FROM-BASE RLVR FOR VERIFY-ONLY JUDGES: A RULE-BASED REWARD TURNS BASE MODELS INTO TRAJECTORY AUDITORS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning from verifiable rewards (RLVR) is the standard recipe for improving how language models *solve* tasks. We ask whether the same rule-based binary signal can instead train an effective *verify-only judge* directly from a base model. Casting judging as a balanced accept/reject task and rewarding the model only when its parsed verdict matches the ground truth—no reward model, preference pairs, or human rationales—we optimize with GRPO on two verification-hard domains: auditing GAIA agent trajectories and verifying MuSiQue multi-hop factual answers. The payoff on trajectory auditing is large and replicates across three model families (Qwen3-8B, Qwen2.5-7B, Llama-3.1): e.g. verify accuracy improves $0.579 \rightarrow 0.903$ (McNemar p=1.8e−47), transfers to a real out-of-distribution set of multi-agent failures, and increases with task difficulty. On multi-hop factual verification, a compact 8B judge attains a lower miss rate than four frontier zero-shot judges, albeit with a false-alarm trade-off. We further report two honest negatives—tripling the data gives no gain (the recipe is sample-efficient), and the multi-hop factual payoff does not replicate on one base—that sharpen when the payoff holds. From-base RLVR is a simple, reproducible route to trustworthy verify-only judges.

1 INTRODUCTION

Reinforcement learning from verifiable rewards (RLVR) has become the dominant recipe for improving how language models *solve* problems: a rule-based checker returns a binary reward and the model is optimized to produce correct solutions (Shao et al., 2024; DeepSeek-AI, 2025; Lambert et al., 2024a; Kimi Team, 2025; Yang et al., 2024). Yet many important tasks are *solving-easy but verification-hard*: given a candidate multi-hop factual answer (Trivedi et al., 2022) or a long agent trajectory (Mialon et al., 2023), deciding whether it is *correct* is itself hard, and even frontier models used as zero-shot judges (Zheng et al., 2023) accept a large fraction of plausible-but-wrong candidates, and exhibit systematic biases (Chen et al., 2024). Reliable *judges*—not just better solvers—are what agentic pipelines (Yao et al., 2023), reward modeling (Lambert et al., 2024b), and automatic evaluation (Liu et al., 2023) increasingly depend on.

We study a simple question: *can the same rule-based RLVR signal that improves solving instead be used to train a good verify-only judge, starting from a base model?* Concretely, we cast judging as a balanced binary task (accept the gold candidate, reject a plausible distractor), reward the model only when its parsed verdict matches the ground truth, and optimize with GRPO (Shao et al., 2024)—no learned reward model, no preference pairs, and no human rationales. We call this recipe *from-base RLVR for verify-only judges*, and we evaluate it on two verification-hard domains: auditing GAIA agent trajectories and verifying MuSiQue multi-hop factual answers.

Our central finding—the **payoff**—is that from-base RLVR turns a base model into an effective judge on the trajectory domain, and the effect replicates across model families (Qwen3-8B, Qwen2.5-7B, and Llama-3.1). On agent-trajectory auditing the gain is large and highly significant (e.g. Qwen2.5-7B verify_acc $0.579 \rightarrow 0.903$, McNemar p=1.8e−47), transfers to a *real* OOD set of genuine multi-agent failures, and *increases with task difficulty*. A compact 8B from-base judge attains a lower miss rate on multi-hop verification than four frontier zero-shot judges. We also report two honest nega-

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.

tives that sharpen the claim: tripling the training data gives no gain (the recipe is sample-efficient), and the multi-hop factual payoff does *not* replicate on one base (a base-dependent boundary). Two design choices are central and, we argue, the reason the recipe is both simple and reproducible. We keep the judge *verify-only*: it emits a single accept/reject verdict rather than a score, a rationale, or a repaired answer, so the reward is a rule-based check against a known label and needs no reward model, preference data, or human explanations. And we start *from a base model* rather than a judge that was supervised-fine-tuned for the task, so any improvement is attributable to the RLVR signal itself rather than to judging-specific supervision. Because both the reward and the evaluation reduce to matching a parsed verdict against ground truth, the entire pipeline is deterministic to score and cheap to reproduce, which also lets us charge silent or malformed verdicts as verification failures rather than discarding them.

