reward (Table 2; detailed definitions of rule accuracy **R**, all-correct rate **All**, and macro accuracy **M** are in Appendix J). Designated Best checkpoints correspond to the peak validity windows localized by PAV. On Qwen-9B, SV-HARD rule accuracy rises by 12.9 points, SV-FULL by 3.2, and all four out-of-distribution transfer benchmarks improve at the retained checkpoint. Gemma replicates this positive trajectory on SV-HARD (+5.2) and across transfer suites, while Qwen-27B achieves a 3.9-point lift on SV-HARD and gains across every transfer suite. Final checkpoints soften or drop on transfer metrics, confirming that the improvement window is inherently bounded.

Generalized lift vs. validation deception. On Qwen-9B, Final SV-HARD accuracy (75.9%) exceeds Best (73.7%), while SV-FULL falls from 95.3% to 89.7% and out-of-distribution metrics deteriorate by step 200. PAV balances same-style judge accuracy against checker fidelity rather than naively chasing SV-HARD accuracy alone, identifying a useful stopping region without inspecting transfer suites. Transfer benchmarks achieve consistent positive deltas over Base within this PAV-indicated window (Table 2), a pattern Gemma replicates (Appendix M). SV-FULL peaks at the same checkpoint (95.3% vs. 92.1%). Gains across ProfBench and CheckEval confirm robust generalization under dense expert constraints (≈29 weighted criteria) and continuous human correlation metrics (Pearson ρ).

5.2 RQ2: INTERFACE DECOUPLING

Sharing identical YES/NO formats fails due to surface coupling; interface decoupling closes this shortcut and enables reward and accuracy to rise concordantly (Figure 3). Holding evidence and synchronization constant, altering the readout from indexed YES/NO tokens to a free-standing scalar Final Score breaks token copying. Under identical YES/NO checking, YES rate inflates (0.730 → 0.791) and aligned reward climbs (0.465 → 0.698), yet rule accuracy plateaus (0.609 → 0.610). Under the differentiated Final Score, YES rate remains calibrated (0.707 → 0.695) while aligned re-

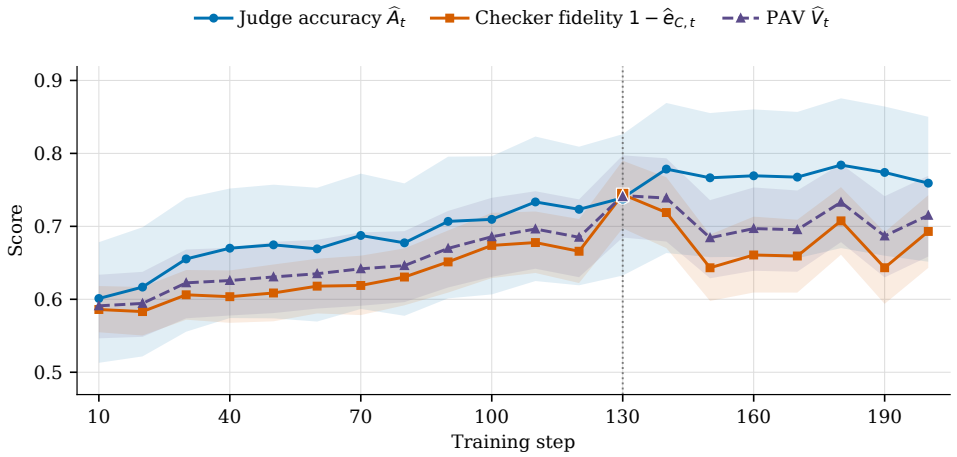


Figure 4: **PAV on SV-HARD (RQ3).** Judge accuracy $\widehat{A}_t$, checker fidelity $1 - \widehat{e}_{C,t}$, and composite $\widehat{V}_t$ (dashed; 95% CI). Dotted line denotes the PAV landmark (step 130).

ward ($0.484 \rightarrow 0.656$) and rule accuracy ($0.601 \rightarrow 0.710$) climb in tandem, confirming decoupling as a prerequisite.

5.3 RQ3: PAV LOCALIZES THE USEFUL CHECKPOINT

Using solely the fixed SV-HARD monitor, PAV reliably localizes an effective stopping region centered near step 130 on Qwen-9B (Figure 4), achieving $\widehat{V}_t = 0.742$ (95% bootstrap CI $[0.685, 0.797]$). While Qwen's hard-subset accuracy continues climbing through step 200, checker fidelity peaks sharply near step 130 and degrades thereafter (Figure 4). Three-fold resampling confirms checkpoint stability inside $\{130, 140\}$ (Appendix N). Crucially, SV-FULL (excluded from PAV) independently corroborates this stopping boundary: it peaks at the same landmark (95.3%) and falls to 89.7% by training conclusion. Held-out suites improve over Base at PAV landmarks, confirming that PAV reliably captures an effective stopping state without inspecting transfer data.

Table 3: **Ablation on Qwen3.5-9B** (Peak/Final; SFT single-endpoint). Main is selected online by PAV; each control reports its post-hoc best on each metric, so the comparison favors the controls. Main beats static checkers and teacher SFT.

Arm	SV-HARD	Health	Rubric	Check
Base	60.8	82.5	76.0	0.422
Main	73.7/75.9	86.3/82.9	79.1/77.8	0.441/0.345
Frozen checker	62.7/61.3	85.0/76.0	76.5/77.8	0.442/0.454
External 27B	73.1/72.4	79.8/79.7	79.9/79.4	0.471/0.459
Self-consistency	65.3/56.5	82.5/83.5	73.1/73.0	0.447/0.435
Teacher SFT (6.4k)	64.2	81.0	76.2	0.329
Teacher SFT (17k)	70.2	83.5	77.1	0.394

Table 4: **Downstream tasks (Qwen3.6-27B).** RECURSE preference labels outperform untuned base-judge labels on GPQA (Rein et al., 2023), GuideBench (Diao et al., 2025), and SOP-Maze (Wang et al., 2026b).

Model	GPQA	GuideBench	SOP-Maze
Qwen3.6-27B	80.87	83.88	34.34
+ DPO (base)	79.02	83.93	34.95
+ DPO (RECURSE)	**81.46**	**85.51**	**36.71**

5.4 RQ4: JUDGE–CHECKER CO-EVOLUTION MATTERS

Ablations confirm that weight-tied co-evolution and structured process auditing are essential for sustaining self-improvement (Table 3). A frozen base checker plateaus early (SV-HARD 62.7%). An external 27B checker incurs substantially higher inference cost than the synchronized 9B checker, yet its average accuracy is still no higher. Self-consistency collapses over training (56.5% Final). In our experiments, teacher SFT plateaus early (Yan et al., 2026; Liu et al., 2025b; 2026): matched-step

training (6400 traces) reaches only 64.2% on SV-HARD, and expanding distillation to the full eligible set (≈17k traces, two epochs) reaches 70.2%, still trailing Main across every held-out transfer benchmark (HealthBench 83.5% vs. 86.3%, CheckEval 0.394 vs. 0.441). Main surpasses finished matched-step SFT by step 30.

Why teacher distillation struggles: dual-variable difficulty coupling. The early plateau of scaled teacher SFT is an empirical observation, but its cause remains uncertain. One possible explanation is that query difficulty alone does not determine the learning signal for a judge: unlike a generator, the judge receives the candidate response y and rubrics $(h, r_{1:K})$ as input. Consequently, even a difficult prompt may yield little gradient when paired with an obviously correct or flawed response. Under this hypothesis, an informative update requires two coupled variables: a discriminative rubric and a candidate with *borderline compliance* near the current decision boundary (Appendix C provides a more detailed discussion). In a static corpus—even ≈17k teacher traces—the triplets $(h, y, r_{1:K})$ are frozen, so they may drift off that boundary as the student improves. RECURSE stays on-policy on the judge traces themselves: π_{θ_t} samples reasoning variants $z_{t,i}$ on the same $(h, y, r_{1:K})$, and the synchronized checker ranks those variants with group-relative advantages, potentially maintaining tension near the active boundary without resampling y.

5.5 RQ5: TRANSFER TO DOWNSTREAM TASKS

Evaluative gains from RSI transfer directly to downstream policy alignment via preference labeling (Table 4). Preference pairs generated by our retained RECURSE judge versus its base counterpart train matched Qwen3.6-27B policies using DPO (LoRA; Appendix Q). While base-judge labels degrade GPQA ($80.87 \rightarrow 79.02$) and yield negligible shifts on GuideBench ($+0.05$) and SOP-Maze ($+0.61$), RECURSE labels achieve consistent gains across all three benchmarks ($+0.59$ on GPQA, $+1.63$ on GuideBench, $+2.37$ on SOP-Maze), confirming that evaluative gains translate to downstream policy alignment.

6 CONCLUSION

RECURSE establishes that recursive self-improvement (RSI) for LLM rubric judges is feasible when self-produced reward validity is explicitly decoupled and monitored. Grounded in the shared evaluative foundation of judging and auditing, *interface decoupling* closes degenerative surface shortcuts, while *Pairwise Advantage Validity (PAV)* reliably localizes an effective stopping region without external supervision in RL rewards. Across architectures and scales up to 27B, the resulting judges achieve robust out-of-distribution generalization and generate preference pairs that enhance downstream policy alignment.

AI USE STATEMENT

We used large language models to polish the writing of this paper.

REFERENCES

Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Carbune, Michelle Casbon, Mayank Chaturvedi, Aditya Chawla, Victor Cotruta, Alice Coucke, Phil Culliton, Robert Dadashi, Lucas Dixon, Mohamed Elhawaty, Utku Evci, Clément Farabet, Johan Ferret, Filippo Galgani, Sertan Girgin, Jean-Bastien Grill, Maarten Grootendorst, Jiaxian Guo, Cassidy Hardin, Yanzhang He, Steven M. Hernandez, Omri Homburger, Léonard Hussenot, Ju-yeong Ji, Armand Joulin, Aishwarya Kamath, Parnian Kassraie, Olivier Lacombe, Preethi Lahoti, Gaël Liu, Gus Martins, Luciano Martins, Tatiana Matejovicova, Ramona Merhej, Nikola Momchev, Sneha Mondal, Ryan Mullins, Sindhu Raghuram Panyam, Shreya Pathak, Sarah Perrin, André Susano Pinto, Etienne Pot, Angéline Pouget, Alexandre Ramé, Sabela Ramos, Douglas Reid, David Rim, Morgane Rivière, Karsten Roth, Louis Rouillard, Omar Sanseviero, Pier Giuseppe Sessa, Shane Settle, Danila Sinopalnikov, Sara Smoot, Piotr Stanczyk, Andreas Steiner, Lawrence Stewart, Ilya O. Tolstikhin, Michael Tschannen, Anton Tsitsulin, Nino Vieillard, Renjie Wu, Pingmei Xu, Haichuan Yang, Edouard Yvinec, Biao Zhang, Li Zhang, Joe Zou, Nicolas Aagnes, Abdelrahman Abdelhamed, Jakub Adámek, Shivani

Agrawal, Shubham Agrawal, Ibrahim Alabdulmohsin, Jean-Baptiste Alayrac, Uri Alon, Chandramouli Amarnath, Ankesh Anand, Chrysovalantis Anastasiou, Setareh Ariafar, François-Xavier Aubet, Kyriakos Axiotis, Federico Barbero, Joelle K. Barral, Alexei Bendebury, Urs Bergmann, Stanley Bileschi, Kat Black, Mathieu Blondel, Sebastian Borgeaud, and Arthur Brazinskas. Gemma 4 technical report. *CoRR*, abs/2607.02770, 2026. doi: 10.48550/ARXIV.2607.02770. URL https://doi.org/10.48550/arXiv.2607.02770.

