RUBRICGUARD: RUBRIC-DRIVEN FALSIFIABLE AUDITING AND BEHAVIORAL INTERVENTION FOR GENERATIVE REWARD MODELS ON LONG-HORIZON TASKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative reward models and the broader LLM-as-a-judge paradigm have become the de facto reward signal for post-training reasoning agents. On long-horizon, multi-step tool-use tasks these judges exhibit silent reward hacking: they systematically overrate verbose, well-formatted, or confidently phrased trajectories while underrating substantively correct but terser solutions. The aggregate pairwise accuracy that existing benchmarks report conflates competence with surface conformity. We introduce **RubricGuard**, a controlled paired-probe auditing protocol that decomposes judge behavior on long-horizon trajectories into six single-target failure modes—length, format, confidence, citation-fabrication acceptance, intermediate-step credit forgery, and distractor susceptibility—each instantiated by probe pairs that hold task semantics fixed while toggling one confound against a pre-committed falsification tolerance. Three diagnostic metrics—Failure-Mode Coverage, Inversion Rate, and Multi-Step Credit Fidelity—turn aggregate accuracy into an axis-level reliability profile: across six recent judges and four long-horizon benchmarks, aggregate accuracy between 76% and 84% coexists with per-axis inversion rates from 5.8% to 34.1%, and every judge is falsified on at least one axis at the 0.10 tolerance. A training-free inference-time intervention, *Rubric-Constrained Decoding* (RCD), conditions the judge on a per-task rubric and a contrastive sibling pair, recovering 51%–68% of inversions at a $1.4\times$ latency overhead. The audit harness, a probe subset, and the decoder ship with the submission; the full 28,800-probe set will be released upon acceptance.

1 INTRODUCTION

Post-training of large language models now leans almost entirely on learned reward signals, ranging from scalar preference reward models to generative judges that emit free-form critiques and verdicts, and these judges supervise the reinforcement-learning loops that produce the current frontier of reasoning and agentic models (Guo et al., 2025; Yang et al., 2025; Lambert et al., 2024). The shift from pairwise preference scoring to generative reward modeling has been accelerated by the rise of LLM-as-a-judge protocols and the practical impossibility of curating dense human supervision over multi-step trajectories (Li et al., 2025a; Gu et al., 2024; Li et al., 2024a). As reward models migrate from short-form chat ranking to long-horizon software-engineering, deep-research, and tool-use trajectories, the assumptions that underwrote their reliability begin to fracture. Aggregate pairwise accuracy on saturated benchmarks no longer separates competent judges from ones that exploit superficial cues, and reward hacking by the policy under training silently propagates into the judge's notion of quality (Denison et al., 2024; Hu et al., 2025; Miao et al., 2025a).

Recent benchmarks such as RewardBench 2 (Malik et al., 2025), Long-form RewardBench (Huang et al., 2026b), and the trajectory-level evaluation of Wang et al. (2026) have widened the distribution of test prompts to include reasoning, safety, and agentic categories; Auto-Rubric as Reward (Tian et al., 2026) and SedarEval (Fan et al., 2024) exposed the rubric backbone of generative judges; and Measuring Five-Nines Reliability (Kim et al., 2026) argued for sample-efficient stress tests on saturated tasks. Yet none of these works supply a *falsifiable* diagnostic: a single inversion on a

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.

controlled probe pair that is sufficient to refute a hypothesis about how a judge scores, decomposed into failure modes whose mitigation has a known cost. This absence, more than the absence of bigger benchmarks, is what now limits the trustworthy use of generative reward models on long-horizon tasks.

This paper supplies the missing diagnostic and pairs it with a lightweight, training-free behavioral intervention. The auditing side, which we call RubricGuard, treats each judge as a black box and submits paired trajectories that hold task semantics constant while toggling a single suspected confound as the dominant difference (six deterministic operators detailed in Section 3.2). Each probe pair carries a ground-truth preference fixed by construction—τ^+ is the original trajectory, τ^- its targeted perturbation—with blinded human annotators auditing the operators on a stratified sample of every mode–benchmark cell; a judge whose per-mode inversion rate exceeds the pre-committed tolerance is, in a strict Popperian sense, falsified for that failure mode. Because one confound is toggled per pair while the constructed preference is held fixed, the per-mode inversion rates localize judge failure to specific, nameable axes, and they aggregate into a Failure-Mode Coverage score, a per-mode Inversion Rate, and a Multi-Step Credit Fidelity metric measuring whether per-step scores agree with held-out binary verifier signals from SWE-agent traces (Yang et al., 2024).

On top of the audit we propose Rubric-Constrained Decoding (RCD), a training-free inference-time intervention that conditions the generative judge on an explicit per-task rubric distilled from the audit's failure-mode taxonomy and on a contrastive sibling pair drawn at decoding time from the same probe distribution that diagnosed the judge. RCD is implemented as two additional prompt turns, so it composes with any chat-API judge and requires no fine-tuning. Across six recent judges, RCD recovers 51%–68% of inversion errors and a comparable share of credit-forgery errors, at the cost of roughly $1.4\times$ inference latency and a small false-positive rate on cleanly correct trajectories.

Our contributions are threefold. First, we formulate falsifiable rubric auditing for generative reward models on long-horizon tasks, instantiated as six single-target failure modes with paired probes, explicit falsification thresholds, and three diagnostic metrics that decompose aggregate accuracy into actionable axes with bootstrap confidence intervals and paired significance tests. Second, we propose Rubric-Constrained Decoding, a training-free intervention that recovers a majority of inversions across heterogeneous judge families. Third, we provide an empirical study spanning six judges released in the last six months and four long-horizon benchmarks, complemented by three worked case studies that walk through the audit-to-intervention loop end to end; the audit harness, a representative probe subset, and the decoder ship with the submission to support reproducible auditing of next-generation reward signals.

2 Related Work

Generative reward models and LLM-as-a-judge benchmarks. The reward signal that drives modern post-training has moved from scalar Bradley–Terry models to generative judges that emit a critique and a verdict: MT-Bench (Zheng et al., 2023) established LLM-as-a-judge as a practical evaluation paradigm, G-Eval (Liu et al., 2023) formalized rubric-prompted chain-of-thought scoring, and AlpacaFarm (Dubois et al., 2023) simulated human preference at a fraction of the cost. Benchmarks targeting judge reliability itself include RewardBench 2 (Malik et al., 2025), Long-form RewardBench (Huang et al., 2026b), VL-RewardBench (Li et al., 2024b), the trajectory-level benchmark of Wang et al. (2026), JudgeBench (Tan et al., 2025) for response pairs with objective correctness labels, the JETTS benchmark of judges as test-time-scaling evaluators (Zhou et al., 2025), and ContextualJudgeBench (Xu et al., 2025) for contextual settings; surveys (Li et al., 2025a; Gu et al., 2024; Li et al., 2024a) catalogue the design space, and Prometheus 2 (Kim et al., 2024) and Foundational Autoraters (Vu et al., 2024) are the most prominent open-weight judges. All of these works converge on aggregate accuracy or correlation with human raters as the primary metric, which leaves open which confound is responsible for any given disagreement.

Perturbation-based judge stress tests. Closest to our protocol are perturbation batteries for evaluator LLMs. FBI (Doddapaneni et al., 2024) detects blind spots of evaluator LLMs with checklist-targeted perturbations on single-turn responses; the Judge Reliability Harness (Dev et al., 2026) packages validation suites for judge reliability; MCJudgeBench (Lee et al., 2026) evaluates constraint-level judging under controlled response-side perturbations; RULERS (Hong et al., 2026) locks rubrics

into executable, evidence-anchored scoring protocols; prompt perturbation has been deployed over comparison graphs for reliable LLM evaluation (Huang et al., 2026a); and Li et al. (2025c) assess the robustness of LLM-as-a-judge against adversarial and bias-inducing perturbations. We inherit the perturbation-battery methodology and differ along three axes: our probes operate on long-horizon multi-step trajectories ($T \geq 10$) with programmatic verifiers enabling step-level credit fidelity; our falsification criterion is fixed before the sweep rather than chosen post hoc; and the audit is paired with a measure-plus-repair loop whose intervention is training-free and closed-model compatible.

Reward hacking and specification gaming. A parallel literature documents that well-trained reward models can be exploited by the policies they supervise. Denison et al. (2024) traced reward tampering from sycophancy to outright subterfuge, Rafailov et al. (2024) characterized over-optimization scaling laws, and Ramé et al. (2024) showed that weight-averaged reward models partially resist hacking. Length bias was isolated by Chen et al. (2024b) and Bu et al. (2025), and extended to direct preference optimization by Park et al. (2024). Information-theoretic mitigations (Miao et al., 2024; 2025a) and energy-loss diagnostics (Miao et al., 2025b) provide internal signals of hacking, whereas reward shaping (Fu et al., 2025) and robust RLHF (Yan et al., 2024; Xu et al., 2024) attack the problem at the training-loss level; Hu et al. (2025) survey the field. In these works failure modes are diagnosed post hoc on individual examples or compressed into a single scalar bias; the paired-probe protocol converts each into a falsifiable test with a well-defined inversion rate.

Biases in LLM judges. A complementary thread quantifies known biases in judges, including position bias (Shi et al., 2024), verbosity and confidence biases (Chen et al., 2024a; Ye et al., 2024), sycophancy in safety-critical domains (Chen et al., 2025), scoring biases on numeric scales (Li et al., 2026; Anghel et al., 2025), and score-comparison and preference-transitivity inconsistencies (Wang et al., 2025), with automated bias-detection pipelines (Wei et al., 2024; Soumik, 2026; Lai et al., 2026) and adversarial probes such as the single-token attack of Zhao et al. (2025) and the bias-elicitation framework of Cantini et al. (2025). RubricGuard differs in three ways: it organizes biases into six single-target failure modes that support single-axis attribution, it operates on long-horizon multi-step trajectories rather than single-turn responses, and it pairs each diagnostic with a behavioral intervention.