Our contributions are:

- We cast verify-only judging as an RLVR target and show that the same rule-based signal used for solving yields a large from-base payoff on verification-hard trajectory auditing, without any learned reward model, preference pairs, or human rationales.
- We show that this payoff is not idiosyncratic to one backbone: it replicates across three model families (Qwen3-8B, Qwen2.5-7B, Llama-3.1) and transfers to a real out-of-distribution set of genuine multi-agent failures, with every comparison backed by paired statistical tests.
- We characterize *when* the payoff holds through difficulty, data-scale, and reward-design analyses, and we report two honest negatives—no gain from tripling the data and a base-dependent boundary on multi-hop verification—that sharpen rather than inflate the claim.

2 Related Work

RLVR and RL for reasoning. Reinforcement learning has been the workhorse for aligning and improving large language models, from PPO (Schulman et al., 2017) and RLHF (Ouyang et al., 2022) to scalable AI-feedback variants (Bai et al., 2022; Lee et al., 2024). A recent shift replaces the learned reward model with a rule-based *verifiable* reward (RLVR), which has driven large gains in mathematical and coding reasoning where a checker supplies a binary correctness signal (Shao et al., 2024; DeepSeek-AI, 2025; Lambert et al., 2024a; Kimi Team, 2025; Yang et al., 2024). The optimizer landscape has correspondingly simplified toward critic-free, REINFORCE-style updates (Ahmadian et al., 2024; Li et al., 2024b) and refined credit assignment (Kazemnejad et al., 2024). All of this prior work applies RLVR to *solving* tasks. Our focus is orthogonal: we ask whether the same rule-based RLVR signal can turn a *base* model into an effective *verify-only judge*, and we find that on solving-easy but verification-hard tasks the payoff is large.

LLM-as-a-judge and automatic evaluation. Using LLMs to judge candidate answers underpins RLHF reward models and automatic evaluation (Zheng et al., 2023; Lambert et al., 2024b). A large body of work builds prompt-based or fine-tuned judges — G-Eval (Liu et al., 2023), AlpacaFarm (Dubois et al., 2023), Prometheus (Kim et al., 2024), JudgeLM (Zhu et al., 2023), PandaLM (Wang et al., 2024c), and Auto-J (Li et al., 2024a) — yet these judges are known to exhibit systematic biases such as position and self-preference bias (Wang et al., 2024a; Chen et al., 2024; Koo et al., 2024). Most judges are either frontier models used zero-shot or models trained by preference learning on human labels. We instead train a judge *from base* with a purely rule-based binary reward (verdict matches ground truth), needing no preference pairs and no human rationales, and we benchmark it against frontier zero-shot judges.

Reward modeling and verifiers. Verifiers trained to score candidate solutions date back to GSM8K (Cobbe et al., 2021), and subsequent work contrasts outcome-based against process-based feedback (Uesato et al., 2022; Lightman et al., 2024; Wang et al., 2024b). Recent verifiers are cast as generative or self-generated reward models (Zhang et al., 2025a; Yuan et al., 2024), and dedicated benchmarks probe their ability to localize errors (Zheng et al., 2025). These verifiers are almost always trained by supervised fine-tuning or preference learning on solving-domain data; we instead *RL-train* a verify-only judge from a base model with a binary verifiable reward, and study when this transfers to verification-hard domains beyond math.

Self-correction and self-verification. A parallel line asks models to critique and revise their own outputs, via self-refinement (Madaan et al., 2023), verbal reinforcement (Shinn et al., 2023), self-verification (Weng et al., 2023), or verify-and-edit pipelines (Zhao et al., 2023). However, Huang et al. (2024) show that LLMs cannot reliably self-correct their own reasoning without external signals, and Kumar et al. (2024) find that naïve multi-turn RL for self-correction collapses. This motivates an *independent* judge rather than self-critique: our verify-only model audits a separate candidate's trajectory rather than revising its own.

LLM agents and trajectory auditing. LLM agents built on reasoning-and-acting (Yao et al., 2023) and reflective (Shinn et al., 2023) paradigms are evaluated by benchmarks such as AgentBench (Liu et al., 2024), WebArena (Zhou et al., 2024), and GAIA (Mialon et al., 2023), with multi-agent debate (Du et al., 2024) and blame-attribution (Zhang et al., 2025b) probing failure analysis. Auditing whether a long-horizon agent trajectory is correct is itself a verification-hard problem. We treat agent-trajectory auditing (GAIA / Who&When) as one of our two evaluation domains and show that from-base RLVR closes much of the frontier gap here across model families.