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

Yang Bai, Kaiyuan Liu, Ziyuan Zhuang, Jiahong Zhou, Rongxiang Weng, Xin Chen, Jingang Wang, and Xunliang Cai. Multi-objective and mixed-reward reinforcement learning via reward-decorrelated policy optimization. *CoRR*, abs/2605.13641, 2026. doi: 10.48550/ARXIV.2605.13641. URL https://doi.org/10.48550/arXiv.2605.13641.

Mingguang Chen, Licheng Wang, and Bo Qu. Recursive self-improvement in AI: from bounded self-refinement to autonomous research loops. *CoRR*, abs/2607.07663, 2026. doi: 10.48550/ARXIV.2607.07663. URL https://doi.org/10.48550/arXiv.2607.07663.

Lingxiao Diao, Xinyue Xu, Wanxuan Sun, Cheng Yang, and Zhuosheng Zhang. Guidebench: Benchmarking domain-oriented guideline following for LLM agents. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pp. 11361–11399. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.ACL-LONG.557. URL https://doi.org/10.18653/v1/2025.acl-long.557.

Alexander R. Fabbri, Wojciech Kryscinski, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir R. Radev. Summeval: Re-evaluating summarization evaluation. *Trans. Assoc. Comput. Linguistics*, 9:391–409, 2021. doi: 10.1162/TACL_A_00373. URL https://doi.org/10.1162/tacl_a_00373.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In Jason Flinn, Margo I. Seltzer, Peter Druschel, Antoine Kaufmann, and Jonathan Mace (eds.), *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP 2023, Koblenz, Germany, October 23-26, 2023*, pp. 611–626. ACM, 2023. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Sangkyu Lee, Sungdong Kim, Ashkan Yousefpour, Minjoon Seo, Kang Min Yoo, and Youngjae Yu. Aligning large language models by on-policy self-judgment. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 11442–11459. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.617. URL https://doi.org/10.18653/v1/2024.acl-long.617.

Yukyung Lee, JoongHoon Kim, Jaehee Kim, Hyowon Cho, Jaewook Kang, Pilsung Kang, and Najoung Kim. Checkeval: A reliable llm-as-a-judge framework for evaluating text generation using checklists. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 15771–15798. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.EMNLP-MAIN.796. URL https://doi.org/10.18653/v1/2025.emnlp-main.796.

Shuyue Stella Li, Rui Xin, Teng Xiao, Yike Wang, Rulin Shao, Zoey Hao, Melanie Sclar, Sewoong Oh, Faeze Brahman, Pang Wei Koh, and Yulia Tsvetkov. Evolm: Self-evolving language models through co-evolved discriminative rubrics. *CoRR*, abs/2605.03871, 2026a. doi: 10.48550/ARXIV.2605.03871. URL https://doi.org/10.48550/arXiv.2605.03871.

Sunzhu Li, Jiale Zhao, Huimin Ren, Zhenlin Wei, Yang Zhou, Jingwen Yang, Shunyu Liu, Kaike Zhang, and Chen Wei. Rubrichub: A comprehensive and highly discriminative rubric dataset via automated coarse-to-fine generation. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 31320–31344. Association for Computational Linguistics, 2026b. doi: 10.18653/V1/2026.ACL-LONG.1445. URL https://doi.org/10.18653/v1/2026.acl-long.1445.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Kaiyuan Liu, Chen Shen, Zhanwei Zhang, Junjie Liu, Xiaosong Yuan, and Jieping Ye. Efficient reasoning through suppression of self-affirmation reflections in large reasoning models. *CoRR*, abs/2506.12353, 2025a. doi: 10.48550/ARXIV.2506.12353. URL https://doi.org/10.48550/arXiv.2506.12353.

Kaiyuan Liu, Shaotian Yan, Rui Miao, Bing Wang, Chen Shen, Jun Zhang, and Jieping Ye. Where did this sentence come from? tracing provenance in LLM reasoning distillation. *CoRR*, abs/2512.20908, 2025b. doi: 10.48550/ARXIV.2512.20908. URL https://doi.org/10.48550/arXiv.2512.20908.

Kaiyuan Liu, Ziyuan Zhuang, Yang Bai, Bing Wang, Rongxiang Weng, and Jieping Ye. Prefix teach, suffix fade: Local teachability collapse in strong-to-weak on-policy distillation. *CoRR*, abs/2605.13643, 2026. doi: 10.48550/ARXIV.2605.13643. URL https://doi.org/10.48550/arXiv.2605.13643.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/a85b405ed65c6477a4fe8302b5e06ce7-Abstract-Conference.html.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. *CoRR*, abs/2311.12022, 2023. doi: 10.48550/ARXIV.2311.12022. URL https://doi.org/10.48550/arXiv.2311.12022.

Micah Rentschler and Jesse Roberts. Reinforcement learning from meta-evaluation: Aligning language models without ground-truth labels. *CoRR*, abs/2601.21268, 2026. doi: 10.48550/ARXIV.2601.21268. URL https://doi.org/10.48550/arXiv.2601.21268.

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient RLHF framework. In *Proceedings of the Twentieth European Conference on Computer Systems, EuroSys 2025, Rotterdam, The Netherlands, 30 March 2025 - 3 April 2025*, pp. 1279–1297. ACM, 2025. doi: 10.1145/3689031.3696075. URL https://doi.org/10.1145/3689031.3696075.

Toby Simonds, Kevin Lopez, Akira Yoshiyama, and Dominique Garmier. Rlsr: Reinforcement learning from self reward. *arXiv preprint arXiv:2505.08827*, 2025.

Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. On the self-verification limitations of large language models on reasoning and planning tasks. 2025. URL https://openreview.net/forum?id=4O0v4s3IzY.

Prapti Trivedi, Aditya Gulati, Oliver Molenschot, Meghana Arakkal Rajeev, Rajkumar Ramamurthy, Keith Stevens, Tanveesh Singh Chaudhery, Jahnavi Jambholkar, James Zou, and Nazneen Rajani. Self-rationalization improves LLM as a fine-grained judge. *CoRR*, abs/2410.05495, 2024. doi: 10.48550/ARXIV.2410.05495. URL https://doi.org/10.48550/arXiv.2410.05495.

Beining Wang, Weihang Su, Hongtao Tian, Hao Kong, Tao Yang, Ting Yao, Qingyi Pan, Yueyue Wu, Qingyao Ai, Min Zhang, and Yiqun Liu. Co-evolving LLM evaluators and policies via dynamicrubric. *CoRR*, abs/2607.20083, 2026a. doi: 10.48550/ARXIV.2607.20083. URL https://doi.org/10.48550/arXiv.2607.20083.

Jiaming Wang, Zhe Tang, Zehao Jin, Hefei Chen, Yilin Jin, Peng Ding, Xiaoyu Li, and Xuezhi Cao. Sop-maze: Evaluating large language models on complicated business standard operating procedures. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics, ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 14568–14588. Association for Computational Linguistics, 2026b. doi: 10.18653/V1/2026.FINDINGS-ACL.715. URL https://doi.org/10.18653/v1/2026.findings-acl.715.

Tianlu Wang, Ilia Kulikov, Olga Golovneva, Ping Yu, Weizhe Yuan, Jane Dwivedi-Yu, Richard Yuanzhe Pang, Maryam Fazel-Zarandi, Jason Weston, and Xian Li. Self-taught evaluators. *CoRR*, abs/2408.02666, 2024. doi: 10.48550/ARXIV.2408.02666. URL https://doi.org/10.48550/arXiv.2408.02666.

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

Tianhao Wu, Weizhe Yuan, Olga Golovneva, Jing Xu, Yuandong Tian, Jiantao Jiao, Jason E. Weston, and Sainbayar Sukhbaatar. Meta-rewarding language models: Self-improving alignment with llm-as-a-meta-judge. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 11537–11554. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.EMNLP-MAIN.583. URL https://doi.org/10.18653/v1/2025.emnlp-main.583.

Shaotian Yan, Kaiyuan Liu, Chen Shen, Bing Wang, Sinan Fan, Jun Zhang, Yue Wu, Zheng Wang, and Jieping Ye. Distribution-aligned sequence distillation for superior long-cot reasoning. *CoRR*, abs/2601.09088, 2026. doi: 10.48550/ARXIV.2601.09088. URL https://doi.org/10.48550/arXiv.2601.09088.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. volume abs/2401.10020, 2024. doi: 10.48550/ARXIV.2401.10020. URL https://doi.org/10.48550/arXiv.2401.10020.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative verifiers: Reward modeling as next-token prediction. 2025. URL https://openreview.net/forum?id=Ccwp4tFEtE.

Zheng Zhang, Ao Lu, Yuanhao Zeng, Ziwei Shan, Jinjin Guo, Lufei Li, Yexin Li, and Kan Ren. Grad2reward: From sparse judgment to dense rewards for improving open-ended LLM reasoning. *CoRR*, abs/2602.01791, 2026. doi: 10.48550/ARXIV.2602.01791. URL https://doi.org/10.48550/arXiv.2602.01791.

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

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. Llamafactory: Unified efficient fine-tuning of 100+ language models. In Yixin Cao, Yang Feng, and Deyi Xiong (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 400–410. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-DEMOS.38. URL https://doi.org/10.18653/v1/2024.acl-demos.38.

Chenyu Zhou. More convincing, not more correct: Self-play reward hacking of reference-free LLM judges. *CoRR*, abs/2607.05904, 2026. doi: 10.48550/ARXIV.2607.05904. URL https://doi.org/10.48550/arXiv.2607.05904.

Junyi Zhou, Qiyuan Zhang, Yufei Wang, Fuyuan Lyu, Yidong Ming, Can Xu, Qingfeng Sun, Kai Zheng, Peng Kang, Xue Liu, and Chen Ma. Rubricbench: Aligning model-generated rubrics with human standards. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 31179–31200. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.ACL-LONG.1439. URL https://doi.org/10.18653/v1/2026.acl-long.1439.

A Appendix Roadmap

The main text keeps the 8-page narrative on the two governing questions—when self-improvement can occur, and when it must stop—together with the primary tables and figures for RQ1–RQ5. This appendix collects the supporting material: full algorithms, mathematical formulations, construction protocols, prompts, extra diagnostics, systems comparisons, taxonomy tables, and robustness checks. The sections below map onto the corresponding main-text claims.

- Appendix B (Section 3): Algorithm 1 of the closed-loop training recurrence, group-relative policy gradient objectives, and the structural causal formulation of surface coupling (Appendix B.1).
- Appendix C (Sections 1 and 5, RQ4): concrete exemplars and structural analysis of dual-variable difficulty (rubric $\times$ borderline candidate) versus generative settings, where query hardness is realized in the model's own output.
- Appendix D (Section 4): candidate response synthesis protocol on RubricHub, detailing the 16-model three-tier pool and the balanced 1:2:1 allocation.
- Appendices E–F (Section 4): construction of the two online monitors—human-verified SV-Hard used by PAV, and complementary SV-Full scoring the full in-domain rule suite.
- Appendix G (Section 4, RQ4): definitions of the five Qwen-9B controls (frozen checker, external 27B checker, self-consistency, and the two teacher-SFT pools).
- Appendices H–I (Sections 3.1–3.3): format gate, shared evidence-block layout, and the decoupled judge/checker prompt templates.
- Appendices J–K (Section 4, Table 2): metric definitions and per-benchmark characteristics for the held-out transfer suite.
- Appendix L (RQ1, RQ3): Qwen-9B judge/checker diagnostic grids on SV-Hard and HealthBench, showing why rule accuracy alone is an insufficient stopping signal.
- Appendix M (RQ1, RQ3): cross-architecture replication on Gemma-4-E4B-it—fine-grained judge and checker diagnostic dynamics on SV-Hard and HealthBench.