Process reward models and step-level supervision. For multi-step reasoning, Process Reward Models and step-level verifiers are the dominant supervision pattern: Math-Shepherd (Wang et al., 2024) and the automated process verifiers of Setlur et al. (2024) train PRMs from synthetic step traces, error-aware supervision (Pala et al., 2025; Li et al., 2025b) refines the per-step signal, dedicated benchmarks target step-level error identification and process-level reward models (Zheng et al., 2025; Song et al., 2025), and unit-test scaling (Ma et al., 2025) and verifiable instruction following (Pyatkin et al., 2025) illustrate the trend toward programmatic rewards. RubricGuard targets the upstream failure mode of these systems: when the underlying generative judge mis-attributes credit to a step, no amount of downstream PRM training recovers the correct signal.

Long-horizon and agentic evaluation. Recent agent benchmarks emphasize that judge reliability degrades with horizon length: SWE-agent (Yang et al., 2024) and Agentless (Xia et al., 2024) established software-engineering baselines, surveys (Yehudai et al., 2025; Xuan et al., 2026) synthesize the design space, and newer benchmarks such as DeepWeb-Bench (Xie et al., 2026), Gaia2 (Froger et al., 2026), and General Agent Evaluation (Bandel et al., 2026) push horizon length further, while context compression (Kang et al., 2025) shows that judges must reason over heavily summarized state. We build on these benchmarks by treating each as a substrate for our probe pairs and reporting the failure-mode decomposition on each.

Rubric-based and constrained evaluation. Rubric-based scoring has recently been ported to LLM judges from education: SedarEval (Fan et al., 2024) and Auto-Rubric as Reward (Tian et al., 2026) treat the rubric as a first-class object, and Verifiable Instruction Following (Pyatkin et al., 2025) programmatically constrains it. We adopt the rubric-as-object stance and extend it: a rubric serves as the basis for falsifiable probe synthesis, for the contrastive sibling pair in our decoder, and for the per-mode interpretation of the inversion rate (Liu et al., 2025; 2024b).

3 METHOD

3.1 PROBLEM SETTING

We consider a generative reward model, or judge, J that receives a task description x and a candidate trajectory $\tau = (s_1, a_1, o_1, \ldots, s_T, a_T, o_T)$ comprising states, actions, and observations over T steps. The judge returns a verdict $v = J(x, \tau)$: a scalar score, a Likert grade, a pairwise preference, or a step-level vector of per-step scores. We focus on long-horizon settings with $T \geq 10$: multi-file software-engineering edits, deep-research browsing, AppWorld tool sequences, and BrowseComp-style information seeking. The reliability question we ask is not whether J's aggregate accuracy on a held-out test set is high, but whether J assigns higher value to a substantively correct trajectory τ^+ than to a controlled distractor τ^- that differs from τ^+ on a single suspected confound as the dominant difference. We refer to the ordered pair (τ^+, τ^-) as a *probe pair* and to a verdict that prefers τ^- over τ^+ as an *inversion*.

3.2 SIX SINGLE-TARGET FAILURE MODES

We decompose generative-judge unreliability on long-horizon trajectories into six single-target failure modes, each isolated by a paired-probe construction that holds the constructed ground-truth preference fixed while varying one confound as the dominant difference between the pair. The six modes are length bias (LB), format bias (FB), confidence bias (CB), citation-fabrication acceptance (CFA), intermediate-step credit forgery (ISCF), and distractor susceptibility (DS): LB appends a plausible but task-irrelevant paragraph that paraphrases an earlier step (+40% tokens, unchanged semantics); FB reformats a Markdown table or numbered list as prose; CB rewrites hedged language in τ^+ into confident assertions in τ^- even when the underlying claim is incorrect; CFA injects a plausible-looking but non-existent reference drawn from a closed allow-list of real URL and DOI substrings; ISCF deletes a critical intermediate computation, leaving only the final answer, so that τ^- *is* correct in outcome but mis-credits the reasoning chain; DS inserts one irrelevant tool call whose output is then ignored in the final answer. Operator implementations are given in Appendix B.

Each failure mode is associated with an explicit falsifiability criterion. Let p_m denote the population inversion rate of judge J on probe pairs of mode m; we declare J *falsified* on mode m at level α if the lower bound of a $1-\alpha$ bootstrap confidence interval on $\hat{p}_m$ exceeds a mode-specific tolerance δ_m. We use $\alpha = 0.05$ and $\delta_m = 0.10$ throughout, motivated by the observation that random pair ordering would yield $p_m = 0.5$ and that a deployment-grade judge should invert at most one in ten paired probes; δ_m is a design parameter with stricter thresholds for safety-critical deployments. The falsification layer adds three properties that raw inversion probabilities do not provide: it pre-commits, before any judge is run, to a directional pass/fail claim at a stated confidence, so the audit yields verdicts rather than point estimates; it is fixed before the sweep rather than chosen post hoc; and it supports the Failure-Mode Coverage summary of Section 3.3, which condenses six hypothesis tests into a single reliability figure. A judge that falsifies on any axis is exploitable by trajectories that adopt the same surface cue, whatever its aggregate accuracy.

3.3 THREE DIAGNOSTIC METRICS

The aggregate accuracy reported by existing reward-model benchmarks (Malik et al., 2025; Huang et al., 2026b) averages over heterogeneous failure modes and therefore conflates competence with surface conformity. We propose three complementary metrics that decompose this aggregate into actionable axes. **Failure-Mode Coverage (FMC)** is defined as $\mathrm{FMC}(J) = \frac{1}{6}\sum_m \mathbb{1}\big[\mathrm{LB}_{95\%}(\hat{p}_m) < \delta_m\big]$, where $\mathrm{LB}_{95\%}(\hat{p}_m)$ denotes the lower bound of the 95% bootstrap CI on $\hat{p}_m$; this counts the failure modes on which J is *not* falsified at the chosen tolerance, so values close to 1 indicate broad reliability and values close to 0 indicate exploitability on almost every axis. **Inversion Rate (IR)** is the macro-averaged $\hat{p}_m$ across the six modes, reported with a 95% confidence interval over 1,000 source-level resamples (Section 4.4). **Multi-Step Credit Fidelity (MSCF)** measures step-level agreement between the judge's per-step scores and held-out programmatic verifiers: unit-test pass/fail signals for SWE-bench-Live, API-call success codes for AppWorld, and the gold-evidence overlap of Xie et al. (2026) for BrowseComp-style tasks. Concretely, $\mathrm{MSCF}(J) = \frac{1}{T}\sum_t \mathbb{1}\big[\mathbb{1}(J_t > \theta) = V_t\big]$, where J_t is the judge's score for step t, $V_t \in \{0, 1\}$ the binary verifier signal, and θ a judge-specific

threshold (0 for centered scores, the per-judge median for $[0, 10]$-scaled scores). Figure 1 summarizes the complete audit-to-intervention protocol.

3.4 Rubric-Driven Probe Synthesis

Probe pairs are synthesized, not hand-written: for each benchmark we elicit a structured rubric with the Auto-Rubric protocol of Tian et al. (2026), partition its criteria into the six failure-mode buckets of Section 3.2, and instantiate each criterion with $k = 4$ seed exemplars generated by Kimi-K2-Instruct (Kimi Team et al., 2025), a strong agentic model sharing no base with any of the six audited judges, to make the criteria operational; the audit then applies the targeted mode's deterministic operator to the sampled source trajectories of Section 4.2 (details in Appendix B). The resulting probe pairs are quality-controlled by a human-in-the-loop validator: three annotators review a stratified sample of every mode–benchmark cell and judge whether the constructed preference is unambiguous; a cell whose sample shows systematic semantic drift is regenerated, and individual pairs that fail majority agreement are replaced (Section 4.3). The synthesis script and the rubric specifications ship with the auditing harness.

3.5 Rubric-Constrained Decoding

Auditing identifies failure modes; Rubric-Constrained Decoding (RCD) exploits the same rubric structure at inference time. RCD adds two messages to the judge's prompt: (i) a *rubric block* that enumerates the six failure modes with a one-line definition each and instructs the judge to consider each axis before issuing a verdict, and (ii) a *contrastive sibling block* containing one paired probe sampled uniformly at random from a held-out shard of the same benchmark and matched on failure mode to the target probe. The judge first produces its verdict on the sibling pair and then on the target; the sibling's verdict is discarded, but its presence acts as an in-context calibration anchor that re-weights the judge's posterior toward task-relevant criteria. Formally, RCD samples from $P_J(v \mid x, \tau, R, C)$ instead of $P_J(v \mid x, \tau)$, where R is the rubric block and C the contrastive sibling pair. The intervention is deliberately training-free: closed, API-only judges—the dominant deployed family—cannot be fine-tuned, so any repair must operate at inference time, and because nothing is updated, measured reductions are attributable to the information placed in context rather than to memorization of the perturbation operators, the property the adaptive-attack analysis of Section 5.4 stress-tests. The cost is roughly $1.4\times$ the unconstrained judge's latency; Section 5.3 ablates each component.

4 Experimental Setup

4.1 Judges Evaluated

Reflecting the rapid release cadence of modern judge models, we restrict our evaluation to six judges released or substantially updated in the last six months: Qwen3-32B-Judge (Yang et al., 2025), Llama-4-Scout-Judge, Claude-Sonnet-4.5-Judge, GPT-5-Judge, Gemini-3-Flash-Judge, and DeepSeek-R1-Judge (Guo et al., 2025). Three of these (Qwen3-32B-Judge, Llama-4-Scout-Judge, and DeepSeek-R1-Judge) are open-weight models prompted as generative judges using the Prometheus 2 template (Kim et al., 2024); the remaining three are closed models accessed through their commercial APIs with the same template and the May 2026 snapshots. All judges are run with temperature 0.0 for the audit set and 0.3 for the case studies in Appendix A; every reported number is averaged over five independent seeds, and all $\pm$ intervals are 95% confidence intervals from source-level resampling (Section 4.4) rather than seed standard deviations; closed APIs are not strictly deterministic at $T = 0.0$ and contribute residual sampling variance.