Multi-hop QA, fact verification, and retrieval. Multi-hop factual QA (Trivedi et al., 2022; Yang et al., 2018; Ho et al., 2020) requires composing evidence across documents, typically supported by retrieval (Lewis et al., 2020; Karpukhin et al., 2020). Verifying such answers connects to fact verification (Thorne et al., 2018), hallucination detection (Manakul et al., 2023; Ji et al., 2023), and attribution (Rashkin et al., 2023), where deciding whether a candidate is faithful is as hard as producing it. We use MuSiQue as our multi-hop factual domain and identify a base-model-dependent boundary where from-base RLVR trades a low miss rate against a higher false-accept rate.

Verification versus generation difficulty. The intuition that verifying can differ sharply from generating underlies best-of-n and verifier-reranking methods (Wang et al., 2023; Li et al., 2023) and test-time compute scaling (Snell et al., 2024), building on chain-of-thought reasoning (Wei et al., 2022). We study the regime where verifying is itself hard — a frontier zero-shot judge misses a large fraction of plausible-but-wrong candidates — and show that targeted RLVR on a base model substantially improves verification without any solving supervision.

Positioning. Three lines of work are closest, and our contribution is defined by how it differs from each. Relative to *RLVR for solving*, we keep the identical rule-based binary signal but change the target from producing a correct answer to emitting a correct accept/reject verdict, so the reward checks a verdict against a label rather than an answer against a checker; this repurposing, not a new optimizer, is what we study. Relative to *trained reward models and LLM judges*, which are typically obtained by supervised or preference fine-tuning and often emit scalar scores or rationales, our judge is verify-only and trained from a base model with no preference data, no reward model, and no human rationales, which isolates the effect of the verifiable signal and keeps scoring deterministic. Relative to *self-verification and self-correction*, where a model checks or repairs its own generations, we train a dedicated third-party judge on paired correct/wrong candidates and evaluate it on verification-hard tasks and on genuine out-of-distribution failures rather than on a model's own trace. Finally, unlike work that reports a single aggregate judging metric, we separate false alarms from misses throughout and report where the recipe helps and where it does not, including two negative results, so the boundaries of from-base RLVR are explicit rather than implied.

3 From-Base RLVR for Verify-Only Judges

Verify-only judge task. A judge receives a task, a candidate answer (multi-hop QA) or a candidate reasoning trajectory (agent auditing), and must emit a single verdict $\hat{y} \in \{\text{CORRECT}, \text{WRONG}\}$. Each evaluation item is paired so that exactly one candidate is CORRECT (the gold answer / gold trajectory) and one is WRONG (a plausible distractor answer / a trajectory with an injected error), giving a balanced binary task. We report overall accuracy (*verify_acc*), the rate of correctly accepting good candidates (*va_true*), the rate of correctly rejecting bad candidates (*va_false*), the false-alarm rate (good candidate not accepted, i.e. va_true's complement), and—our headline verification-hard metric—the **miss** rate (a wrong candidate not rejected, i.e. va_false's complement). We adopt a strict verdict protocol: a judge must emit an explicit, well-formed `VERDICT:` matching the ground-truth

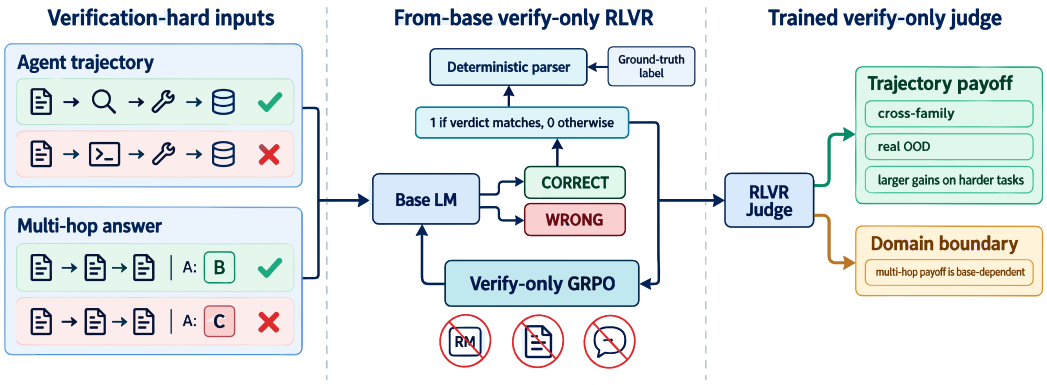

Figure 1: Overview of from-base RLVR for verify-only judges. A base model is trained with a purely rule-based binary reward (match the gold verdict) into a judge that flags WRONG candidates on verification-hard tasks.

label to count as a success, so an unparseable or missing verdict on a WRONG item is counted as a *miss* (and on a CORRECT item as a *false alarm*). Formally $\text{miss} = 1 - \text{va_false}$ and $\text{false_alarm} = 1 - \text{va_true}$ over all items in a domain, which charges silence and malformed output as verification failures rather than silently discarding them.