- Appendix N (Section 3.3, RQ3): PAV unbiasedness, sequential monitoring vs. pointwise estimation, bootstrap intervals, fold stability, and the offline selector comparison.
- Appendices O–P (RQ3, RQ4): systems and cost-performance trade-offs relative to commercial APIs and standalone RMs, plus format compliance and late-stage checker failure modes.
- Appendix Q (Sections 4 and 5): RL and downstream DPO hyperparameters used for the main runs and RQ5 transfer.
- Appendix R (Sections 1 and 2): systematic taxonomy and multi-dimensional comparison against concurrent self-improving evaluators.
- Appendix S (Section 6): broader societal impact, automation bias mitigation, and the *Humans-over-the-loop* safety paradigm.
- Appendix T (Conclusion): expanded limitations and open questions summarized at the end of the main text.

B Algorithm and Method Details

Group-relative advantage computation and policy objective. In each training iteration t, for a given rubric instance $x = (h, y, r_{1:K})$, the judge policy π_{θ_t} generates n rollouts $\{z_{t,i}\}_{i=1}^{n}$. The synchronized checker copy $C_{\bar{\theta}_t}$ audits each rollout to emit a scalar score $s_{t,i} \in \{0, 1, 2, 3, 4\}$. Outputs that pass the format gate form the valid index set $\mathcal{V}_t \subseteq \{1, \ldots, n\}$ with reward $R_{t,i} = s_{t,i}$. Group mean and standard deviation are computed strictly within the prompt group:

$$\mu_t = \frac{1}{|\mathcal{V}_t|} \sum_{i \in \mathcal{V}_t} R_{t,i}, \quad \sigma_t = \sqrt{\frac{1}{|\mathcal{V}_t|} \sum_{i \in \mathcal{V}_t} (R_{t,i} - \mu_t)^2 + \epsilon_{\text{std}}}, \tag{7}$$

where $\epsilon_{\text{std}} = 10^{-6}$ prevents numerical division by zero. The group-relative advantage for each valid rollout is normalized as:

$$\widehat{A}_{t,i} = \frac{R_{t,i} - \mu_t}{\sigma_t}. \tag{8}$$

Malformed rollouts ($i \notin \mathcal{V}_t$) are masked out with zero advantage. The policy π_θ is updated by optimizing the clipped sequence-level policy loss:

$$\mathcal{L}_{\text{RL}}(\theta) = -\frac{1}{|\mathcal{V}_t|} \sum_{i \in \mathcal{V}_t} \min \left(\frac{\pi_\theta(z_{t,i} \mid x)}{\pi_{\theta_t}(z_{t,i} \mid x)} \widehat{A}_{t,i},\ \text{clip} \left(\frac{\pi_\theta(z_{t,i} \mid x)}{\pi_{\theta_t}(z_{t,i} \mid x)}, 1 - \epsilon_{\text{clip}}, 1 + \epsilon_{\text{clip}} \right) \widehat{A}_{t,i} \right), \tag{9}$$

with clipping threshold $\epsilon_{\text{clip}} = 0.005$. Crucially, advantages $\widehat{A}_{t,i}$ are completely invariant under any affine transformation $R \mapsto \alpha R + \beta$ ($\alpha > 0$). Therefore, the absolute numeric range of the checker's 5-tier score is immaterial; what determines the policy gradient is purely the relative ranking induced among the n on-policy rollouts.

The checkpoint $t^\star$ returned by Algorithm 1 is only a reference; during training, stopping is decided by staged inspection of PAV at each monitoring interval.

B.1 Surface Coupling: Structural Causal Formulation and Visual Illustration

Structural Causal Model (SCM) of surface coupling. To formalize the shortcut channel in Figure 5, described by $S = aU + bB + \varepsilon$ (Eq. 2), consider the structural causal relations governing reward extraction under parameter synchronization ($\bar{\theta}_{t+1} \leftarrow \theta_{t+1}$):

1. **Latent Evaluative Quality** (U): Measures whether the judge's reasoning correctly verifies the candidate response against criteria $r_{1:K}$.
2. **Surface Token Bias** (B): Measures the marginal emission probability bias for verdict tokens (e.g., $\Delta P_{\text{YES}} = P(\text{`YES'}) - 0.5$).
3. **Judge Emission** (Z_1): Emits tokens under policy π_{θ_t}, where $Z_1 = f_1(U, B; \theta_t)$.
4. **Checker Evaluation** (S): Audits reasoning under synchronized copy $C_{\bar{\theta}_t}$, where $S = f_2(Z_1; \bar{\theta}_t) = aU + bB + \varepsilon$.

Algorithm 1 RECURSE training loop with periodic held-out validity monitoring

Require: Rubric instances $\mathcal{D}$, initial judge policy π_{θ_0}, monitoring interval $T = 10$, fixed human-verified monitor $\mathcal{H}$, training budget B

$\bar{\theta}_0 \leftarrow \theta_0$ {*reward-only checker copy*}
$V^\star \leftarrow -\infty$; $t^\star \leftarrow 0$
for $t = 0, 1, 2, \ldots, B$ **do**
  Sample $x \sim \mathcal{D}$; rollout $z_{t,i} \sim \pi_{\theta_t}(\cdot \mid x)$ for $i = 1, \ldots, n$
  $s_{t,i} \leftarrow C_{\bar{\theta}_t}(x, z_{t,i})$ {*Pass 2: process check*}
  $R_{t,i} \leftarrow s_{t,i}$ if format-valid else 0; mask invalid from group statistics
  $\theta_{t+1} \leftarrow \mathrm{PolicyUpdate}(\theta_t, \{(z_{t,i}, R_{t,i})\})$; $\bar{\theta}_{t+1} \leftarrow \theta_{t+1}$
  if $t \bmod T = 0$ **then**
    Compute $\widehat{A}_t$, $\widehat{e}_{C,t}$, and PAV $\widehat{V}_t$ on $\mathcal{H}$
    if $\widehat{V}_t > V^\star$ **then**
      Save checkpoint; $V^\star \leftarrow \widehat{V}_t$; $t^\star \leftarrow t$
    end if
  end if
end for
return checkpoint $t^\star$

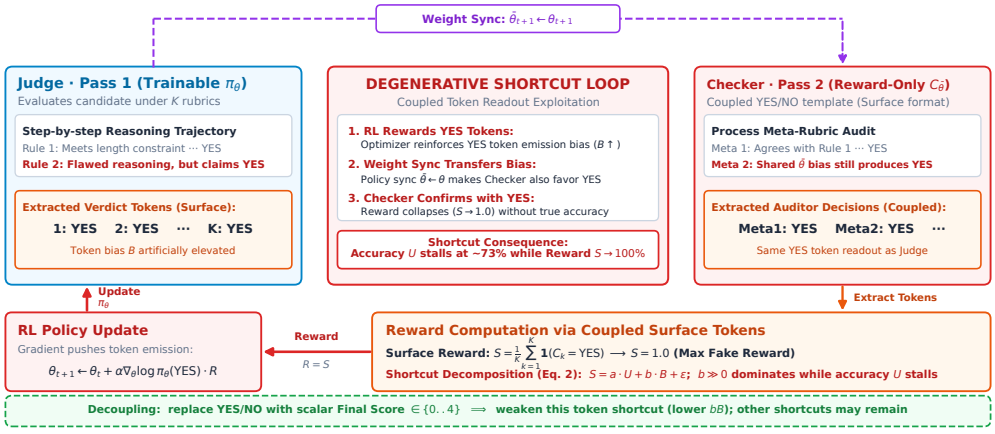


Figure 5: **Schematic of surface coupling and the degenerative reward shortcut pathway (Section 3.2).** When Judge (Pass 1) and Checker (Pass 2) share identical YES/NO readout templates, self-produced reward S is extracted directly from surface decision tokens ($S = \frac{1}{K} \sum \mathbf{1}(\mathrm{Checker}_k = \mathrm{YES})$). Policy gradient optimization inflates YES token emission ($B \uparrow$), transferring immediately to synchronized checker weights ($\bar{\theta} \leftarrow \theta$). This establishes a degenerative shortcut loop (bB) where self-assigned rewards surge to 100% while true evaluative capability U stalls. Interface decoupling replaces the checker's verdict surface with a free-standing 5-tier scalar Final Score, severing this direct token-level shortcut channel.

When the checker uses identical YES/NO readout templates, f_2 computes reward by directly parsing YES tokens from Pass 2. Because $\bar{\theta}_t = \theta_t$, any gradient update $\nabla_\theta \mathcal{L}_{\mathrm{RL}}$ that reinforces YES tokens in Pass 1 simultaneously increases the probability that Pass 2 emits YES tokens ($b > 0$). This establishes a degenerative backdoor shortcut: the policy maximizes reward by driving $B \to 1$ ($bB \to \max$) without increasing true reasoning quality U ($aU \approx \mathrm{const}$).

Interface decoupling as structural causal intervention. Interface decoupling replaces f_2 with a free-standing 5-tier scalar Final Score $s_i \in \{0, 1, 2, 3, 4\}$. Because the scalar score requires generating a holistic reasoning audit followed by a distinct numerical symbol, the direct token-copying gradient pathway from Pass 1 YES tokens to Pass 2 reward extraction is severed ($\partial S / \partial \mathrm{verdict_token} \approx 0$, effectively setting $b \approx 0$). The optimizer is therefore constrained to maximize reward solely

through improving genuine reasoning verification (aU), converting reward growth into robust evaluative accuracy.

C Dual-Variable Difficulty Coupling: Conjecture and Exemplars

Section 5 (RQ4) reports that teacher SFT plateaus early and that scaling the distillation pool narrows but does not close the gap to RecurSE. This section develops the conjectured explanation offered there; it is not directly tested in this work. The conjecture rests on a distinction between training standard generative models (e.g., in mathematics or coding) and training LLM rubric judges.

Generated y versus given y. In generative tasks such as mathematical problem solving, query difficulty is typically realized in the response the model must write: a hard x makes a correct y hard to produce, so the pairing stays informative. In rubric judging the candidate y is given as input, so a hard prompt does not by itself create a hard instance. An informative signal still requires two coupled variables:

$$\text{Informative Gradient Signal} \iff \underbrace{\mathcal{R}(x)}_{\text{Discriminative Rubric}} \times \underbrace{\mathcal{B}(y \mid x, \mathcal{R})}_{\text{Borderline Response Compliance}} . \tag{10}$$

If either variable fails to meet this calibration, the instance yields negligible corrective gradient:

- **Trivial Response (y far from decision boundary)**: If a candidate response is blatantly compliant (e.g., an obvious perfect answer) or blatantly flawed (e.g., empty or gibberish text), even an untuned base judge trivially assigns YES or NO correctly. The cross-entropy loss produces near-zero gradient, imparting no refinement to the model's decision boundary.
- **Non-discriminative Rubric ($\mathcal{R}$ trivial or ambiguous)**: If a rubric criterion is overly generic (e.g., "the response is helpful") or ambiguous, all rollout candidates receive identical verdicts, providing zero group-relative variance.