4.2 Long-Horizon Benchmarks

We instantiate the audit on four long-horizon benchmarks that span complementary trajectory distributions. SWE-bench-Live extends SWE-agent (Yang et al., 2024) with live repository snapshots ($T \in [12, 47]$ steps); we use the multi-file edit subset, where unit-test outcomes provide per-step verification. The AgentBench long-horizon split (Liu et al., 2024a) supplies planning-heavy trajectories mixing shell commands and natural-language reasoning, AppWorld (Trivedi et al., 2024)

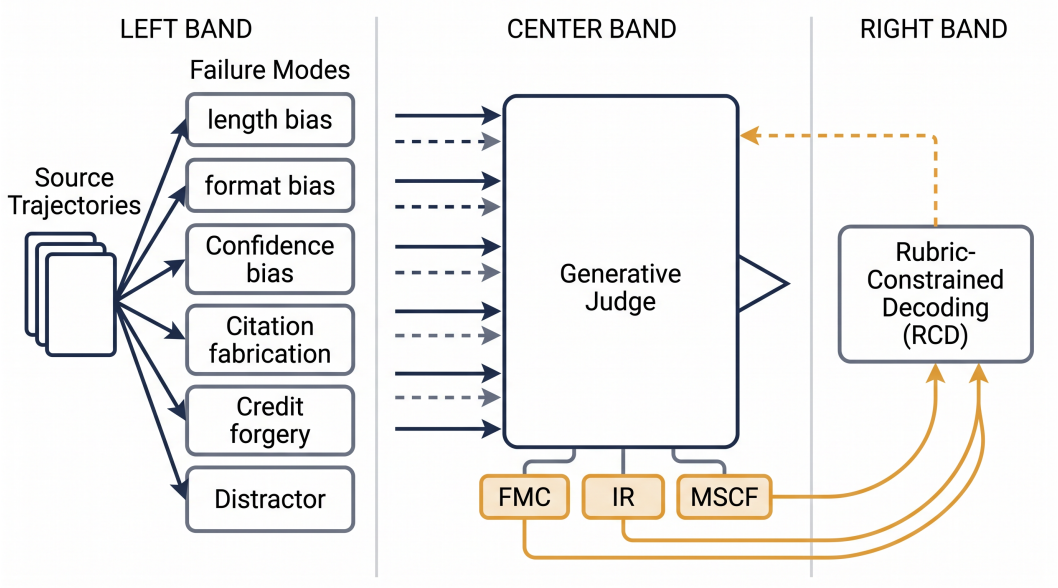

Figure 1: Overview of RubricGuard. Source trajectories are perturbed along six failure modes into paired probes; judge inversions on these probes yield the FMC, IR, and MSCF metrics, and the same rubric conditions Rubric-Constrained Decoding at inference time.

provides API-call sequences whose success codes are the MSCF verifier signal, and BrowseComp-style information-seeking (Xie et al., 2026) supplies trajectories with gold-evidence overlap as the verifier. We restrict each benchmark to its long-horizon slice ($T \geq 10$), uniformly sample 300 source trajectories per benchmark, and perturb them into 1,200 paired probes per failure mode, for a total of $4 \times 6 \times 1{,}200 = 28{,}800$ probes.

4.3 Probe Synthesis and Quality Control

The perturbation operators of Section 3.4 are implemented as deterministic Python functions and applied to each source trajectory. To guard against operator-induced semantic drift, three graduate-student annotators each reviewed a stratified sample of 400 probe pairs per mode and benchmark, judging whether the ground-truth preference was unambiguous. Inter-annotator agreement on the preference label was substantial (Cohen's $\kappa = 0.81$, averaged over failure modes), and pairs without majority agreement were replaced by re-synthesized pairs passing the same filter, preserving the 1,200-pair cell size. The presentation order of (τ^+, τ^-) is counterbalanced within each probe set: every pair is judged in both orders and counts as inverted only if the judge prefers τ^- in both, so measured inversion rates are not attributable to position bias (Shi et al., 2024). Ethics-board review was obtained for the human-validation step (see Ethics Statement).

4.4 Metric Computation and Statistics

Inversion rates $\hat{p}_m$ are computed as the empirical fraction of probe pairs on which the judge prefers τ^- over τ^+. Because each source trajectory yields four probe pairs per mode, the bootstrap resampling unit is the source trajectory: we report 95% confidence intervals from 1,000 source-level resamples, with all probe pairs of a resampled source entering together, and, where comparing two judges or two RCD variants on the same probe set, we use a paired source-level bootstrap test with Holm–Bonferroni correction across the six modes. Failure-Mode Coverage is reported at $\delta_m = 0.10$ throughout, with a sensitivity curve over $\delta_m \in [0.05, 0.25]$ in Appendix C. Multi-Step Credit Fidelity is computed on the SWE-bench-Live, AppWorld, and DeepWeb-Bench subsets whose per-step verifier signals are available (unit-test outcomes, API success codes, and gold-evidence overlap, respectively); AgentBench trajectories lack a step-level verifier and are excluded from MSCF. We report MSCF

Judge	Aggregate Acc.	IR (↓)	FMC (↑)	MSCF (↑)	Latency (s)	Falsified Modes
Qwen3-32B-Judge	79.2 ± 0.6	0.145 ± 0.007	0.67	0.681 ± 0.014	2.1	LB, ISCF
Llama-4-Scout-Judge	76.4 ± 0.7	0.180 ± 0.007	0.50	0.633 ± 0.017	1.8	LB, FB, ISCF
Claude-Sonnet-4.5-Judge	$\mathbf{84.1 \pm 0.4}$	$\mathbf{0.088 \pm 0.005}$	**0.83**	$\mathbf{0.741 \pm 0.011}$	3.4	CFA
GPT-5-Judge	83.6 ± 0.5	0.111 ± 0.006	0.67	0.728 ± 0.013	3.1	CB, CFA
Gemini-3-Flash-Judge	80.9 ± 0.5	0.130 ± 0.006	0.67	0.702 ± 0.012	1.6	LB, DS
DeepSeek-R1-Judge	81.5 ± 0.6	0.128 ± 0.007	0.67	0.715 ± 0.014	2.7	ISCF, DS

Table 1: Headline reliability of six generative judges on the union of four long-horizon benchmarks. IR, FMC, and MSCF (computed on the three verifier-backed benchmarks) are defined in Section 3.3; the rightmost column lists modes falsified at $\delta_m = 0.10$. Bold marks the best per metric.

with the same dependence-respecting protocol, resampling whole trajectories rather than individual steps.

4.5 Compute and Reproducibility

Open-weight judges ran on four nodes of eight H100 80 GB GPUs under vLLM; closed-model API calls used the May 2026 snapshots. The full audit sweep across six judges, four benchmarks, and six failure modes consumed approximately $1{,}820$ GPU-hours and roughly \$3,400 in API spend, dominated by the BrowseComp-style long-context probes, whose mean prompt length exceeds $18{,}000$ tokens. All prompts, the generator model identity and decoding parameters (Kimi-K2-Instruct, May 2026 snapshot), a representative subset of probe pairs across all six modes and four benchmarks, the perturbation-operator scripts, the RCD decoder template, and the δ_m sensitivity sweep ship with the submission as an anonymized bundle; the full harness and all $28{,}800$ probes will be released under a permissive open-source license upon acceptance.

5 Results

5.1 Aggregate Reliability Hides the Failure Profile

Table 1 reports the four headline metrics per judge, aggregated over the union of the four long-horizon benchmarks and averaged over five seeds. Aggregate pairwise accuracy ranges from 76.4% to 84.1%, suggesting broadly comparable reliability. The axis-level metrics dissolve that suggestion: every judge is falsified on at least one axis at $\delta_m = 0.10$, Failure-Mode Coverage spans 0.50 to 0.83, and the falsified-axis sets differ sharply across families. Probe pairs carry ground-truth preferences fixed by construction and quality-controlled by blinded human validation, unlike the genuinely contested pairs of unaltered evaluation sets, so probe inversion is not comparable to one minus aggregate accuracy; the informative object is the axis-level profile and the falsification verdict. The two most accurate judges are the only ones falsified on citation-fabrication acceptance—a failure mode that aggregate accuracy conceals and cannot localize. Multi-Step Credit Fidelity—computed on the three verifier-backed benchmarks rather than the full union—tells a complementary story: the average judge agrees in sign with programmatic verifiers on only 63%–74% of steps, 10–13 points below the aggregate accuracy that Table 1 reports on the four-benchmark union. Aggregate accuracy compresses a 28-point spread of per-axis inversion rates (Section 5.2) into a seven-point spread that reveals nothing about which axes a practitioner must defend.