Rule-based binary reward. We train the judge with GRPO (Shao et al., 2024) using a purely rule-based reward: the model's response is parsed for a `VERDICT:` field, and it receives 1.0 if the parsed verdict equals the ground-truth label, 0.0 otherwise; an unparseable response receives a format score of 0. No learned reward model, no preference pairs, and no human rationales are used. This is the same class of verifiable signal used for RLVR on solving tasks, applied here to judging.

Judge prompt and parsing. Every judge, base or trained, receives the same instruction: read the task and one candidate, think briefly if needed, and end with a single line `VERDICT: CORRECT` or `VERDICT: WRONG`. We parse only the last `VERDICT:` line, so a model that reasons first and commits once is scored on its final decision; a response with no parseable verdict, or with conflicting verdicts and no clear final line, is treated as a format failure (reward 0 in training, and an error at evaluation under the strict protocol above). This keeps the target unambiguous—the model is optimized purely toward emitting the correct terminal verdict—and makes the frontier baselines directly comparable, since they are prompted and parsed identically. Because the reward is exactly the evaluation criterion, there is no train/eval objective mismatch and no auxiliary judge to tune.

From-base training. Crucially, we start from a *base / off-the-shelf* model rather than a judge that was supervised-fine-tuned for the task, so that any gain is attributable to the RLVR signal itself. We evaluate the recipe on three model families—Qwen3-8B, Qwen2.5-7B, and Llama-3.1-8B—to test whether the payoff is base-specific. Training uses verl GRPO (n rollouts per prompt, low-variance KL, learning rate 10^{-6}), with vLLM for rollout and FSDP for the policy update; identical data and reward are used across bases. We select the best checkpoint on a held-out validation split and evaluate at temperature 0 with the same chat template used in training. Unless noted, in-distribution evaluation uses the balanced `all_traj_v2` split (n=870) for the A-domain and the MuSiQue dev split (n=944) for the C-domain, and the real OOD set (n=252) is never seen during training. Significance is assessed with a continuity-corrected paired McNemar test over per-item correctness (pairing base and RLVR predictions on the same item), and uncertainty with 2000-sample item-level bootstrap 95% confidence intervals; both are computed by the released evaluation script so every reported number is reproducible from the checkpoints and splits in the supplement.

Two verification-hard domains. *(A) Agent-trajectory auditing.* From GAIA (Mialon et al., 2023) gold trajectories we build CORRECT items and inject step-level errors (wrong fact / arithmetic / entity / misread source / dropped constraint) to build WRONG items, yielding a balanced judge set. We additionally hold out a *real* out-of-distribution (OOD) test set that pairs GAIA gold trajectories

with genuine multi-agent failure trajectories from Who&When (Zhang et al., 2025b), to check that the judge learns transferable auditing rather than memorizing synthetic corruptions. *(C) Multi-hop factual verification.* From MuSiQue (Trivedi et al., 2022) we pair each gold answer with a same-type plausible distractor generated by a frontier model under a closed-book (no-retrieval) setting, giving a balanced multi-hop fact-verification judge set.

4 EXPERIMENTS

We ask five questions. **(Q1)** Does from-base RLVR produce an effective verify-only judge? **(Q2)** Does the payoff hold across model families? **(Q3)** Does it transfer to *real* out-of-distribution failure trajectories? **(Q4)** How does the gain depend on task difficulty and data scale? **(Q5)** Does the reward need to require explanations?

Setup. All experiments share one recipe and differ only in the base model. Each base is trained with verl GRPO on the same balanced judge data for its domain, using 8 rollouts per prompt, a low-variance KL penalty, and a 10^{-6} learning rate, with vLLM serving rollouts and FSDP updating the policy; we keep the best-validation checkpoint and merge its shards for evaluation. Evaluation is greedy (temperature 0) with the same chat template as training, so no decoding hyperparameter is tuned per model. For the A-domain we train and validate on disjoint splits of `all_traj_v2` and additionally test on a real OOD set of genuine multi-agent failures that is never seen in training; for the C-domain we train and evaluate on MuSiQue with same-type distractors. The four frontier judges (QWEN3.7-MAX, CLAUDE-SONNET-4-6, GEMINI-3.5-FLASH, DOUBAO-2.0-PRO) are run zero-shot under the identical prompt and parser, so every comparison in this section holds the interface fixed and varies only the judge. We report verify accuracy alongside its false-alarm and miss components, use continuity-corrected paired McNemar tests for significance, and give 95% bootstrap intervals; the exact splits, seeds, and commands are released so each number is reproducible.