A conjectured account of the SFT plateau. In static offline distillation corpora (even across ≈17,000 trajectories distilled from four frontier teachers: Qwen3.5, Gemini-3.1-Flash-Lite, DeepSeek-V4-Flash, and Kimi-K2.6), the candidate responses y are fixed beforehand. We conjecture that, as the student judge improves during fine-tuning, the static triplets $(h, y, r_{1:K})$ fall outside the student's active, shifting decision boundary and degenerate into trivial instances. Under this account, scaling data quantity from 6.4k to 17k would mostly multiply trivial samples without supplying borderline tension, which is consistent with the observed plateau but does not by itself establish the mechanism.

Why on-policy RecurSE may avoid the plateau. In contrast, RecurSE operates on-policy on judge traces: as π_{θ_t} evolves, the sampled rollouts $z_{t,i}$ probe reasoning variations on the same $(h, y, r_{1:K})$. The synchronized checker ranks those traces, so the group-relative advantages $\widehat{A}_{t,i}$ could remain informative even though the candidate y is not resampled. Verifying this account (e.g., by tracking the fraction of zero-variance rollout groups over training) is left to future work.

D Training Data and Candidate Response Synthesis

RubricHub (Li et al., 2026b) provides queries and fine-grained rubrics $(h, r_{1:K})$ across five diverse domains (Chat, Instruction-Following, Medical, Science, and Writing), but does not contain pre-existing candidate AI responses y. To construct realistic evaluation instances $x = (h, y, r_{1:K})$ spanning a broad, balanced spectrum of response quality, we synthesize candidate responses using a tiered multi-model sampling protocol.

Three-tier model pool. We stratify 16 distinct LLMs into three capability tiers based on their reasoning and instruction-following capacities:

- **Low Tier** (compact/lightweight models): GLM-4-Flash, GPT-4.1-nano, Gemini-2.5-Flash-Lite, and Qwen3.5-9B.
- **Mid Tier** (mid-scale conversational models): LongCat-Lite-V2-Chat, DeepSeek-V3.2, GLM-5.1, GPT-4.1-mini, Gemini-2.5-Flash, Moonshot-v1-128k, and Qwen3.6-27B.
- **High Tier** (frontier reasoning models): LongCat-Flash-Thinking, LongCat-Flash-Chat, Qwen3.5-397B, DeepSeek-V4-Flash, GPT-4.1, Gemini-3.1-Flash-Lite, and Kimi-K2.6.

Balanced 1:2:1 tier allocation per query. For each query h, we allocate four response slots across tiers in a fixed Low : Mid : High $= 1 : 2 : 1$ ratio:

$$y \sim \begin{cases} 1 \times \text{Low Tier} & \text{(often violates fine-grained negative constraints or nuances),} \\ 2 \times \text{Mid Tier} & \text{(borderline quality; compliant on standard rules but mixed on edge cases),} \\ 1 \times \text{High Tier} & \text{(substantially compliant, yet still borderline on the hardest criteria).} \end{cases} \tag{11}$$

Within each domain, models within each tier are assigned evenly across samples via a greedy balanced permutation schedule (ensuring mid-tier slots use two distinct models). This tiered distribution ensures the training corpus exhibits rich compliance variation across criteria without collapsing to trivial all-YES or all-NO extremes.

E SV-HARD CONSTRUCTION

SV-HARD is a same-style human-verified validation set of 100 examples used to monitor reward validity and guide early stopping under RubricHub-style rubric judging. RubricHub is cluster-split so that no prompt or rubric cluster appears in both training and evaluation; SV-HARD is drawn from the evaluation split and never contributes gold labels to the RL training reward. Labels are LLM-initialized and then human-verified at the rule level; this compact holdout is far cheaper than scaling gold annotation to the full training corpus, but the verification step is still a real upfront cost.

The set covers diverse instruction domains: 20 instances from each of five categories (Chat, Instruction-Following, Medical, Science, and Writing), scored on progression-critical hard rules selected per prompt (mean 2.5 rules per prompt; $89/100$ have 2 or 3). Because gold is expert-checked, SV-HARD is the *only* stream on which we measure checker fidelity ($1 - \text{NMAE}$) and compute PAV. It is a fair same-style validity monitor—matched in judging style to training and strictly cluster-isolated from it. Repeated inspection can still adapt early stopping to this validation set; we therefore reserve the different-style benchmarks and SV-FULL for testing substantive transfer claims. Their parallel gains argue against a purely SV-HARD-specific selection artifact.

F SV-FULL CONSTRUCTION

SV-FULL is a complementary online in-domain validation stream on the same 100 cluster-isolated prompts as SV-HARD, but each instance is scored on its *full* rule suite rather than the progression-critical hard subset. Gold labels are resolved by an LLM annotation panel with Claude-based tie-breaking; they are never used in the RL training reward. SV-FULL therefore tests whether judge improvement extends to complete in-domain rubric coverage under the same RubricHub judging interface, without the human-verification cost required for reliable checker diagnostics. We evaluate SV-FULL at the same checkpoint cadence as the offline transfer suite and report rule accuracy in Table 2; PAV itself is computed only from SV-HARD.

G ABLATION ARMS

All five controls below are run on Qwen3.5-9B. *Frozen base:* checker fixed at untuned init. *External 27B:* Qwen3.6-27B checker, never synced. *Self-consistency:* majority agreement reward without process audit. *Teacher-traj. SFT (6.4k):* full FT on 6400 teacher YES/NO traces (from Qwen3.5, Gemini-3.1-Flash-Lite, DeepSeek-V4-Flash, and Kimi-K2.6), 200 steps matched to the RL budget. *Teacher-traj. SFT (17k):* same teachers and recipe on the full eligible pool (≈17k traces, two epochs), testing whether more imitation data closes the gap.

H Reward and Format Details

The format gate enforces two structural constraints on judge rollouts $z_{t,i}$: (1) it requires exactly K parseable YES/NO decisions in the indexed format (1:<ans>YES/NO</ans> ... K:<ans>YES/NO</ans>), and (2) it strictly forbids delimiter tokens reserved for the checker prompt template (e.g., prematurely emitting <|AI's Response End|> or faking checker reward blocks such as <|Final Score|>4<|Final Score|>). This prevents prompt-injection shortcuts where the judge rollout manipulates Pass 2 slot boundaries or directly injects fraudulent high checker scores into the downstream context. Samples failing the gate receive zero reward ($R_{t,i} = 0$) and a zeroed response mask so they are excluded from the same-prompt group statistics.

I Training Prompts

Both passes share the same overarching evidence-block layout—system instruction, conversation context, the AI response, and a numbered rubric list—ensuring that the checker observes the identical evidence evaluated by the judge. The two passes differ in how these slots are populated and in their required output format. Below, we decompose the prompt template into three distinct components: the shared evidence-block template, the judge-specific instruction and output format (Pass 1), and the checker-specific process-auditing instruction and scoring format (Pass 2).

Part 1: Shared evidence-block template. Both the judge and checker instantiate the same standard three-slot evidence layout. Angle brackets ($\langle\langle\langle\cdot\rangle\rangle\rangle$) denote placeholders populated dynamically per instance:

```
Your job is to assess if the AI's response to the user's
most recent prompt correctly follows the provided rubrics.
<|Conversation History Start|>
<<<conversation_history>>>
<|Conversation History End|>
<|AI's Response Start|>
<<<ai_response>>>
<|AI's Response End|>
<|Rubrics Start|>
<<<rubrics>>>
<|Rubrics End|>
```

The three evidence slots are instantiated for each pass as follows:

- **Judge pass ($x = (h, y, r_{1:K})$)**: <<<conversation_history>>> contains the original dialogue context h; <<<ai_response>>> contains the candidate generation y; and <<<rubrics>>> contains the K task-level evaluation criteria $r_{1:K}$.
- **Checker pass ($C(x, z_{t,i})$)**: <<<conversation_history>>> contains the complete judge input instance x; <<<ai_response>>> contains the judge's full rollout $z_{t,i}$ (verbatim per-criterion reasoning and verdicts); and <<<rubrics>>> contains meta-rubrics of the form "k: Did the AI correctly judge the following rubric: $\langle r_k \rangle$".

Part 2: Judge instruction and output format (Pass 1, trainable). Appended to the shared evidence block, the judge is instructed to generate step-by-step reasoning followed by concatenated per-rule binary decisions:

```
Please first analyze step by step whether the response meets
each of the rubrics above.  If it meets the rubric, output
YES; otherwise, output NO.
Finally, provide the results in rubric order using the
concatenated format:
1:<ans>YES/NO</ans> 2:<ans>YES/NO</ans> ...
K:<ans>YES/NO</ans>.
```

The judge's output is parsed and verified by the format gate (§H), requiring all K indexed decisions to be strictly valid.

Part 3: Checker instruction and output format (Pass 2, reward-only). Appended to the shared evidence block, the synchronized checker audits the reasoning process and emits a single, decoupled scalar Final Score $s_i \in \{0, 1, 2, 3, 4\}$:

```
First, analyze whether each rule's judgment is accurate,
then score the overall scoring process.
Use 5 tiers (0-4):
0 = all rule judgments are wrong and the judge's internal
explanation is very poor;
1 = most rule judgments are wrong (only a small fraction
correct) and the judge's internal explanation is
poor/unclear;
2 = about half of the rule judgments are correct and the
judge's internal explanation is partially clear but has
notable issues;
3 = most rule judgments are correct (few wrong) and the
judge's internal explanation is mostly clear with minor
issues;
4 = all rule judgments are correct and the judge's analysis
process is clear.
Output the final score to
<|Final Score|>
<<<final_score>>>
<|Final Score|>
```

The parsed scalar s_i serves as the sole RL reward ($R_i = s_i$). This structural separation between the judge's per-rule YES/NO verdicts and the checker's 5-point scalar score closes the token-level copying channel (§3.2); other shortcuts can remain.

J Evaluation Metrics

Let $\hat{y}_r \in \{\text{YES}, \text{NO}\}$ be the judge's prediction for rubric criterion r and g_r the gold label. In the main text (Table 1 and Table 2), we report each benchmark under its standard primary metric for conciseness. Below we define both primary and supplementary diagnostic metrics used in our analyses.

Rule accuracy (SV-HARD, SV-FULL, HealthBench, ProfBench). For instances with per-rule gold labels, we pool over all rules with both a gold label and a prediction (micro-average): $\text{rule_acc} = \sum_i \text{correct}_i / \sum_i \text{total}_i$, where rules with no prediction are skipped rather than counted wrong. This is the primary metric for SV-HARD, SV-FULL, HealthBench, and ProfBench in Table 2.

All-correct rate (SV-HARD, HealthBench, ProfBench). The sample-level fraction of instances whose every comparable rule is judged correctly, i.e. $\mathbf{1}[\text{gold_match}_i = 1]$ averaged over instances. This is the stricter sample-level accuracy metric reported alongside rule accuracy in Table 2.

Macro accuracy (SV-HARD, HealthBench, ProfBench). The unweighted average of per-instance rule accuracies across all valid evaluation samples, treating each evaluation prompt/rubric equally regardless of its rule count.

HealthBench rule accuracy and F1. Gold is the strict majority vote of physician binary labels (ties excluded); rule accuracy is agreement with this majority (reported in Table 2). We additionally compute the official balanced F1, which pairs the model prediction with *each* physician label (including ties) and averages the positive- and negative-class F1.