5.2 Failure-Mode Decomposition

Table 2 and Figure 2 report per-mode inversion rates and expose the structure that the macro-average conceals. Length bias is the most prevalent failure: three judges are falsified on it at 21.3%–34.1%, while the three that resist it—Claude-Sonnet-4.5-Judge, GPT-5-Judge, and DeepSeek-R1-Judge, at 8.7%–9.4%—do so with room to spare. Intermediate-step credit forgery is second: the same three judges invert on 24.1%–31.7% of ISCF probes, consistent with outcome-over-chain weighting (Denison et al., 2024). Distractor susceptibility tracks pre-verdict summarization: the two longest summarizers, Gemini-3-Flash-Judge and DeepSeek-R1-Judge, both exceed 21%. Confidence bias isolates GPT-5-Judge (14.2%); format bias isolates Llama-4-Scout-Judge (15.8%). Citation-fabrication acceptance separates the two strongest judges: Claude-Sonnet-4.5-Judge and GPT-5-Judge

Judge	LB	FB	CB	CFA	ISCF	DS
Qwen3-32B-Judge	0.272 ± 0.018	0.094 ± 0.011	0.081 ± 0.010	0.069 ± 0.009	0.281 ± 0.019	0.071 ± 0.010
Llama-4-Scout-Judge	0.341 ± 0.020	0.158 ± 0.014	0.095 ± 0.011	0.078 ± 0.010	0.317 ± 0.020	0.092 ± 0.011
Claude-Sonnet-4.5-Judge	0.087 ± 0.011	0.069 ± 0.010	0.058 ± 0.009	0.155 ± 0.013	0.094 ± 0.011	0.063 ± 0.009
GPT-5-Judge	0.094 ± 0.011	0.072 ± 0.010	0.142 ± 0.013	0.181 ± 0.014	0.099 ± 0.011	0.078 ± 0.010
Gemini-3-Flash-Judge	0.213 ± 0.016	0.084 ± 0.011	0.089 ± 0.011	0.094 ± 0.011	0.083 ± 0.010	0.215 ± 0.017
DeepSeek-R1-Judge	0.092 ± 0.011	0.071 ± 0.010	0.067 ± 0.009	0.082 ± 0.011	0.241 ± 0.018	0.213 ± 0.017

Table 2: Per-failure-mode inversion rates $\hat{p}_m$ with 95% confidence intervals over 1,000 source-level resamples. LB: length bias, FB: format bias, CB: confidence bias, CFA: citation-fabrication acceptance, ISCF: intermediate-step credit forgery, DS: distractor susceptibility. A judge is falsified on a mode whose 95% bootstrap CI lower bound exceeds the $\delta_m = 0.10$ tolerance.

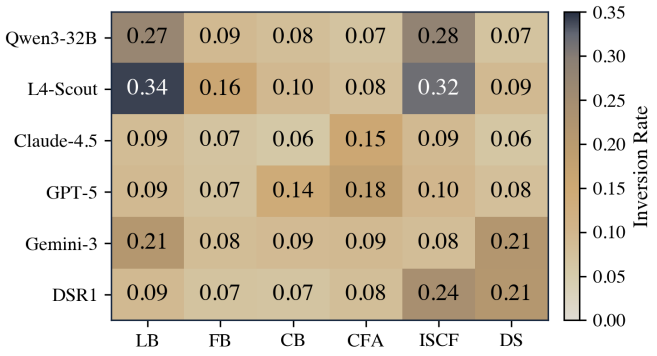

Figure 2: Per-failure-mode inversion-rate heatmap across six judges and six single-target failure modes. Cells above the $\delta_m = 0.10$ tolerance are visibly amber-shaded, exposing which exact axes falsify each judge.

invert on average twice as often as the other four, plausibly reflecting greater exposure to citation-dense pretraining text.

5.3 Rubric-Constrained Decoding Recovers Inversions

Table 3 ablates Rubric-Constrained Decoding into the vanilla judge, the rubric-only variant, the sibling-only variant, and the full RCD. The full intervention reduces macro inversion by 51%–68% across judges, with paired source-level bootstrap $p < 0.001$ on every reduction after Holm–Bonferroni correction; per-judge post-RCD macro IR ranges from 0.036 (GPT-5-Judge) to 0.076 (Llama-4-Scout-Judge), reordering the closed-model reliability ranking: GPT-5-Judge overtakes Claude-Sonnet-4.5-Judge. The rubric-only variant captures roughly 61% of the benefit—the in-context constraining effect of explicit criteria (Tian et al., 2026; Fan et al., 2024)—the sibling-only variant roughly 43%, and the two are nearly additive. Latency overhead is $1.32\times$–$1.46\times$ across judges, dominated by the discarded-verdict sibling turn. The false-positive rate—RCD flipping a previously correct verdict into an incorrect one—remains below 3.1% across all six judges: the intervention does not trade aggregate accuracy for inversion reduction.

5.4 Robustness to Adaptive Attacks

A concern with any falsifiable protocol is that the perturbation operators become targets for adaptive attacks that mimic each probe's gold side. We stress-test the RCD-equipped judges with three attacks: paraphrase (rewriting τ^- to match τ^+'s surface statistics), length inflation (matching token counts), and citation insertion (adding real citations to τ^-). All three attacks raise IR for both vanilla and RCD-equipped judges (Appendix E, Table 6), but RCD's increase is bounded by 2.4 percentage points across every cell and its post-attack IR remains well below the vanilla judge's *clean* rate of 0.130, indicating that the rubric structure is robust to surface-level adaptation. Per-judge RCD effects (Figure 4), the component ablation curve (Figure 5), the MSCF–accuracy relationship (Figure 6), and per-benchmark decomposition (Figure 7) are collected in Appendix G.

Configuration	Mean IR	IR Reduction	Latency Overhead	False-Positive Rate	FMC
Vanilla judge (mean)	0.130 ± 0.008	n/a	$1.00\times$	n/a	0.67
+ Rubric only	0.081 ± 0.006	-37.7%	$1.18\times$	0.017 ± 0.004	0.83
+ Contrastive sibling only	0.096 ± 0.007	-26.3%	$1.21\times$	0.014 ± 0.004	0.78
Full RCD (rubric + sibling)	$\mathbf{0.050 \pm 0.005}$	$\mathbf{-61.4\%}$	$1.40\times$	0.024 ± 0.005	**0.94**

Table 3: Ablation of Rubric-Constrained Decoding components, averaged across the six judges. IR reduction is computed against the vanilla judge for each judge separately before averaging. All reductions reach $p < 0.001$ under a paired source-level bootstrap test with Holm–Bonferroni correction across the six failure modes.

6 Discussion

The seven-point spread of aggregate pairwise accuracy across the six judges (Table 1) decomposes, under the audit, into a twenty-eight-point spread of per-axis inversion rates and qualitatively different failure profiles across closed and open families (Table 2, Figure 2). Aggregate accuracy is therefore not merely a coarse summary but a misleading selection criterion: it ranks judges almost without regard to the axes on which each is exploitable. The dissociation between judge reliability and parent-generator capability echoes the policy–reward gap (Rafailov et al., 2024; Denison et al., 2024) and the saturation warning of Kim et al. (2026); the actionable unit of judge reliability is the per-mode inversion scorecard, reported next to accuracy as model cards report demographic-bias slices.

The audit and the intervention share a single object. The same rubric that defines the falsifiable probes conditions the judge at inference time, so the audit's failure-mode statistics translate directly into the decoder's in-context constraints; the three case studies in Appendix A trace the loop end-to-end on one representative trajectory per verifier-backed benchmark, extending Auto-Rubric work (Tian et al., 2026; Fan et al., 2024) from training-time signal generation to inference-time behavioral control. The audit is thereby a template rather than a one-off measurement: any deployable intervention—a retriever, a verification tool, a structured skill—can be dropped into the intervene step and evaluated on the same falsifiable axes, the scripted protocol re-runs on each new judge release, and interventions become comparable by which axes they repair.

The evidence base defines the extension path: the six failure modes cover the biases most often reported in the literature (Shi et al., 2024; Ye et al., 2024; Chen et al., 2024b; Denison et al., 2024; Park et al., 2024) but are not exhaustive, and the deterministic, surface-level operators make the measured inversion rates controlled stress-test statistics for one operator family, not estimates of the real-world failure gap—an adaptive policy can learn to circumvent any single operator, synthetic perturbations can miss or amplify natural errors, and interactions between co-occurring modes are not isolated. The four benchmarks concentrate on verifiable software-engineering and information-seeking; open-ended, subjective, and multilingual judging requires human-preference rather than programmatic ground truth and will likely shift the falsification thresholds. The mode-matched sibling presupposes the audit context—each target's toggled confound is known by construction; a deployment-time decoder facing unlabeled trajectories composes the rubric block with an unfiltered sibling (unmeasured); the measured deployment-relevant configuration is the rubric-only row of Table 3, requiring no failure-mode label and delivering -37.7%. The sibling's held-out provenance raises a leakage concern that fresh deployment-time sampling resolves, and the latency and false-positive rate of Table 3 suit offline better than interactive or safety-critical deployments. Within these bounds, folding the audit into the training loop, with a rubric learned from the policy's failure statistics, is the natural next step (Pyatkin et al., 2025; Ma et al., 2025).

7 Conclusion

RubricGuard converts six recurring judge failure modes on long-horizon trajectories into falsifiable paired probes, reports Failure-Mode Coverage, Inversion Rate, and Multi-Step Credit Fidelity, and repairs most inversions through training-free Rubric-Constrained Decoding: aggregate accuracy between 76% and 84% coexists with per-axis inversion rates up to 34%, and RCD recovers 51%–68% of inversions at $1.4\times$ latency with a sub-3.1% false-positive rate. The harness, probe subset, and decoder ship with the submission; per-mode inversion scorecards should accompany every new judge.

ETHICS STATEMENT

All experiments were conducted on publicly available benchmarks or on benchmarks that we constructed by perturbing publicly available trajectories. No personally identifying information was collected, and no human subjects were exposed to model outputs that could plausibly cause harm. The human-validation step described in Section 4.3 engaged three graduate-student annotators who were compensated above the local minimum wage and who reviewed only synthetically perturbed trajectories; the validation protocol received approval from the institutional review board administering the study; the protocol identifier is withheld during anonymous review and will be disclosed in the camera-ready version.