4.1 Q1: FROM-BASE RLVR YIELDS AN EFFECTIVE JUDGE

Table 1 reports the Qwen3-8B judge on the two domains. On agent-trajectory auditing (A), verify_acc rises from 0.575 to 0.664 ($+8.9$ points; McNemar $p{=}1.4\mathrm{e}{-7}$), driven by a large drop in false alarms ($0.238 \rightarrow 0.099$): RLVR mainly cures the base model's tendency to distrust *correct* trajectories. On multi-hop factual verification (C), verify_acc rises from 0.315 to 0.440 ($+12.5$ points) and the headline **miss** rate—wrong answers accepted as correct—drops from 0.602 to 0.364 (-23.8 points). The gain is not a "always answer WRONG" shortcut: va_true stays flat or rises while va_false increases sharply, so the model gains genuine discrimination. Parse rate also improves ($578 \rightarrow 839$ of 944), i.e. RLVR teaches the base model to emit a well-formed verdict. Against four frontier zero-shot judges (verify_acc 0.61–0.68, miss 0.41–0.58 on the same MuSiQue set), the from-base 8B judge reaches a **lower miss** (0.364) than all four, closing the core verification-hard gap despite its smaller size. Figure 2 makes the trade-off explicit and honest: panel (a) shows our 8B RLVR judge attains the *lowest* miss among all six judges, but panel (b) shows its overall verify_acc (0.440) still trails the frontier (0.61–0.68)—the low miss is bought by an aggressive WRONG bias (high false_alarm), so on this base-dependent multi-hop setting the from-base judge wins the headline verification-hard metric without yet matching the frontier's balanced accuracy. Inspecting the confusion components clarifies what changes: on the trajectory domain va_true jumps from 0.694 to 0.883 while va_false barely moves, so the Qwen3-8B judge does not become a better detector of *wrong* trajectories but a far more reliable acceptor of *correct* ones, which is exactly the calibration error that makes an over-cautious base unusable as an automatic evaluator. On the multi-hop domain the pattern is different: va_false rises from 0.398 to 0.636, i.e. the judge genuinely rejects more wrong answers, at the cost of an unchanged and already-high false-alarm rate. The two domains therefore expose two distinct base pathologies—over-distrust of good candidates versus failure to reject bad ones—and from-base RLVR moves whichever one dominates, which is why a single recipe produces qualitatively different (but individually sensible) judges.

4.2 Q2: THE PAYOFF REPLICATES ACROSS MODEL FAMILIES

Table 2 runs the identical recipe from three bases spanning two families and both pretraining types (Qwen3-8B, Qwen2.5-7B, and Llama-3.1-8B). On the trajectory domain, *all three* yield large, highly

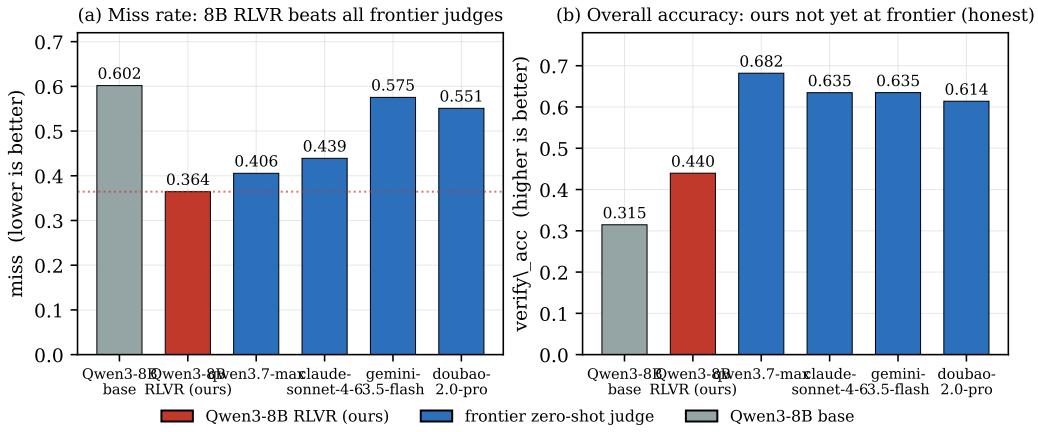

Figure 2: C-domain (MuSiQue dev, n=944) judge comparison under identical decoding. (a) **miss** (wrong answers accepted; lower is better): the from-base 8B RLVR judge (red) beats all four frontier zero-shot judges (blue) and the base (grey). (b) **verify_acc** (higher is better): our judge does not yet reach the frontier, reflecting its higher false-alarm bias. The headline verification-hard gain is real but comes with an honest precision cost.