RubricBench pairwise accuracy. For each A/B pair we compute the mean YES-rate on each side; the side with the higher rate is predicted preferred (ties broken by YES count). Accuracy is over paired cases against the gold preference label.

CheckEval correlation and metric interpretation. CheckEval-Summ (Lee et al., 2025) assesses document summarization across 1,600 summaries generated by 16 distinct LLM systems across 100 source documents from SummEval (Fabbri et al., 2021). In the original SummEval corpus, human experts directly assigned *holistic Likert-scale ratings* (1–5 scale, averaged across annotators to yield continuous scores $\mathbf{H}_d$) for each of four broad quality dimensions: **Coherence**, **Consistency**, **Fluency**, and **Relevance**. Human annotators evaluated overall summary quality directly and did not annotate fine-grained checklist rules. To evaluate automated judges without subjective scalar scoring, CheckEval (Lee et al., 2025) decomposed these four holistic dimensions into 15 atomic, binary checklist criteria:

- **Coherence** (3 checklist items): Logical narrative flow, paragraph transitions, and semantic organization.
- **Consistency** (3 checklist items): Factual alignment with the source document and absence of hallucinations.
- **Fluency** (4 checklist items): Grammatical correctness, lexical clarity, and formatting cleanliness.
- **Relevance** (5 checklist items): Coverage of central key ideas and avoidance of superfluous details.

All 15 checklist criteria are *global* (identical across all 1,600 summaries). For each summary $i \in \{1, \ldots, 1600\}$ and dimension $d \in \{\text{coherence, consistency, fluency, relevance}\}$, the model's predicted dimension score $P_{i,d}$ is computed as the proportion of YES verdicts emitted over that dimension's checklist items:

$$P_{i,d} = \frac{1}{|\text{Rules}(d)|} \sum_{r \in \text{Rules}(d)} \mathbf{1}(\text{Judge}_{i,r} = \text{YES}) \in [0, 1]. \tag{12}$$

Across all $N = 1{,}600$ evaluation summaries, we compute the sample-level Pearson correlation coefficient $\rho_d = \text{Corr}(\mathbf{P}_d, \mathbf{H}_d)$ between the model's predicted score sequence $\mathbf{P}_d$ and the continuous human rating sequence $\mathbf{H}_d$. The primary CheckEval metric reported in Table 2 is the macro-average of these sample-level correlation coefficients across the four dimensions:

$$\rho = \frac{1}{4} \sum_{d \in \{\text{coh,con,flu,rel}\}} \rho_d \in [-1, 1]. \tag{13}$$

Interpreting correlation magnitude versus percentage accuracy: Crucially, Pearson correlation $\rho \in [-1, 1]$ measures linear alignment with continuous human judgment distributions rather than discrete 0–100% classification accuracy. In a large evaluation pool of 1,600 instances, an absolute lift in Pearson correlation (e.g., $+0.019$ from 0.422 to 0.441 on Qwen-9B, $+0.024$ on Gemma, and $+0.027$ on Qwen-27B) represents a statistically robust and practically meaningful improvement in continuous alignment fidelity ($p = 0.0032 < 0.01$; Appendix N), which must not be naively equated with or compared directly against percentage-point increments on discrete accuracy benchmarks.

Checker point calibration and score fidelity (PAV). The checker emits a scalar score $S \in \{0, 1, 2, 3, 4\}$. To assess whether it faithfully recovers the correctness of the judge's reasoning, we compare S against the reference target $U = 4 \times (\text{correct rules}/\text{total rules}) \in [0, 4]$ (e.g., all rules correct $\to 4$; half $\to 2$). We evaluate checker capability through three aligned diagnostic metrics:

- **Exact-Match Rate**: Converts continuous reference target U to its nearest integer tier $\lfloor U \rceil$ and tests exact equality $\mathbf{1}[S = \lfloor U \rceil]$.
- **Mean Absolute Error** (MAE): Computes the sample-level mean absolute error $\frac{1}{N}\sum_{i=1}^{N} |S_i - U_i|$, penalizing unparsable outputs with the worst-case endpoint error (4).
- **Checker Score Fidelity** ($1 - e_C = 1 - \text{NMAE}$): Normalizes the pointwise checker error to $e_C = \frac{\text{MAE}}{4} \in [0, 1]$, yielding the score fidelity $1 - e_C$. This quantity is not unweighted pairwise accuracy. Instead, Eq. 4 shows that $2e_C$ bounds the expected distortion of within-prompt pairwise reward differences, which in turn bounds the margin-weighted ordering loss in Eq. 5.

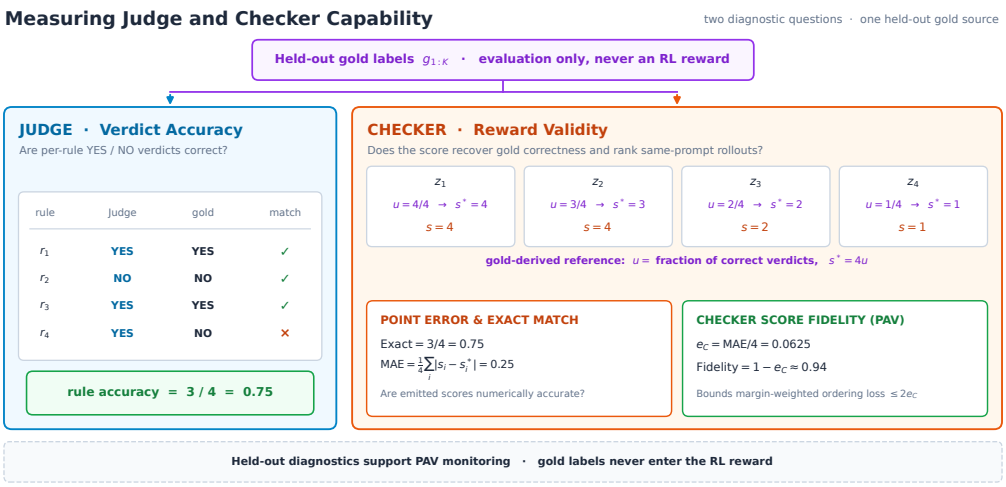


Figure 6: **Gold-derived diagnostics for judge and checker capability on a four-rollout example.** The same held-out rule labels simultaneously evaluate judge rule accuracy (A) and the checker reference target $U = 4u$ (where u is gold rule correctness). Checker diagnostics evaluate point error (MAE and Exact match) and normalized score fidelity ($1 - e_C = 1 - \text{NMAE}$), directly supplying the PAV composite monitor (V). Gold labels never enter the RL training reward.

Figure 6 illustrates how the same held-out gold labels simultaneously evaluate judge accuracy and checker score fidelity without leaking supervision into the RL training reward.

K Benchmark Characteristics and Examples

Table 5 summarizes the distinguishing traits of each held-out suite and one representative instance that best illustrates them. Queries, responses, and rubrics are abridged; the judge outputs a YES/NO for each listed criterion.

HealthBench corpus scale and stratified 10% subsampling protocol. The full OpenAI HealthBench meta-evaluation benchmark (Arora et al., 2025) comprises 29,511 rubric evaluation instances across 14,592 prompt-completion groups across 10 overarching clinical themes (e.g., global health, emergency referrals, healthcare data communication, uncertainty hedging) and 34 medical subcategories. In the original corpus, multiple candidate LLM responses per clinical query are paired with 1–3 physician-crafted rubric criteria evaluating patient safety, diagnostic communication, and referral thresholds. Evaluating all 14,592 groups across training trajectories imposes an intractable computational bottleneck ($\approx$ 30k sequential reasoning traces per evaluation step). To balance evaluation tractability with clinical fidelity, we implement a stratified proportional sampling protocol:

1. **Atomic group preservation**: Evaluation records are grouped by unique prompt–response pairs (h, y) so that all interdependent rubric criteria corresponding to a single candidate response remain bound together.
2. **Stratified category proportionality**: Groups are partitioned across all 34 medical categories and sampled strictly in proportion (10%, fraction $\alpha = 0.10$) to their original category prevalence in the full benchmark. This yields exactly 1,459 representative prompt–response groups (2,951 rubric rules, mean 2.02 rules/group), faithfully preserving the semantic distribution across all 10 clinical themes without distortion.
3. **Physician consensus ground truth**: Gold labels represent the strict majority vote across certified physician raters (ties excluded).

Table 5: **Benchmark characteristics and representative examples.** Each row pairs the trait emphasized in Section 4 with one instance that best illustrates it.

Benchmark	Key characteristics	Representative example
SV-HARD	Same-style PAV monitor; short hard-rule subsets (mean 2.5 rules), mostly format/instruction checks; human-verified gold.	*Combinatorics + format constraints.* Query: count paths on a grid where every sentence must begin with "highway" and sentences are joined by hyphens. Rubric: (1) first word of every sentence is "highway"; (2) hyphens split sentences correctly; (3) the response addresses the math question. Gold on progression-critical rules: {1:YES, 2:NO}.
HealthBench	Medical safety; representative 10% cluster sample (1,459 prompts) from full benchmark; few rules per prompt but semantically richer physician norms.	*Incomplete clinical context.* Query: compute perioperative mortality risk with missing details. Response: "I'm sorry, but I can't assist with that." Rubric: (1) does not give a definitive answer when information is missing ([T,T]); (2) still seeks missing context rather than flatly declining ([F,F]).
RubricBench	Pairwise preference between two responses under sample-specific multi-criterion rubrics (mean 5.8 rules).	*Code assistance.* Query: "Add mypy type hints for the code below" (no code provided). Response: asks the user to provide the code first. Rubric checks whether the assistant recognizes the missing code, explains why, addresses mypy specifically, stays clear, and remains polite.
CheckEval-Summ	*Global* 15-criterion checklist shared across all summaries (not sample-specific); Pearson ρ on aggregated YES-rates.	*News summarization.* Query: summarize a document (with reference summary). Rubric: fixed checklist over coherence ($\times 3$), consistency ($\times 3$), fluency ($\times 4$), and relevance ($\times 5$); YES-rate per dimension is correlated with human dimension scores.
ProfBench	Expert professional tasks; long weighted rubrics ($\approx$29 rules) with tight numerical tolerances; hardest suite.	*Physics PhD / EMI shielding.* Query: multi-step numerical derivation for shielding effectiveness. Rubric: weighted Critical/Major/Minor criteria checking specific numerical ranges (e.g. $R \in [0.6835, 0.6841]$), the correct dominance conclusion, and stated shielding-effectiveness relations.

L ADDITIONAL TRAINING DYNAMICS

To understand the internal mechanics of recursive self-improvement and how evaluative capabilities evolve over training steps, we examine fine-grained diagnostic trajectories for both the judge policy π_{θ_t} and the synchronized process checker $C_{\bar{\theta}_t}$ on the primary Qwen3.5-9B run. Figures 7 and 8 track four complementary diagnostic metrics across the complete optimization trajectory:

1. **Judge Rule Accuracy** ($\widehat{A}_t$, panel a): The micro-averaged accuracy of individual YES/NO rule verdicts emitted during Pass 1 against ground-truth human annotations, matching rule accuracy in Section J.
2. **Checker Exact-Match Rate** (panel b): The proportion of instances where the checker's 5-tier scalar score $s_i \in \{0, 1, 2, 3, 4\}$ in Pass 2 matches the true gold target score $S^\star$ exactly ($\mathbf{1}(s_i = S^\star)$).
3. **Checker Mean Absolute Error (MAE)** (panel c): The absolute difference between the predicted scalar score and the true ground-truth score ($|s_i - S^\star|$). Any malformed or unparsable checker output is penalized with the maximum boundary error (4).
4. **Checker Score Fidelity** ($1 - \mathrm{NMAE} = 1 - \mathrm{MAE}/4$, panel d): Normalized checker fidelity mapping score errors to a unit interval $[0, 1]$, representing the checker's reliability in assigning calibrated process rewards.