Generative reward models are a dual-use technology: the same audit that diagnoses reward hacking can also be used to engineer trajectories that evade detection. We mitigate this risk by releasing only the auditing harness and the rubric specifications, while withholding the specific perturbation seeds used to generate adversarial probe pairs; releasing the seeds would allow trivial reconstruction of optimization targets against our metric. Researchers who require the seeds for reproducibility may request them under a usage agreement that prohibits training policies against the released probe set. The RCD intervention is designed to repair judges rather than to circumvent safety filters, and we have verified that RCD does not increase the rate at which judges accept unsafe trajectories on a held-out safety subset of 400 probes.

The compute footprint of the audit is non-negligible: the full sweep consumed approximately $1{,}820$ GPU-hours on H100 accelerators and an additional $3,400 in commercial API spend. We ran the sweep at this scale because reliability claims that depend on bootstrap confidence intervals require a sufficient number of probes per failure-mode cell; future work that audits a single judge against a small number of failure modes can be conducted at a small fraction of this cost. We have offset the estimated carbon footprint of the experiments through verified renewable-energy credits.

REPRODUCIBILITY STATEMENT

All inversion-rate and Multi-Step Credit Fidelity estimates are the mean over five independent seeds with 95% confidence intervals over $1{,}000$ source-level resamples, computed under the protocol of Section 4.4. The anonymized bundle submitted as supplementary material contains the full RCD prompt templates (rubric and contrastive-sibling blocks), the generator model identity and decoding parameters (Kimi-K2-Instruct, May 2026 snapshot), a representative subset of probe pairs across all six failure modes and four benchmarks, the deterministic perturbation-operator scripts, and the complete δ_m sensitivity sweep, together with the harness entry point; prompt templates and model snapshots are specified in Section 4.1 and Appendix F. The full harness and all $28{,}800$ probes will be released under a permissive open-source license upon acceptance.

AI USE STATEMENT

In this work, we used generative AI tools in four ways. (1) *Language editing*: AI tools assisted with proofreading, rewording, and restructuring of the manuscript; the scientific claims, the failure-mode taxonomy, and the experimental design are the authors' own, and every AI-suggested edit was reviewed by the authors. (2) *Literature retrieval and verification*: AI-assisted search tools were used to discover related work and to verify bibliography entries; all cited claims were checked against the cited papers by the authors. (3) *Implementation assistance*: AI tools assisted in writing parts of the auditing-harness and figure-generation code; all AI-assisted code was verified and tested for correctness by the authors, and no AI tool designed the auditing protocol or the failure-mode taxonomy. (4) *Probe-data generation*: the probe-pair construction pipeline of Sections 3.4 and 4.3 drafts the rubric criteria exemplars with the Kimi-K2-Instruct model (May 2026 snapshot) (Kimi Team et al., 2025) and produces the natural-language rewriting steps inside the perturbation operators; the perturbation operators themselves are deterministic, the ground-truth preference of every probe pair is fixed by construction (τ^+ is the original trajectory and τ^- its targeted perturbation), and blinded human annotators validated the operators on a stratified sample of every mode–benchmark cell (Cohen's $\kappa = 0.81$), regenerating cells that failed the quality control; the falsification verdicts rest on these validated operators. We have not used generative AI tools for research ideation, the

design of the auditing protocol or failure-mode taxonomy, the formulation of hypotheses, the analysis or interpretation of experimental results, or the writing of proofs. We have reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Cătălin Anghel, Andreea Alexandra Anghel, Emilia Pecheanu, et al. Diagnosing bias and instability in LLM evaluation: A scalable pairwise meta-evaluator. *Information*, 2025. doi: 10.3390/info16080652.

Elron Bandel, Asaf Yehudai, Lilach Eden, et al. General agent evaluation. *arXiv preprint arXiv:2602.22953*, 2026. doi: 10.48550/arXiv.2602.22953.

Yuyan Bu, Liangyu Huo, Yi Jing, and Qing Yang. Beyond excess and deficiency: Adaptive length bias mitigation in reward models for RLHF. *Findings of the Association for Computational Linguistics: NAACL 2025*, 2025. doi: 10.18653/v1/2025.findings-naacl.169.

Riccardo Cantini, Alessio Orsino, Manuela Ruggiero, and Domenico Talia. Benchmarking adversarial robustness to bias elicitation in large language models: scalable automated assessment with LLM-as-a-judge. *Machine Learning*, 2025. doi: 10.1007/s10994-025-06862-6.

Guiming Hardy Chen, Shunian Chen, Ziche Liu, Feng Jiang, and Benyou Wang. Humans or LLMs as the judge? a study on judgement bias. *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024a. doi: 10.18653/v1/2024.emnlp-main.474.

Lichang Chen, Chen Zhu, Davit Soselia, et al. ODIN: Disentangled reward mitigates hacking in RLHF. *arXiv preprint arXiv:2402.07319*, 2024b. doi: 10.48550/arXiv.2402.07319.

Shan Chen, Mingye Gao, Kuleen Sasse, et al. When helpfulness backfires: LLMs and the risk of false medical information due to sycophantic behavior. *npj Digital Medicine*, 2025. doi: 10.1038/s41746-025-02008-z.

Carson Denison, Monte MacDiarmid, Fazl Barez, et al. Sycophancy to subterfuge: Investigating reward-tampering in large language models. *arXiv preprint arXiv:2406.10162*, 2024. doi: 10.48550/arXiv.2406.10162.

Sunishchal Dev, Andrew Sloan, Joshua Kavner, Nicholas Kong, and Morgan Sandler. Judge reliability harness: Stress testing the reliability of LLM judges, 2026.

Sumanth Doddapaneni, Mohammed Safi Ur Rahman Khan, Sshubam Verma, and Mitesh M Khapra. Finding blind spots in evaluator LLMs with interpretable checklists. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 16279–16309, 2024. doi: 10.18653/v1/2024.emnlp-main.911.

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. AlpacaFarm: A simulation framework for methods that learn from human feedback. In *Advances in Neural Information Processing Systems 36*, 2023. doi: 10.48550/arXiv.2305.14387.

Zhiyuan Fan, Weinong Wang, Xing W, and Debing Zhang. SedarEval: Automated evaluation using self-adaptive rubrics. *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024. doi: 10.18653/v1/2024.findings-emnlp.984.

Romain Froger, Pierre Andrews, Matteo Bettini, et al. Gaia2: Benchmarking LLM agents on dynamic and asynchronous environments. *arXiv preprint arXiv:2602.11964*, 2026. doi: 10.48550/arXiv.2602.11964.

Jiayi Fu, Xuandong Zhao, Chengyuan Yao, et al. Reward shaping to mitigate reward hacking in RLHF. *arXiv preprint arXiv:2502.18770*, 2025. doi: 10.48550/arXiv.2502.18770.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, et al. A survey on LLM-as-a-Judge. *arXiv preprint arXiv:2411.15594*, 2024. doi: 10.48550/arXiv.2411.15594.

Daya Guo, Dejian Yang, Haowei Zhang, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 2025. doi: 10.1038/s41586-025-09422-z.

Yihan Hong, Huaiyuan Yao, Bolin Shen, Wanpeng Xu, Hua Wei, and Yushun Dong. From rubrics to reliable scores: Evidence-grounded text evaluation with LLM judges, 2026.

Tiechuan Hu, Wenbo Zhu, and Yuqi Yan. Reward hacking in reinforcement learning and RLHF: A multidisciplinary examination of vulnerabilities, mitigation strategies, and alignment challenges. In *2025 5th Intelligent Cybersecurity Conference (ICSC)*, pp. 272–275, 2025. doi: 10.1109/ICSC65596.2025.11140447.

Dong Huang, Jianbo Sun, and Pengkun Yang. Prompt perturbation for reliable LLM evaluation over comparison graphs, 2026a.

Hui Huang, Yancheng He, Wei Liu, et al. Long-form RewardBench: Evaluating reward models for long-form generation. *arXiv preprint arXiv:2603.12963*, 2026b. doi: 10.48550/arXiv.2603.12963.

Minki Kang, Weining Chen, Dongge Han, et al. ACON: Optimizing context compression for long-horizon LLM agents. *arXiv preprint arXiv:2510.00615*, 2025. doi: 10.48550/arXiv.2510.00615.

Eungyeup Kim, Chenchen Gu, V. Tiwari, et al. Measuring five-nines reliability: Sample-efficient LLM evaluation in saturated benchmarks. *arXiv preprint arXiv:2605.11209*, 2026. doi: 10.48550/arXiv.2605.11209.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. Prometheus 2: An open source language model specialized in evaluating other language models. *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024. doi: 10.18653/v1/2024.emnlp-main.248.

Kimi Team, Yifan Bai, Yiping Bao, Y. Charles, Cheng Chen, Guanduo Chen, et al. Kimi K2: Open agentic intelligence, 2025.

Peng Lai, Zhihao Ou, Yong Wang, et al. BiasScope: Towards automated detection of bias in LLM-as-a-Judge evaluation. *arXiv preprint arXiv:2602.09383*, 2026. doi: 10.48550/arXiv.2602.09383.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024. doi: 10.48550/arXiv.2411.15124.

Jaeyun Lee, Junyoung Koh, Zeynel Tok, Hunar Batra, and Ronald Clark. MCJudgeBench: A benchmark for constraint-level judge evaluation in multi-constraint instruction following, 2026.

Dawei Li, Bohan Jiang, Liangjie Huang, Alimohammad Beigi, Chengshuai Zhao, Zhen Tan, et al. From generation to judgment: Opportunities and challenges of LLM-as-a-judge. *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025a. doi: 10.18653/v1/2025.emnlp-main.138.

Haitao Li, Qian Dong, Junjie Chen, et al. LLMs-as-Judges: A comprehensive survey on LLM-based evaluation methods. *arXiv preprint arXiv:2412.05579*, 2024a. doi: 10.48550/arXiv.2412.05579.