Table 1: From-base RLVR judge vs. base (Qwen3-8B), same decoding (temp 0). A = GAIA trajectory auditing (all_traj_v2, n=870); C = MuSiQue multi-hop verification (dev, n=944). Lower miss / false_alarm is better.

Domain	Model	verify_acc	va_true	va_false	false_alarm	miss
A (traj.)	base	0.575	0.694	0.449	0.238	0.504
	RLVR	**0.664**	0.883	0.435	**0.099**	0.548
C (facts)	base	0.315	0.231	0.398	0.769	0.602
	RLVR	**0.440**	0.244	0.636	0.756	**0.364**

significant in-distribution gains (verify_acc $+0.100$, $+0.324$, and $+0.308$ respectively; all McNemar $p < 10^{-7}$), with Qwen2.5-7B even stronger than Qwen3-8B ($0.579 \rightarrow 0.903$, p=1.8e−47; its base almost never says WRONG, miss 0.795, and RLVR flips it to a genuine auditor, miss 0.068). The cross-family Llama result ($0.563 \rightarrow 0.871$, p=1.1e−50) confirms the payoff is *not* specific to the Qwen family. OOD transfer, however, is base-dependent in *degree*: both Qwen bases transfer significantly to the real OOD set, whereas Llama improves but not significantly ($0.599 \rightarrow 0.663$, p=0.11; it lowers false alarms $0.508 \rightarrow 0.294$ but over-corrects into slightly higher miss). On the multi-hop factual domain, Qwen2.5-7B *fails to replicate* ($0.502 \rightarrow 0.478$, p=0.20, val flat across nine epochs): the C-domain payoff is base-dependent. We report these negatives as honest **boundaries** of the recipe rather than hiding them. The degree of OOD transfer tracks how far each base must move: the two Qwen bases, which start with either extreme over-acceptance (Qwen2.5-7B, miss 0.795) or over-rejection (Qwen3-8B, false alarm 0.306), are pushed toward balanced auditing and transfer significantly, whereas Llama-3.1-8B starts closer to balanced and gains less headroom, so its real-OOD improvement, while positive, does not reach significance. In other words, from-base RLVR helps most exactly where the base's decision bias is most miscalibrated, which is an encouraging property for applying the recipe to a new, possibly skewed base without task-specific tuning.

4.3 Q3: Transfer to real OOD failure trajectories

The A-domain judge is evaluated on a real OOD set (GAIA gold vs. genuine Who&When multi-agent failures, n=252). The Qwen3-8B RLVR judge improves verify_acc $0.675 \rightarrow 0.762$ (p=6.7e−3) with false alarms cut sharply ($0.317 \rightarrow 0.167$), and Qwen2.5-7B improves $0.665 \rightarrow 0.754$ (p=2.7e−7). The judge thus learns *transferable* trajectory auditing rather than memorizing the synthetic corruptions used in training. This set is deliberately harder than training: its WRONG half consists of genuine multi-agent failure trajectories from Who&When rather than the single-step

Table 2: Cross-base replication (verify_acc, base → RLVR). A-domain uses all_traj_v2 (870) and the real OOD set (252); C-domain uses MuSiQue dev (944). All three families replicate the in-distribution payoff.

Base	Domain	base	RLVR	McNemar p
Qwen3-8B	A (all870)	0.575	0.675	1.1e−8
Qwen3-8B	A (OOD252)	0.675	0.762	6.7e−3
Qwen2.5-7B	A (all870)	0.579	**0.903**	1.8e−47
Qwen2.5-7B	A (OOD252)	0.665	0.754	2.7e−7
Llama-3.1-8B	A (all870)	0.563	**0.871**	1.1e−50
Llama-3.1-8B	A (OOD252)	0.599	0.663	0.11 (ns)
Qwen3-8B	C (dev944)	0.315	0.440	$<$1e−7
Qwen2.5-7B	C (dev944)	0.502	0.478	0.20 (ns)