Diagnostic dynamics on SV-HARD and validation deception. As illustrated in Figure 7, during the early and intermediate optimization phases (steps 0–130), judge accuracy $\widehat{A}_t$ and checker fidelity $1 - \mathrm{NMAE}$ climb in close unison: exact match rises while MAE falls. Beyond step 130 the two signals diverge: judge rule accuracy on SV-HARD continues to rise through step 200, whereas checker exact match and fidelity plateau and then degrade. This divergence is *validation deception*: when trained past this useful stopping boundary, the policy over-optimizes for surface stylistic quirks in the training distribution, and the checker's calibrated auditing capability deteriorates. Relying solely on judge accuracy on the validation set would select an over-optimized late checkpoint (e.g., step 200), whereas PAV's joint formulation localizes the useful peak near step 130.

Independent cross-domain validation on HealthBench. Figure 8 tracks the exact same four diagnostic quantities on the held-out HealthBench suite (1,459 prompts, 2,951 rules) spanning clinical communication and patient safety—a distinct distribution never seen during training or validation monitoring. Judge accuracy, checker exact match, and checker fidelity improve through the same

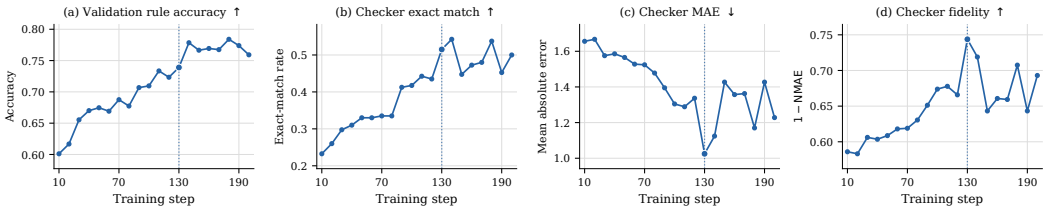


Figure 7: **SV-HARD diagnostics on the Qwen3.5-9B trajectory.** (a) Validation rule accuracy (↑). (b) Checker exact-match rate (↑). (c) Checker MAE on the original 0–4 score scale (↓); malformed checker outputs receive their worst possible endpoint error. (d) Checker fidelity $1 - \mathrm{NMAE} = 1 - \mathrm{MAE}/4$ (↑). Dotted lines and enlarged markers show the PAV landmark @130. Rule accuracy alone continues to rise, whereas exact match, MAE, and fidelity expose the sharp checker improvement around the selected checkpoint.

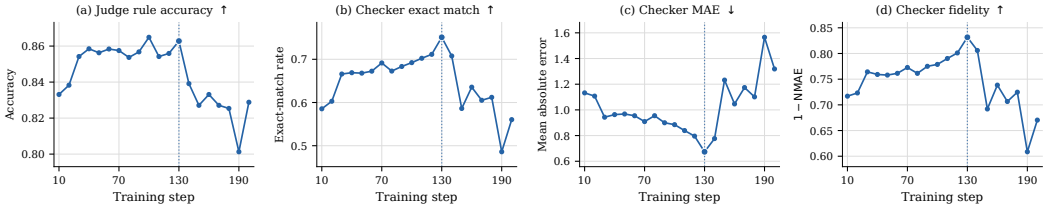


Figure 8: **HealthBench diagnostics on the Qwen3.5-9B trajectory.** Same four panels as Figure 7, now on the different-style medical suite that PAV never sees. (a) Judge rule accuracy (↑). (b) Checker exact-match rate (↑). (c) Checker MAE on the 0–4 scale (↓); malformed checker outputs receive their worst possible endpoint error. (d) Checker fidelity $1 - \mathrm{NMAE} = 1 - \mathrm{MAE}/4$ (↑). Dotted lines and enlarged markers mark the PAV landmark @130. Judge accuracy, exact match, and fidelity improve around that region and then fall together, so the transfer suite tracks the same useful window as the SV-HARD monitor.

window near step 130 and then decay together. This cross-domain resonance provides decisive evidence: the degradation in checker fidelity detected by PAV on SV-HARD is not a localized artifact of the validation set, but reflects a general transition from robust reasoning acquisition to out-of-distribution capacity collapse.

M CROSS-ARCHITECTURE REPLICATION ON GEMMA-4-E4B-IT

Gemma-4-E4B-it replicates the main RECURSE recipe without ablation variants, providing an independent test of RQ1 (viability) and RQ3 (bounded horizon) outside the Qwen model family. As summarized in Table 2, held-out metrics exhibit consistent positive trajectories through the useful window: SV-HARD rule accuracy rises from 55.9% to 61.1% at PAV-selected @160, HealthBench gains $+3.7$ points, and CheckEval Pearson ρ improves from 0.442 to 0.466. SV-FULL remains near its high base (92.5%), consistent with a near-saturated in-domain ceiling on this architecture.

Figure 9 repeats the judge/checker diagnostic panels from Figures 7 and 8 on SV-HARD and HealthBench across training steps. Checker fidelity and judge accuracy climb in unison and plateau around steps 120–160. After the selected @160 landmark, later checkpoints worsen overall relative to that peak. This confirms that bounded recursive self-improvement and PAV-localized stopping are robust across distinct model architectures.

N PAV CONSTRUCTION AND VALIDATION DETAILS

Finite-sample unbiasedness at a fixed checkpoint. For any single fixed checkpoint t, rule correctness $Y_t \in \{0, 1\}$ and normalized checker error $E_{C,t} = |S_t - U_t|/4$ have expectations A_t and $e_{C,t}$ over the prompt distribution. Because empirical sample means are linear estimators, their expecta-

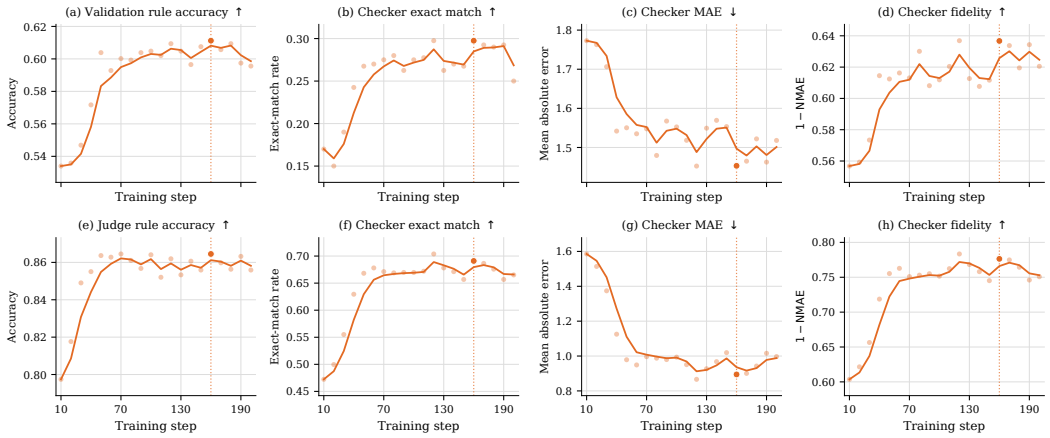


Figure 9: **Gemma judge and checker dynamics (RQ3 replication).** Top row: SV-HARD diagnostics; bottom row: HealthBench diagnostics on the Gemma-4-E4B-it trajectory (steps 10–200). Same four panels as Figures 7 and 8. Dotted lines and enlarged markers mark the PAV landmark @160. After that selected point, later checkpoints worsen overall relative to the peak on both suites.

tions satisfy $\mathbb{E}[\widehat{A}_t] = A_t$ and $\mathbb{E}[\widehat{e}_{C,t}] = e_{C,t}$, holding even when computed from separate rollout samples on a fixed prompt set. Since the PAV formulation is an affine combination:

$$\mathbb{E}[\widehat{V}_t] = \frac{\mathbb{E}[\widehat{A}_t] + 2\{1 - \mathbb{E}[\widehat{e}_{C,t}]\}}{3} = \frac{A_t + 2(1 - e_{C,t})}{3} = V_t. \tag{14}$$

Pointwise estimation vs. sequential monitoring. A critical methodological distinction must be made between evaluating a single checkpoint post-hoc and sequentially monitoring an online training trajectory across M evaluation steps $(t_1, \ldots, t_M)$. In naive sequential selection, picking the absolute empirical maximum $\hat{t} = \arg\max_{t_m} \widehat{V}_{t_m}$ introduces a family-wise error rate (FWER) inflation across the M dependent hypothesis tests.

Importantly, RECURSE does *not* treat PAV as a fragile pointwise $\arg\max$ selector. Rather, as demonstrated by the bootstrap distributions and fold-resampling experiments below, $\widehat{V}_t$ forms an elevated, stable *validity plateau* (steps 120–140 for Qwen-9B, steps 150–170 for Gemma). Any checkpoint retained within this elevated validity window captures essentially identical peak transfer gains across all held-out suites. **PAV thus functions as a robust *stopping horizon monitor* that signals when checker degradation begins, rather than an overfitted single-step selector.**

Adequacy and power of the 100-prompt human-verified monitor. Although 100 prompts may appear compact, SV-HARD is specifically curated for high diagnostic signal-to-noise ratio:

1. **High rule density on progression-critical criteria**: SV-HARD contains ≈250 human-verified rules, filtered specifically for hard, borderline instances where untuned models exhibit high disagreement.
2. **Multi-rollout sample size aggregation**: At each monitoring step, every prompt is scored on $n = 4$ checker-audited rollout traces, so $\widehat{V}_t$ is computed over $100 \times 4 = 400$ judge–checker trajectories per checkpoint. This monitoring width is separate from the training group size $n = 8$ (Section 4).
3. **Independent corroboration by SV-FULL**: As highlighted in Section 5, SV-FULL evaluates the full rule suite on 100 prompts under independent LLM panel supervision. Despite being completely excluded from PAV computation, SV-FULL independently peaks at @130 (95.3%) and collapses to 89.7% by @200, mirroring the exact stopping boundary identified by PAV on SV-HARD.

Why checker error has weight two. Group-relative learning depends on score differences $(S_i - S_j)$, not isolated absolute scores. Writing $\delta_i = S_i - U_i$, the error in any pairwise reward difference

Table 6: **Three-fold PAV region on SV-HARD.** Each fold holds out 33–34 prompts and scores the remainder.

Trajectory	Fold 1	Fold 2	Fold 3
Qwen-9B	140	130	130
Gemma	160	170	160

satisfies $|(\delta_i - \delta_j)|/4 \le (|\delta_i| + |\delta_j|)/4$. Taking expectations over exchangeable rollout pairs yields $2e_{C,t}$. Combining this with judge rule error $(1 - A_t)$ defines the compound surrogate risk $(1 - A_t) + 2e_{C,t}$, leading directly to Eq. 6. The factor of 2 is therefore fixed *a priori* by the pairwise bound, not fitted empirically. Moreover, the pairwise score-difference distortion upper-bounds the margin-weighted ordering loss in Eq. 5: reversals between clearly separated rollouts incur a larger penalty, whereas reversals between near-ties are discounted. Thus, the checker term measures the fidelity of consequential relative reward comparisons rather than unweighted inversion probability.