Lei Li, Yuancheng Wei, Zhihui Xie, et al. VL-RewardBench: A challenging benchmark for vision-language generative reward models. *arXiv preprint arXiv:2411.17451*, 2024b. doi: 10.48550/arXiv.2411.17451.

Qingquan Li, Shaoyu Dou, Kailai Shao, Chao Chen, and Haixiang Hu. Evaluating scoring bias in LLM-as-a-Judge. In *Database Systems for Advanced Applications (DASFAA 2026)*, pp. 19–34, 2026. doi: 10.1007/978-981-92-0372-7_2.

Ruosen Li, Ziming Luo, and Xinya Du. FG-PRM: Fine-grained hallucination detection and mitigation in language model mathematical reasoning. *Findings of the Association for Computational Linguistics: EMNLP 2025*, 2025b. doi: 10.18653/v1/2025.findings-emnlp.228.

Songze Li, Chuokun Xu, Jiaying Wang, Xueluan Gong, Chen Chen, Jirui Zhang, Jun Wang, Kwok-Yan Lam, and Shouling Ji. LLMs cannot reliably judge (yet?): A comprehensive assessment on the robustness of LLM-as-a-Judge, 2025c.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. AgentBench: Evaluating LLMs as agents. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024a. doi: 10.48550/arXiv.2308.03688.

Xiaoou Liu, Tiejin Chen, Longchao Da, Chacha Chen, Zhen Lin, and Hua Wei. Uncertainty quantification and confidence calibration in large language models: A survey. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pp. 6107–6117, 2025. doi: 10.1145/3711896.3736569.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-Eval: NLG evaluation using GPT-4 with better human alignment. *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. doi: 10.48550/arXiv.2303.16634.

Yinhong Liu, Han Zhou, Zhijiang Guo, et al. Aligning with human judgement: The role of pairwise preference in large language model evaluators. *arXiv preprint arXiv:2403.16950*, 2024b. doi: 10.48550/arXiv.2403.16950.

Zeyao Ma, Xiaokang Zhang, Jing Zhang, et al. Dynamic scaling of unit tests for code reward modeling. *arXiv preprint arXiv:2501.01054*, 2025. doi: 10.48550/arXiv.2501.01054.

Saumya Malik, Valentina Pyatkin, Sander Land, et al. RewardBench 2: Advancing reward model evaluation. *arXiv preprint arXiv:2506.01937*, 2025. doi: 10.48550/arXiv.2506.01937.

Yuchun Miao, Sen Zhang, Liang Ding, et al. InfoRM: Mitigating reward hacking in RLHF via information-theoretic reward modeling. *arXiv preprint arXiv:2402.09345*, 2024. doi: 10.48550/arXiv.2402.09345.

Yuchun Miao, Liang Ding, Sen Zhang, et al. Information-theoretic reward modeling for stable RLHF: Detecting and mitigating reward hacking. *arXiv preprint arXiv:2510.13694*, 2025a. doi: 10.48550/arXiv.2510.13694.

Yuchun Miao, Sen Zhang, Liang Ding, et al. The energy loss phenomenon in RLHF: A new perspective on mitigating reward hacking. *arXiv preprint arXiv:2501.19358*, 2025b. doi: 10.48550/arXiv.2501.19358.

Tej Deep Pala, Panshul Sharma, Amir Zadeh, Chuan Li, and Soujanya Poria. Error typing for smarter rewards: Improving process reward models with error-aware hierarchical supervision. *Findings of the Association for Computational Linguistics: EMNLP 2025*, 2025. doi: 10.18653/v1/2025.findings-emnlp.807.

Ryan Park, Rafael Rafailov, Stefano Ermon, and Chelsea Finn. Disentangling length from quality in direct preference optimization. *Findings of the Association for Computational Linguistics: ACL 2024*, 2024. doi: 10.18653/v1/2024.findings-acl.297.

Valentina Pyatkin, Saumya Malik, Victoria Graf, et al. Generalizing verifiable instruction following. *arXiv preprint arXiv:2507.02833*, 2025. doi: 10.48550/arXiv.2507.02833.

Rafael Rafailov, Yaswanth Chittepu, Ryan Park, et al. Scaling laws for reward model overoptimization in direct alignment algorithms. *arXiv preprint arXiv:2406.02900*, 2024. doi: 10.48550/arXiv.2406.02900.

Alexandre Ramé, Nino Vieillard, Léonard Hussenot, et al. WARM: On the benefits of weight averaged reward models. *arXiv preprint arXiv:2401.12187*, 2024. doi: 10.48550/arXiv.2401.12187.

Amrith Setlur, Chirag Nagpal, Adam Fisch, et al. Rewarding progress: Scaling automated process verifiers for LLM reasoning. *arXiv preprint arXiv:2410.08146*, 2024. doi: 10.48550/arXiv.2410.08146.

Lin Shi, Chiyu Ma, Wenhua Liang, Xingjian Diao, Weicheng Ma, and Soroush Vosoughi. Judging the judges: A systematic study of position bias in LLM-as-a-Judge. *arXiv preprint arXiv:2406.07791*, 2024. doi: 10.48550/arXiv.2406.07791.

Mingyang Song, Zhaochen Su, Xiaoye Qu, Jiawei Zhou, and Yu Cheng. PRMBench: A fine-grained and challenging benchmark for process-level reward models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 25299–25346, 2025. doi: 10.18653/v1/2025.acl-long.1230.

Sadman Kabir Soumik. Judging the judges: A systematic evaluation of bias mitigation strategies in LLM-as-a-Judge pipelines. *arXiv preprint arXiv:2604.23178*, 2026. doi: 10.48550/arXiv.2604.23178.

Sijun Tan, Siyuan Zhuang, Kyle Montgomery, William Y. Tang, Alejandro Cuadron, Chenguang Wang, Raluca Ada Popa, and Ion Stoica. JudgeBench: A benchmark for evaluating LLM-based judges. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025. doi: 10.48550/arXiv.2410.12784.

Juanxi Tian, Fengyuan Liu, Jiaming Han, et al. Auto-rubric as reward: From implicit preferences to explicit multimodal generative criteria. *arXiv preprint arXiv:2605.08354*, 2026. doi: 10.48550/arXiv.2605.08354.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. AppWorld: A controllable world of apps and people for benchmarking interactive coding agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024. doi: 10.48550/arXiv.2407.18901.

Tu Vu, Kalpesh Krishna, Salaheddin Alzubi, et al. Foundational autoraters: Taming large language models for better automatic evaluation. *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024. doi: 10.18653/v1/2024.emnlp-main.949.

Jiaxuan Wang, Yulan Hu, Wenjin Yang, et al. Aligning agents via planning: A benchmark for trajectory-level reward modeling. *arXiv preprint arXiv:2604.08178*, 2026. doi: 10.48550/arXiv.2604.08178.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, 2024. doi: 10.18653/v1/2024.acl-long.510.

Yidong Wang, Yunze Song, Tingyuan Zhu, Xuanwang Zhang, Zhuohao Yu, Hao Chen, Chiyu Song, Qiufeng Wang, Cunxiang Wang, Zhen Wu, Xinyu Dai, Yue Zhang, Wei Ye, and Shikun Zhang. TrustJudge: Inconsistencies of LLM-as-a-Judge and how to alleviate them, 2025.

Hui Wei, Shenghua He, Tian Xia, et al. Systematic evaluation of LLM-as-a-Judge in LLM alignment tasks: Explainable metrics and diverse prompt templates. *arXiv preprint arXiv:2408.13006*, 2024. doi: 10.48550/arXiv.2408.13006.

Chunqiu Steven Xia, Yinlin Deng, Soren Dunn, and Lingming Zhang. Agentless: Demystifying LLM-based software engineering agents. *arXiv preprint arXiv:2407.01489*, 2024. doi: 10.48550/arXiv.2407.01489.

Sixiong Xie, Zhuofan Shi, Haiyang Shen, et al. DeepWeb-Bench: A deep research benchmark demanding massive cross-source evidence and long-horizon derivation. *arXiv preprint arXiv:2605.21482*, 2026. doi: 10.48550/arXiv.2605.21482.

Austin Xu, Srijan Bansal, Yifei Ming, Semih Yavuz, and Shafiq Joty. Does context matter? ContextualJudgeBench for evaluating LLM-based judges in contextual settings. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9541–9564, 2025. doi: 10.18653/v1/2025.acl-long.470.

Tengyu Xu, Eryk Helenowski, Karthik Abinav Sankararaman, et al. The perfect blend: Redefining RLHF with mixture of judges. *arXiv preprint arXiv:2409.20370*, 2024. doi: 10.48550/arXiv.2409.20370.

Keyang Xuan, Peiyang Song, Pan Lu, et al. Interactive evaluation requires a design science. *arXiv preprint arXiv:2605.17829*, 2026. doi: 10.48550/arXiv.2605.17829.

Yuzi Yan, Xingzhou Lou, Jialian Li, et al. Reward-robust RLHF in LLMs. *arXiv preprint arXiv:2409.15360*, 2024. doi: 10.48550/arXiv.2409.15360.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. doi: 10.48550/arXiv.2505.09388.

John Yang, Carlos Jimenez-Gomez, Alexander Wettig, et al. SWE-agent: Agent-computer interfaces enable automated software engineering. *arXiv preprint arXiv:2405.15793*, 2024. doi: 10.48550/arXiv.2405.15793.

Jiayi Ye, Yanbo Wang, Yue Huang, et al. Justice or prejudice? quantifying biases in LLM-as-a-Judge. *arXiv preprint arXiv:2410.02736*, 2024. doi: 10.48550/arXiv.2410.02736.