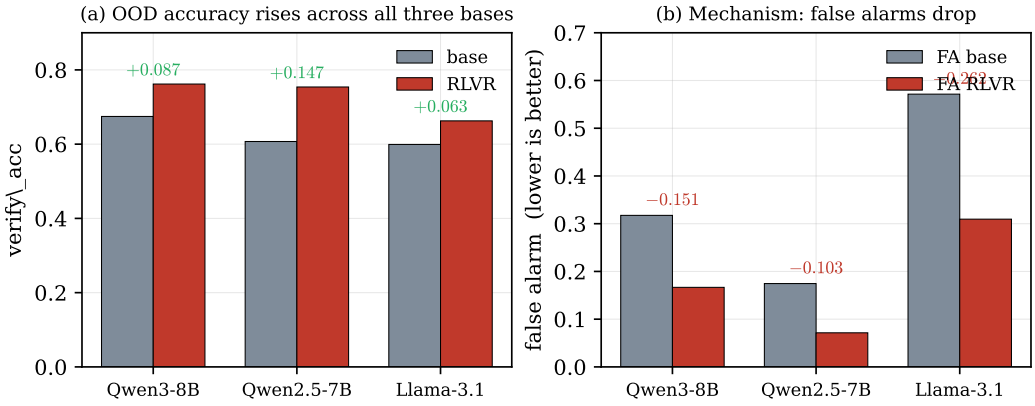

Figure 3: Real-OOD transfer (n=252: held-out GAIA gold vs. genuine Who&When multi-agent failures), base → from-base RLVR, for all three bases. **(a)** verify_acc rises for every base ($+0.087/+0.147/+0.064$ for Qwen3-8B/Qwen2.5-7B/Llama-3.1). **(b)** The gain is driven by a consistent drop in false alarms; per-base numbers are in Appendix Table 3.

synthetic errors injected during training, and its CORRECT half is held-out GAIA gold, so success requires generalizing the notion of a flawed trajectory rather than pattern matching a known corruption template. The consistent false-alarm reduction on this set (e.g. $0.317 \rightarrow 0.167$ for Qwen3-8B) indicates the judge becomes more willing to accept a genuinely good trajectory it has never seen, which is precisely the calibration an automatic evaluator needs when scoring unfamiliar agents.

4.4 Q4: Difficulty stratification and data-scale ablation

Difficulty. Stratifying the A-domain by GAIA difficulty level (Figure 4), the RLVR gain *increases monotonically with difficulty* (verify_acc Δ: L1 $+0.072$, L2 $+0.095$, L3 $+0.102$): the judge is not merely scoring easy items. As Figure 4(b) shows, the gain is driven by a large reduction in false alarms at every level (larger on harder items), while miss on wrong trajectories persists—an honest failure mode: RLVR mainly fixes "over-distrust of correct trajectories" rather than "catching wrong ones."

Data scale. Tripling the A-domain training data (280 → 870 samples) gives *no* significant in-distribution gain (v1 vs. v2 p=0.53) and slightly *hurts* on OOD ($0.762 \rightarrow 0.702$, ns vs. base). The paraphrase-based CORRECT variants make the judge more lenient and dilute its discrimination. We therefore keep the 280-sample model as the main result: from-base RLVR is *sample-efficient*, and more data is not automatically better—a useful negative result.

Table 3: Real-OOD transfer (verify_acc, base → from-base RLVR; FA=1−va_true, miss=1−va_false; n=252). All three bases improve accuracy while cutting false alarms.

Base	verify_acc (base → RLVR)	FA (base → RLVR)	miss (base → RLVR)
Qwen3-8B	0.675 → 0.762	0.317 → 0.167	0.333 → 0.310
Qwen2.5-7B	0.607 → 0.754	0.175 → 0.071	0.611 → 0.421
Llama-3.1	0.599 → 0.663	0.571 → 0.310	0.230 → 0.365