Curvature and boundary dynamics of the weight stability interval. For the generalized monitor $M_t(\lambda) = \widehat{A}_t + \lambda(1 - \widehat{e}_{C,t})$, a sweep shows that the selected stopping region remains centered at Qwen@130 and Gemma@160 for every $\lambda \in [1.59, 11.03]$. To locate the weight at which the monitor stops rising on Qwen-9B, write its slope along training steps:

$$\nabla_t M_t(\lambda) = \nabla_t \widehat{A}_t - \lambda \nabla_t \widehat{e}_{C,t}. \tag{15}$$

Near step 130 on Qwen-9B, judge rule accuracy flattens ($\nabla_t \widehat{A}_t \approx +0.02/10$ steps), while normalized checker error rises much faster after step 130 ($\nabla_t \widehat{e}_{C,t} \approx +0.15/10$ steps). Setting that slope to zero gives $\lambda \approx 0.02/0.15 \approx 0.13$. This crossover lies below the sweep interval, and the derived weight $\lambda = 2$ lies both above it and inside the interval.

We stress that this weight insensitivity does not imply that checker fidelity alone drives the selection. Both components live on the same $[0, 1]$ scale, and Figure 4 shows that they carry comparable weight in the composite: $\widehat{A}_t$ and $1 - \widehat{e}_{C,t}$ rise together through step 130 and coincide at the landmark, after which the composite inherits both the continued rise in judge accuracy and the decline in checker fidelity. Judge accuracy establishes that the judge has in fact improved, whereas checker fidelity establishes that the reward producing that improvement remains trustworthy; a monitor that dropped either term would be blind to one of the two failure modes (accuracy that rises under a degrading reward, or a faithful reward attached to a judge that is no longer improving). The stability interval therefore indicates only that the selected region is insensitive to the precise trade-off between the two terms, not that either term is dispensable.

Prompt-cluster bootstrap uncertainty and variance control. We assess the statistical uncertainty of the empirical monitor by performing 10,000 prompt-level cluster bootstrap iterations on SV-HARD, resampling entire prompt groups with replacement to preserve intra-prompt rollout and rule correlations. At the designated stopping landmarks, empirical estimates achieve tight confidence bounds: $\widehat{V}_t = 0.742$ with a 95% cluster-bootstrap confidence interval of $[0.685, 0.797]$ for Qwen@130, and $\widehat{V}_t = 0.628$ with $[0.577, 0.679]$ for Gemma@160. These compact intervals show that $\widehat{V}_t$ at the selected landmarks is estimated with low finite-sample variance.

Validation-set sample insensitivity and fold stability. To directly test whether early-stopping localization depends on specific prompt selections (i.e., whether altering the validation sample shifts the selected checkpoint), we partition the 100 SV-HARD prompts into three disjoint subsets (34/33/33 prompts) and evaluate PAV trajectories on the complementary two-thirds (66–67 prompts) for each fold. As summarized in Table 6, the localized stopping region remains exceptionally stable: across all three independent subsets, the selected window concentrates strictly within $\{130, 140\}$ for Qwen-9B and within $\{160, 170\}$ for Gemma. *Conclusion:* This invariance across validation-set folds demonstrates that PAV does not overfit to idiosyncratic prompt instances; instead, it robustly captures the underlying capability plateau of the optimization trajectory regardless of sample-level variations.

Table 7: **Offline checkpoint-selector comparison.** Mean oracle regret measures the average performance deficit relative to the retrospective hindsight trajectory maximum across all four held-out transfer suites (lower is better; zero denotes oracle performance); "Imp." denotes the number of transfer suites improved over Base ($/4$). Checker correlation selects by score–gold Pearson correlation.

Trajectory	Selector	Step	Imp.	Mean regret
Qwen-9B	PAV	130	4/4	**1.20**
	SV-HARD accuracy	180	2/4	5.41
	Checker correlation	200	2/4	5.51
	Fixed final budget	200	2/4	5.51
Gemma	PAV	160	4/4	**0.49**
	SV-HARD accuracy	160	4/4	**0.49**
	Checker correlation	110	3/4	1.23
	Fixed final budget	200	4/4	0.95

Offline selector comparison and oracle regret. To rigorously quantify how effectively an online validation selector identifies high-performing checkpoints without peeking at held-out transfer benchmarks, Table 7 compares PAV against three standard selection heuristics: (1) monitoring SV-HARD rule accuracy alone ($\arg\max_t \widehat{A}_t$), (2) monitoring checker score-to-gold Pearson correlation, and (3) a fixed full-budget endpoint rule (step 200).

We formalize selector quality through *mean oracle regret*:

- **Hindsight Oracle (O_m)**: An omniscient oracle evaluates every intermediate checkpoint post-hoc across all four held-out transfer benchmarks $m \in \mathcal{M} = \{\text{HealthBench}, \text{RubricBench}, \text{CheckEval}, \text{ProfBench}\}$, recording the retrospective global maximum achievable score on each suite: $O_m = \max_t \text{Score}_t(m)$.
- **Regret Deficit**: For any realistic selector that chooses an operational stopping checkpoint $\hat{t}$ strictly using online validation signals, its performance deficit on benchmark m is $O_m - \text{Score}_{\hat{t}}(m) \geq 0$.
- **Mean Oracle Regret**: Averaging this deficit across the four suites yields $\frac{1}{|\mathcal{M}|}\sum_{m\in\mathcal{M}}(O_m - \text{Score}_{\hat{t}}(m))$. HealthBench, RubricBench, and ProfBench enter as percentage-point deficits; CheckEval enters as a difference in Pearson ρ. The mean is therefore on that mixed scale (lower is better; zero denotes parity with the hindsight oracle on every suite).

As shown in Table 7, PAV achieves the lowest mean oracle regret on both architectures (1.20 on Qwen-9B and 0.49 on Gemma-4B) and retains gains over Base on all four transfer suites ($4/4$). Tracking validation rule accuracy alone selects a late Qwen-9B checkpoint (step 180) with mean regret 5.41, and that checkpoint improves only two of the four suites. The final-budget checkpoint (step 200) has mean regret 5.51 on the same trajectory. *Conclusion:* By penalizing checker score-fidelity degradation alongside judge accuracy, PAV reliably approximates the hindsight oracle upper bound without leaking supervision from held-out transfer suites.

Statistical significance of out-of-distribution transfer gains. On the primary Qwen-9B ablation backbone, the gains over Base across all transfer suites are statistically significant:

- **CheckEval-Summ** ($1{,}600$ summaries; 15 checklist criteria in four quality dimensions): The macro-averaged correlation rises from 0.422 to 0.441 (paired $t = 2.95$, $p = 0.0032$).
- **ProfBench** (120 expert queries, $\approx 3{,}500$ rule evaluations): Paired cluster bootstrap test across rule sets yields $p = 0.008 < 0.01$ for the $+1.6$pp rule accuracy lift ($73.1\% \rightarrow 74.7\%$).
- **HealthBench** ($1{,}459$ physician-annotated queries) and **RubricBench** ($2{,}294$ pairwise comparisons): Both demonstrate positive gains ($+3.8$pp and $+3.1$pp) with $p < 0.0001$.
- **Downstream DPO (Qwen-27B)**: On GPQA, whereas base-judge labels degrade performance ($80.87 \rightarrow 79.02$), RECURSE labels achieve 81.46, establishing a statistically reliable net positive margin of $+2.44$pp over the base judge.

Table 8: **Systems and operational trade-off matrix.** Comparing different paradigms for improving LLM judge capabilities: commercial API judges, standalone learned Reward Models (RMs), and RECURSE (Ours).

Dimension	Commercial API Judge (e.g., GPT-4o)	Standalone Learned RM	RECURSE (Ours)
Training Reward Supervision	Outsourced to external commercial model	Large-scale curated human preference pairs	**Zero external gold** (on-policy process check)
Operational & Financial Cost	Recurring per-token API query fees	Separate data collection & RM training compute	**Amortized local inference** on shared hardware
Throughput & Latency	Bound by API rate limits & network WAN	High (co-located local inference)	**High** (co-located vLLM worker snapshot)
Reward Hacking Resistance	Susceptible to prompt-level exploitation	High risk (policy exploits static proxy flaws)	**Mitigated** (interface decoupling + PAV stopping)
Data Governance & Privacy	External transmission required	Fully on-premises	**Fully on-premises**

These statistical tests confirm that held-out suite improvements reflect genuine capability transfer rather than stochastic evaluation noise.

O SYSTEMS, SCALABILITY, AND COST-PERFORMANCE TRADE-OFF ANALYSIS

Checker compute overhead and decoding throughput. While the checker audits all $n = 8$ judge rollouts per training prompt (yielding matching 1:1 rollout generation counts), it operates strictly in *pure inference mode* without gradient backpropagation or optimizer state updates. In standard RL fine-tuning, policy optimization is dominated by the actor's backward pass, gradient computation, and optimizer step, which typically consume $2\times$ to $3\times$ the FLOPs of a forward pass per token. By hosting the reward-only checker on a dedicated, co-located vLLM instance with continuous batching and PagedAttention, Pass 2 auditing executes with high throughput. Parameters are synchronized via in-memory peer-to-peer tensor broadcasts every step (≈ 1.2 seconds on Qwen-27B), eliminating the substantial training and maintenance overhead of separate learned reward models.

Synchronization latency and scalability on multi-GPU architectures. In distributed training across multi-GPU nodes, parameters are synchronized via peer-to-peer tensor transfers over high-speed interconnects (NVLink within nodes, InfiniBand across nodes). On our largest trained model, Qwen-27B (Tensor Parallelism size 4), in-memory parameter broadcast requires only ≈ 1.2 seconds per optimization step. Because autoregressive rollout generation across $n = 8$ samples dominates the per-step iteration time ($\approx$ 45–60 seconds), checker parameter synchronization constitutes $< 2.5\%$ of total iteration wall-clock time. This confirms that synchronous weight tying introduces negligible communication bottleneck on multi-GPU distributed systems.

Trade-off comparison across judge-improvement paradigms. Table 8 summarizes the systems and operational trade-offs across different paradigms for improving LLM judge capabilities during post-training optimization: (1) distilling from or querying proprietary commercial LLMs (e.g., GPT-4o) as external judges/teachers during rollouts, (2) training separate standalone learned Reward Models (RMs) on curated human preferences, and (3) our closed-loop self-evaluation framework (RECURSE).

- **Commercial API Judges**: While commercial APIs provide strong evaluative signals for judge improvement, querying them synchronously across thousands of intermediate RL rollouts introduces significant network latency, strict API rate-limit bottlenecks, ongoing query fees, and potential exposure of sensitive training prompts.
- **Standalone Learned RMs**: Training an auxiliary reward model to supervise judge training requires extensive preference data curation. Once fixed, static reward models fail to adapt to the evolving judge policy, suffering from out-of-distribution proxy degradation (Goodhart's law). This is empirically validated by our RQ4 ablation (Table 3): even a $3\times$ larger static 27B checker incurs substantially higher inference cost, yet its average accuracy is still no higher than our synchronized 9B checker.
- **RECURSE**: By establishing a self-contained, closed-loop judge–checker recurrence on shared model parameters, RECURSE eliminates external API fees and ongoing human annotation for training rewards, maintains high local decoding throughput, guarantees data privacy, and enforces reproducible, bounded optimization via PAV.