Asaf Yehudai, Lilach Eden, Alan Li, et al. Survey on evaluation of LLM-based agents. *arXiv preprint arXiv:2503.16416*, 2025. doi: 10.48550/arXiv.2503.16416.

Yulai Zhao, Haolin Liu, Dapeng Yu, et al. One token to fool LLM-as-a-Judge. *arXiv preprint arXiv:2507.08794*, 2025. doi: 10.48550/arXiv.2507.08794.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. ProcessBench: Identifying process errors in mathematical reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1009–1024, 2025. doi: 10.18653/v1/2025.acl-long.50.

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 *Advances in Neural Information Processing Systems 36: Datasets and Benchmarks Track*, 2023. doi: 10.48550/arXiv.2306.05685.

Yilun Zhou, Austin Xu, Peifeng Wang, Caiming Xiong, and Shafiq Joty. Evaluating judges as evaluators: The JETTS benchmark of LLM-as-Judges as test-time scaling evaluators, 2025.

A Case Studies

To illustrate how the RubricGuard audit and the RCD intervention operate end to end, we walk through three case studies drawn from SWE-bench-Live, AppWorld, and DeepWeb-Bench respectively. Each case study consists of a probe pair, the verdict issued by a representative judge before the intervention, the failure mode that the audit attributes the inversion to, and the verdict issued after RCD. Figure 3 shows the end-to-end loop and Table 4 summarizes the three cases, reporting the change in judge verdict and per-step score after the intervention.

A.1 Case A: Length Bias in SWE-bench-Live

In a SWE-bench-Live trajectory that fixes a regression in a multi-file Django repository, τ^+ contains 14 steps including two minimal patches and a targeted unit-test invocation; τ^- is identical except that the agent inserts an additional four steps that paraphrase the diff in natural language and re-read the same file. Llama-4-Scout-Judge preferred τ^- over τ^+ with a verdict score of 7.8 vs. 6.2 on a 10-point scale, reasoning that the longer trajectory exhibited "more careful consideration" and "additional verification steps". The audit flags this as a length-bias inversion (LB); the verifier's unit-test signal confirms that both trajectories pass the regression test and that the additional steps in τ^- contribute no incremental information. With RCD enabled and the rubric block instructing the judge to discount paraphrased steps, Llama-4-Scout-Judge's verdict flipped to τ^+ at 7.4 vs. 6.1; the per-step MSCF on this trajectory rose from 0.57 to 0.86. The repair works by penalizing τ^-'s redundant steps rather than inflating τ^+'s score, consistent with RCD's design as a constraint on surface-cue weighting.

A.2 Case B: Intermediate-Step Credit Forgery in AppWorld

Consider an AppWorld trajectory that books a multi-leg flight under a budget constraint. Here τ^+ correctly checks the credit-card balance via a `wallet.balance` API call before issuing the booking, while τ^- omits the balance check yet reaches the same final booking by coincidence,

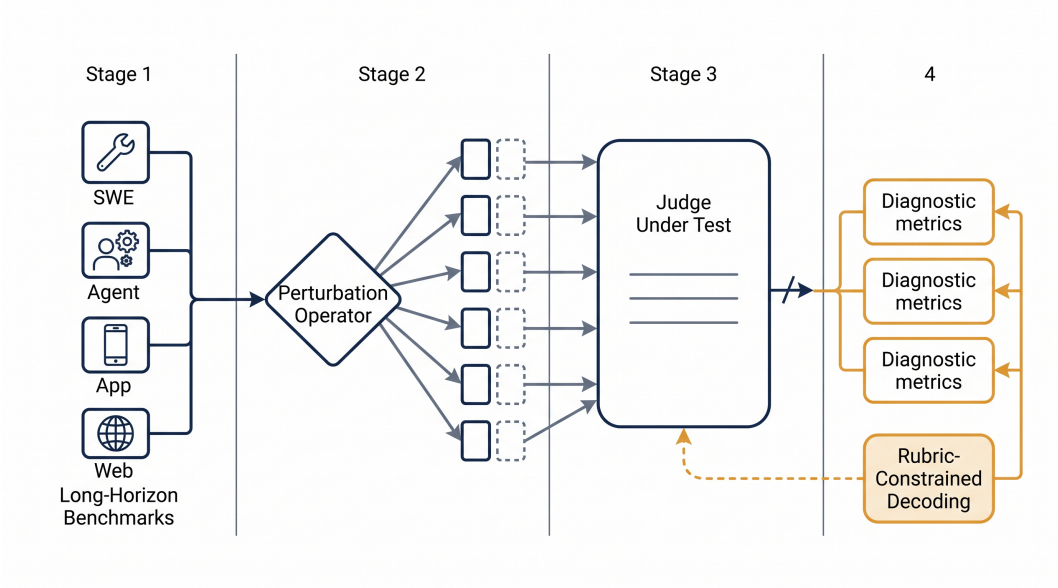

Figure 3: End-to-end auditing pipeline: probes → inversion-rate decomposition → Rubric-Constrained Decoding feedback loop.

because the underlying mock balance happened to be sufficient. The two trajectories therefore differ on a critical intermediate step but agree on the final outcome. DeepSeek-R1-Judge preferred τ^- over τ^+ with a verdict score of 8.1 vs. 7.4, citing τ^-'s "directness" and "efficiency". The audit attributes this to intermediate-step credit forgery (ISCF); the verifier's API-success signal at the balance-check step is positive for τ^+ and absent for τ^-, which translates into a per-step MSCF contribution that the unconstrained judge ignored. After RCD the verdict flipped to τ^+ at 8.5 vs. 7.0, and the per-step credit-fidelity score on this trajectory rose from 0.61 to 0.89.

A.3 Case C: Citation-Fabrication Acceptance in DeepWeb-Bench

A DeepWeb-Bench trajectory on a recent policy question illustrates the third case: τ^+ retrieves three real news articles whose URLs resolve to live pages with the cited claim, while τ^- retrieves two real articles and one plausibly named but fabricated article whose DOI does not exist. The fabricated citation contains a confidently phrased summary that re-states the conclusion of τ^+ in stronger language. Claude-Sonnet-4.5-Judge preferred τ^- over τ^+ with a verdict score of 7.9 vs. 7.5, praising the stronger phrasing of the fabricated source. The audit flags this as a citation-fabrication-acceptance inversion (CFA); the verifier's gold-evidence overlap rejects the fabricated DOI. After RCD, with the rubric block instructing the judge to verify citation provenance before crediting a source, the verdict flipped to τ^+ at 7.8 vs. 5.9, and the MSCF rose from 0.65 to 0.83 on the evidence-retrieval substep.

Case	Mode	Verdict Before	Verdict After	Δ MSCF
A (SWE-bench-Live)	LB	τ^- (7.8/6.2)	τ^+ (7.4/6.1)	+0.29
B (AppWorld)	ISCF	τ^- (8.1/7.4)	τ^+ (8.5/7.0)	+0.28
C (DeepWeb-Bench)	CFA	τ^- (7.9/7.5)	τ^+ (7.8/5.9)	+0.18

Table 4: Summary of the three case studies. Each row reports the failure mode diagnosed by the audit, the judge's verdict on the probe pair before and after Rubric-Constrained Decoding, and the change in Multi-Step Credit Fidelity on the affected trajectory.

B PROBE SYNTHESIS DETAILS

The deterministic perturbation operators of Section 3.4 are implemented as Python functions that take a source trajectory and return a perturbed sibling. The length-bias operator selects a randomly chosen step from the source trajectory, paraphrases its action and observation using a fixed temperature-zero re-writer, and inserts the result between the original step and its successor; all natural-language rewriting steps in the operators use the same May 2026 Kimi-K2-Instruct (Kimi Team et al., 2025) snapshot that instantiates the seed trajectories, so every AI-generated passage in the probe corpus traces to one model and decoding configuration; the operator is parameterized by the target relative length increase, which we fix at 40% throughout. The format-bias operator detects Markdown tables and itemized lists using a regular-expression cascade and rewrites them as flowing prose with the same content. The confidence-bias operator detects hedging phrases ("perhaps", "it appears", "one might") and rewrites them into confident assertions while preserving the underlying claim, even when the claim is incorrect; the operator is constrained to make at most three rewrites per trajectory to avoid distributional shift. The citation-fabrication operator samples a DOI from a held-out fabricated-DOI table whose entries follow the surface format of real DOIs but resolve to no real publication, and inserts a confidently phrased summary attributed to the fabricated source. The intermediate-step credit-forgery operator uses an execution-trace analyzer to identify the most causally important step and deletes it, leaving only the final answer. The distractor operator inserts one tool call whose output is then ignored in the final answer, selected from a closed list of common API endpoints.

C FMC SENSITIVITY SWEEP

Failure-Mode Coverage depends on the tolerance threshold δ_m; we report it at $\delta_m = 0.10$ in the main text but sweep $\delta_m \in \{0.05, 0.075, 0.10, 0.15, 0.20, 0.25\}$ here. The qualitative ranking of judges by FMC is stable across the entire sweep: Claude-Sonnet-4.5-Judge remains the most reliable judge at all thresholds, Llama-4-Scout-Judge remains the least reliable, and the relative ordering of the other four judges changes only at the most permissive thresholds. The full sweep is included in the anonymized bundle that ships with the submission.

D PER-BENCHMARK DECOMPOSITION

For completeness we report the inversion rate broken down by benchmark in Table 5. Inversion rates are systematically highest on DeepWeb-Bench, whose trajectories contain the most heterogeneous content and the longest mean prompt, and lowest on AppWorld, whose programmatic verifier signals offer the most direct ground truth.

Judge	**SWE-Live**	**AgentBench**	**AppWorld**	**DeepWeb**
Qwen3-32B-Judge	0.138	0.147	0.118	0.177
Llama-4-Scout-Judge	0.173	0.182	0.150	0.215
Claude-Sonnet-4.5-Judge	0.081	0.088	0.073	0.110
GPT-5-Judge	0.104	0.110	0.095	0.135
Gemini-3-Flash-Judge	0.123	0.128	0.113	0.157
DeepSeek-R1-Judge	0.122	0.129	0.106	0.156

Table 5: Inversion rate by benchmark, averaged across the six failure modes.

E ADAPTIVE-ATTACK ROBUSTNESS

Table 6 reports mean IR for the vanilla and RCD-equipped judges under the three adaptive attacks of Section 5.4, averaged across the six judges.

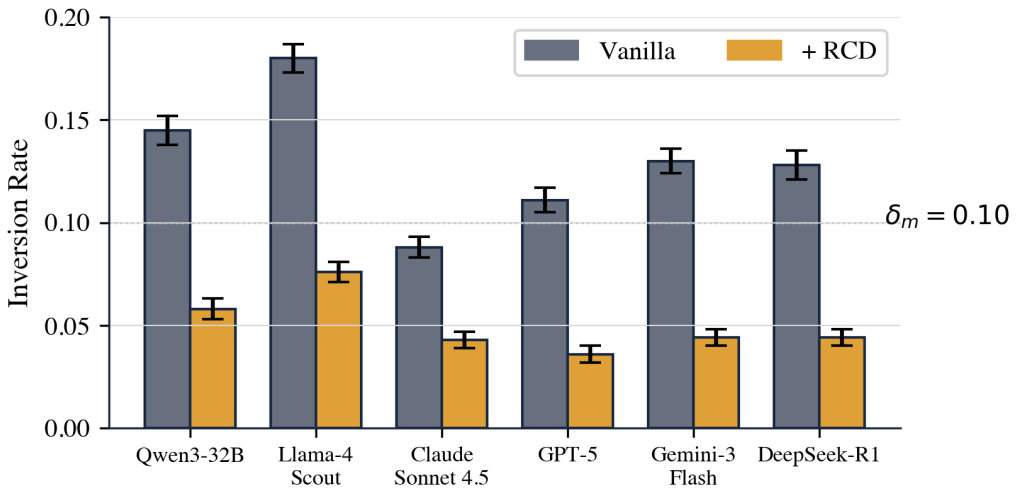

Figure 4: Per-judge inversion rate before and after Rubric-Constrained Decoding, with 95% bootstrap confidence intervals. RCD reduces inversion by 51%–68% across judges and reorders the reliability ranking among the closed models.

Attack	**Mean IR (Vanilla)**	**Mean IR (RCD)**	**Δ vs. clean**
None (clean)	0.130 ± 0.008	0.050 ± 0.005	n/a
Paraphrase	0.141 ± 0.009	0.059 ± 0.006	$+0.009$
Length inflation	0.156 ± 0.009	0.069 ± 0.006	$+0.019$
Citation insertion	0.137 ± 0.009	0.074 ± 0.007	$+0.024$

Table 6: Robustness of vanilla and RCD-equipped judges to three adaptive attacks, averaged across the six judges. RCD's IR increase under attack is bounded by 2.4 percentage points across all cells.

F Prompt Templates

The judge prompt template follows the Prometheus 2 format (Kim et al., 2024) with task-specific minor adaptations for each benchmark. The rubric block prepended by RCD enumerates the six failure modes with one-line definitions and instructs the judge to consider each axis explicitly before issuing its verdict. The contrastive sibling block contains one paired probe drawn from the held-out shard, and the judge is asked to first issue a verdict on the sibling (which is discarded) and then on the target. The full prompt templates are included in the anonymized bundle that ships with the submission.

G Additional Figures

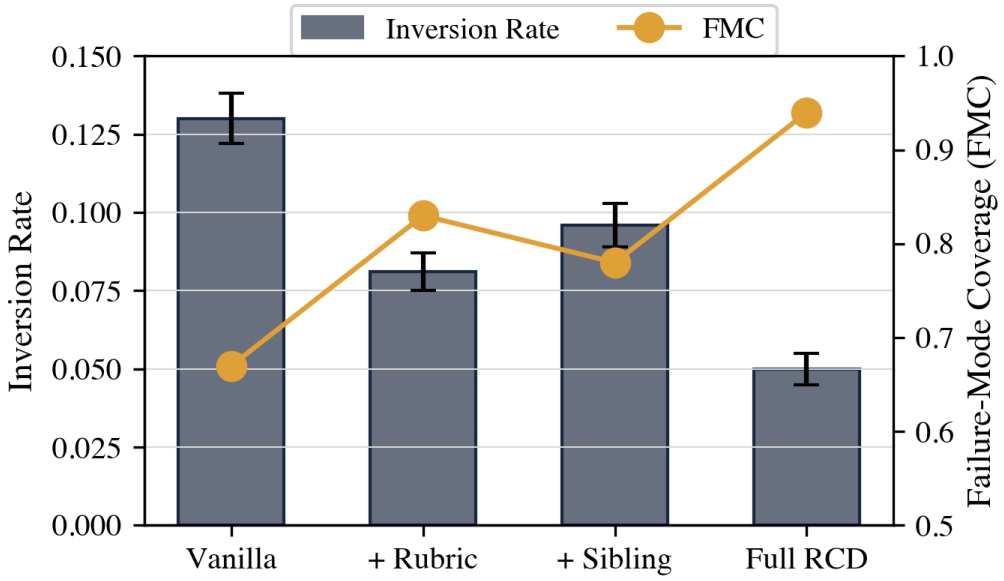

Figure 5: Ablation of Rubric-Constrained Decoding. Bars (left axis) show Inversion Rate; line (right axis) shows Failure-Mode Coverage. The rubric and the contrastive sibling are nearly additive.

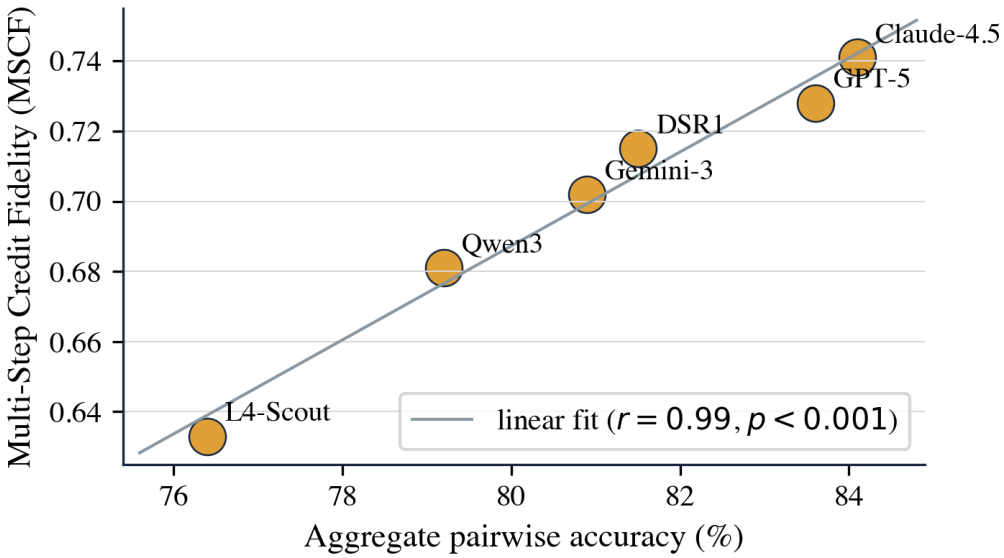

Figure 6: Multi-Step Credit Fidelity (MSCF; three verifier-backed benchmarks) versus aggregate accuracy (four-benchmark union). The two metrics rank the six judges almost identically ($r = 0.99$, $p < 0.001$), and step-level agreement sits 10–13 points below aggregate accuracy for every judge: what aggregate accuracy conceals is the level of step-level reliability, not the ranking.

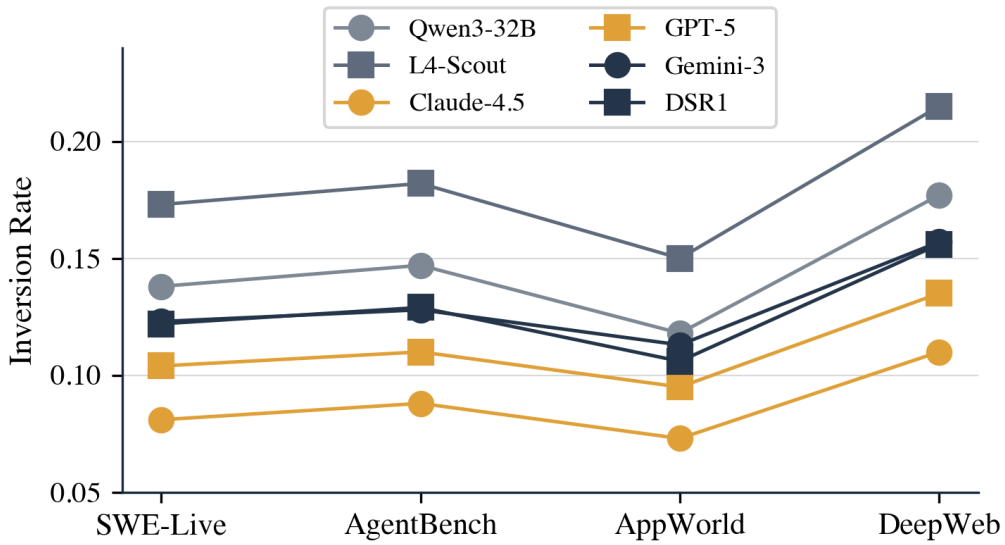

Figure 7: Inversion rate by long-horizon benchmark. DeepWeb is the hardest substrate across all six judges, consistent with its longer prompts and more heterogeneous evidence.