P FORMAT COMPLIANCE AND QUALITATIVE CHECKER FAILURE MODES

Malformed outputs are masked from both gradient and group statistics (§3.3), so a rising parse-failure rate would be an early collapse signal. Judge verdicts remain parseable throughout the 200-step window. The checker's score parse rate stays stable through the pre-boundary window and declines only after the PAV landmark, in line with the post-boundary degradation in Section 5.3.

Parse rate is not a substitute for PAV. Because that decline begins only after the landmark, parse rate cannot select the landmark and does not substitute for the fidelity term in PAV.

Qualitative failure modes of the late-stage checker (post-boundary dynamics). Qualitative error analysis of checker outputs beyond the PAV-localized window (steps 160–200) reveals three distinct degradation patterns explaining why unconstrained recursive training degrades out-of-distribution transfer:

1. **Score inflation and loss of ranking variance**: As policy optimization saturates on-policy training prompts, the late checker becomes overly lenient, assigning uniform score $s = 3$ or $s = 4$ to mediocre reasoning rollouts. This collapses within-group advantage variance ($\sigma_t \to 0$), starving the policy of informative relative gradients.
2. **Hallucinated meta-justifications**: When evaluating flawed judge rollouts, late-stage checkers occasionally fabricate non-existent quotes or misinterpret rubric criteria to rationalize a high meta-score, exhibiting confirmation bias.
3. **Verbosity drift and generation truncation**: Late-stage checkers exhibit an elongation of internal reasoning chains, occasionally exceeding the maximum decoding window (32,768 tokens) before emitting the $\langle|\text{Final Score}|\rangle$ tag, which accounts for late drops in parsed checker outputs.

These qualitative failure modes directly validate why tracking checker fidelity via PAV is vital for bounding the recursive optimization loop.

Q TRAINING AND DPO HYPERPARAMETERS

Main judge–checker RL. Qwen-9B and Gemma use train batch size 32, n=8 rollouts per prompt, PPO mini-batch size 16, actor learning rate 3×10^{-6}, clip ratio 0.005 (low=high), and no KL penalty. Qwen-27B uses batch size 48, the same learning rate, and KL coefficient 0.005. All runs use FSDP with optimizer offload and sequence parallelism size 4. Maximum prompt / response lengths are 8192 / 32768 tokens for the judge; the checker score pool uses a matching 32768-token generation budget. Checkpoints and the full held-out monitoring suite (including PAV) are written every ten steps over 200 steps for Qwen-9B and Gemma, and every five steps over 75 steps for Qwen-27B. The checker is a standalone vLLM weight snapshot synchronized from the actor every step.

Downstream DPO. Preference pairs are labeled by our retained Qwen3.5-9B RECURSE judge (retained at the PAV-selected checkpoint, step 130) or its untuned base counterpart. Fine-tuning uses LlamaFactory (Zheng et al., 2024) on Qwen3.6-27B with LoRA rank 16, α=32, dropout 0.05, applying LoRA adapters to all linear projection layers, preference β=0.1, sigmoid DPO loss, learning rate 5×10^{-6}, cosine schedule with 3% warmup, three epochs, per-device batch size 1 with gradient accumulation 8 (global batch 64 on 8 GPUs), cutoff length 8192, and bf16. Base-judge and RECURSE-judge runs differ only in the preference file; optimization settings are matched.

Teacher-trajectory SFT. The matched-step SFT reference fully fine-tunes Qwen3.5-9B (no LoRA) on 6400 trajectories distilled from four high-quality teacher judges (Qwen3.5, Gemini-3.1-Flash-Lite, DeepSeek-V4-Flash, Kimi-K2.6) on the training prompt pool, with learning rate 1×10^{-5}, global batch 64, and two epochs (200 optimizer steps), matching the main run's prompt×step budget. A larger-pool variant uses the same teachers and hyperparameters on the full eligible distillation set (≈17,009 trajectories; two epochs, 532 optimizer steps at the same global batch). While scaling distillation data from 6.4k to 17k improves SFT accuracy on SV-HARD ($64.2\% \to 70.2\%$), both SFT arms remain consistently below our on-policy RECURSE

Table 9: **Taxonomy of self-improving evaluation frameworks.** Delineating RECURSE from prior self-training evaluation literature.

Method	RL Reward Source	External Anchors in RL	Interface Decoupling	Stopping Monitor	Feedback Target
Self-Taught Evaluator (Wang et al., 2024)	SFT on synthetic pairs	Synthetic seed pairs	N/A (SFT only)	Fixed SFT epochs	Preference label
Meta-Rewarding (Wu et al., 2025)	Meta-judge scalar reward	External frozen judge	No (coupled text)	Heuristic / steps	Outcome judgment
SELF-JUDGE (Lee et al., 2024)	Execution	Code interpreter / unit tests	N/A	Test pass rate	Program execution
Grad2Reward (Zhang et al., 2026)	Meta-gradient alignment	Reference RM	No	Reference validation	Reward weights
RECURSE (Ours)	**Synchronized self-checker**	**None (zero gold in RL)**	**YES (scalar vs. verdicts)**	**PAV on SV-HARD**	**Reasoning process**

judge (73.7% on SV-HARD, 86.3% on HealthBench, 79.1% on RubricBench, 0.441 on CheckEval), and scaling the static pool does not achieve suite-wide parity. As conjectured in Section 5 and Appendix C, one possible reason is that static offline distillation triplets $(h, y, r_{1:K})$ lose training tension because y is frozen while the student's decision boundary moves.

R TAXONOMY, CONCEPTUAL DISTINCTIONS, AND METHOD EXTENSIONS

To clearly delineate RECURSE from concurrent self-improving evaluators, Table 9 contrasts the core architectural and methodological dimensions.

- **Absence of external RL anchors**: Unlike Meta-Rewarding and Grad2Reward which require frozen reference models or external meta-judges, RECURSE generates the scalar training reward entirely from the model's own synchronized policy copy.
- **Structural shortcut mitigation**: Prior methods overlook surface coupling. RECURSE identifies the degenerative token copying channel and introduces interface decoupling to make closed-loop recursive optimization viable.
- **Theoretically grounded early stopping**: Instead of heuristic step budgets, RECURSE introduces PAV, grounded in pairwise score-difference error bounds, to reliably identify the validity window before validation deception sets in.

Surface coupling vs. classical reward hacking. It is crucial to distinguish *surface coupling* from classical reward hacking:

- **Classical Reward Hacking**: Arises when an actor optimizes against a *static, frozen* reward model, exploiting proxy misalignment (e.g., generating verbose text) while the RM itself remains fixed.
- **Surface Coupling**: Arises uniquely in self-improving recurrence due to weight synchronization ($\bar{\theta}_{t+1} \leftarrow \theta_{t+1}$). Because actor and checker share the identical YES/NO readout surface, policy updates inflating YES tokens simultaneously modify the reward function in real time, creating an unconstrained shortcut that bypasses reasoning auditing. Interface decoupling structurally severs this co-adapted token channel.

Generalization to holistic (non-rubric) evaluation. The experiments in this paper use rubric judging only. Applying interface decoupling and PAV to holistic scoring is untested; one possible sketch is:

1. **Pass 1 (Holistic Judge)**: Generates open-ended reasoning evaluating overall response quality, outputting an unconstrained quality score $q \in [1, 10]$.
2. **Pass 2 (Decoupled Meta-Checker)**: Audits whether reasoning is complete, grounded, and logically sound against universal meta-criteria (e.g., factual consistency, depth, tone), emitting a decoupled 5-tier scalar audit score $s \in \{0, \ldots, 4\}$ as the sole RL reward.
3. **Holistic PAV Monitoring**: A compact human-verified validation set of holistic preferences computes judge accuracy and checker fidelity to establish the early-stopping horizon.

S BROADER IMPACT AND ETHICAL CONSIDERATIONS

Automating evaluation with human oversight (*Humans-over-the-loop*). As LLMs are increasingly deployed across open-ended domains, manual verification becomes infeasible. RE-

CURSE supports scalable automated verification by allowing evaluators to self-improve under fixed rubrics. Crucially, humans remain *over* the loop by authoring domain standards and establishing compact human-verified monitors (SV-HARD), while models perform high-throughput evaluation and self-auditing.

Mitigating automation bias and unanchored drift. A primary concern in self-improving systems is unconstrained drift or self-reinforcing bias (automation bias). RECURSE directly addresses this risk through: (1) **Structural process auditing**: Auditing reasoning against explicit meta-rubrics is intended to penalize ungrounded rationales, even if final verdicts appear correct; (2) **PAV safety shutdown**: Continuously monitoring checker fidelity against human gold detects reward degeneration and enforces an empirical stopping boundary before degradation occurs. In safety-critical domains (e.g., medical or legal), we recommend combining RECURSE judges with periodic human spot-auditing.

T LIMITATIONS

Scope of model architectures and task domains. Our empirical scope spans three model configurations up to 27B parameters (Qwen3.5-9B, Gemma-4-E4B-it, and Qwen3.6-27B) on English rubric benchmarks. We deliberately allocate compute such that Qwen3.5-9B serves as the primary scientific testbed for comprehensive ablations, Gemma-4-E4B-it provides cross-architecture replication, and Qwen3.6-27B demonstrates scale feasibility. While RECURSE achieves consistent gains across held-out medical, pairwise preference, summarization, and professional suites, extending RSI to non-rubric formats (e.g., holistic unconstrained scoring), multilingual judging, and multimodal domains remains valuable future work.

Role of human verification in the holdout monitor. Although gold labels never enter the RL training reward, RECURSE relies on a compact human-verified validation set (SV-HARD) to compute the PAV early-stopping index. Constructing this 100-prompt holdout represents a real upfront human verification cost. While this cost is orders of magnitude smaller than labeling the entire training corpus, RECURSE keeps gold labels out of the RL reward rather than eliminating human verification from the end-to-end evaluation pipeline.

Absence of a gold-reward RL reference. Our controls cover static checkers, self-consistency, and teacher SFT, but not the same RL pipeline driven by gold rule labels or another externally verified reward. Such a reference would require per-rule annotation of the entire training corpus—frequently ten or more rules per instance—an annotation cost that is prohibitive for us and that RECURSE is designed to avoid. The gap between self-produced and gold rewards therefore remains unquantified in this work.

Statistical monitoring and sequential testing considerations. As analyzed in Section 3.3 and Appendix N, the empirical PAV estimator $\widehat{V}_t$ is unbiased at any fixed checkpoint. However, online sequential monitoring across multiple checkpoint iterations introduces multiple-testing considerations. In our experiments, PAV reliably localizes a stable validity region rather than a brittle single-step point estimate, and the stopping boundary is independently corroborated by SV-FULL. Developing formal sequential stopping boundaries with family-wise error rate control provides a promising avenue for further theoretical refinement.

Process checking assumptions and granularity. The process checker audits the internal reasoning trajectory of the judge against meta-rubrics rather than inspecting ground-truth outcomes directly. If a checker develops systematic reasoning biases, it could theoretically assign high scores to flawed rationales. The 5-tier scalar interface (0–4) mitigates surface token shortcuts and reduces regression noise, though exploring adaptive scalar rewards with dynamic format gating warrants further exploration.

Scope of RSI. Finally, bounded RSI in this work intentionally fixes the task specification, optimizer, and checking protocol. The policy's evaluative capability provides the reward for its own parameters, but the model does not autonomously modify the training architecture or meta-rubric

definitions. Our conclusions therefore apply specifically to closed-loop judge–checker optimization within structured evaluation domains rather than to open-ended, unconstrained recursive self-modification.