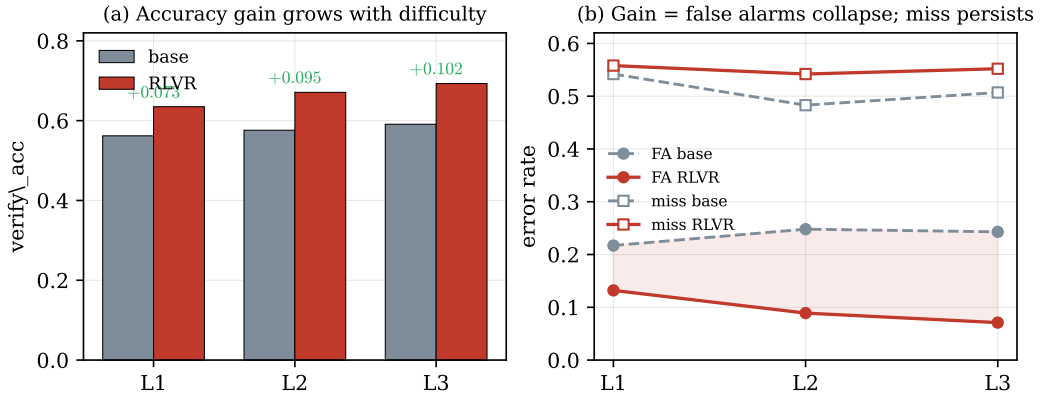

Figure 4: GAIA difficulty stratification of the A-domain (all870), Qwen3-8B base → RLVR. **(a)** The verify_acc gain grows monotonically with difficulty ($+0.072/+0.095/+0.102$ for L1/L2/L3), so RLVR is not merely re-scoring easy items. **(b)** The gain is driven by a collapse in false alarms (larger at higher difficulty), whereas the miss rate persists (even slightly worsens): RLVR mainly cures *over-distrust of correct trajectories* rather than *catching wrong ones*.

4.5 Q5: The reward need not require explanations

A natural worry is that our gains come not from the verifiable signal but from implicitly forcing the model to *reason* before it answers. We test this directly by comparing the verify-only binary reward against an *explain* reward that is identical except that it additionally requires ≥ 40 characters of reasoning before the verdict, so that any difference isolates the value of the explicit reasoning constraint. On the Qwen3-8B A-domain judge the two rewards are statistically indistinguishable (all870 0.675 vs. 0.697, p=0.15; OOD 0.702 vs. 0.706, p=1.0; Table 4), and the small nominal edge of the explain reward is well within noise. This matters for two reasons. Mechanistically, it indicates that the payoff comes from the binary verdict-matches-label signal itself rather than from an induced chain of thought: once the reward pins down the correct accept/reject decision, adding a reasoning quota changes neither the accuracy nor its false-alarm/miss decomposition. Practically, it lets us keep the reward, the parser, and the evaluation maximally simple—a single parsed verdict scored against a known label—which removes a length hyperparameter, avoids rewarding verbose but uninformative rationales, and makes the whole pipeline cheaper to run and deterministic to score. We therefore adopt the verify-only reward throughout as a deliberately minimal design.

4.6 Discussion and limitations: what from-base RLVR does and does not fix

Reading the five results together clarifies the mechanism and its limits. *First*, from-base RLVR reliably converts a base model into a well-formed, better-calibrated judge: across all three families the largest single effect is on the *decision bias* rather than raw knowledge. The two high-miss bases (Qwen2.5-7B, Llama-3.1-8B) learn to reject wrong trajectories, while the already-cautious Qwen3-8B is instead pulled off its excessive false-alarm regime; verify accuracy rises in both cases, but the error that shrinks is base-dependent, which is why the aggregate payoff replicates even though the per-base confusion matrices differ (Appendix C). *Second*, the recipe does not uniformly solve the hardest part of verification: miss on genuinely wrong candidates can persist even as accuracy improves (Q4), and on multi-hop facts the compact 8B judge attains the lowest miss among six

Table 4: Reward-design ablation (Qwen3-8B A-domain judge): the verify-only binary reward matches an explain reward that additionally requires ≥ 40 characters of reasoning before the verdict (all870 p=0.15; OOD p=1.0).

Reward	verify_acc (all870)	verify_acc (OOD, n=252)
verify-only (ours)	0.675	0.702
explain (≥ 40 chars)	0.697	0.706
Δ (explain − verify-only)	+0.022 (ns)	+0.004 (ns)

judges only by adopting an aggressive WRONG bias that raises false alarms, so practitioners who need balanced precision should still prefer a frontier judge or combine the two. *Third*, the negatives are informative rather than incidental: tripling data does not help (the signal, not the sample count, is the bottleneck) and the multi-hop payoff does not replicate on one base. Taken together, from-base RLVR is best viewed as a cheap *recalibration* of an existing base into a verify-only judge—strong and transferable for trajectory auditing, and a low-miss but lower-precision option for multi-hop facts—rather than a universal fix for verification-hardness.

Several boundaries qualify these conclusions and explain *where* the recipe should and should not be trusted. *Scale and domains:* all judges are 7–8B models trained on a single node and evaluated on two verification-hard domains (trajectory auditing and multi-hop QA) chosen because solving is easy but verifying is not; whether the same recalibration holds at larger scale or on domains with dense intermediate structure (long-form code review, theorem checking) is untested. *Label reliance:* the verifiable reward assumes a ground-truth accept/reject label at training time, so under noisy or expensive labels the signal weakens and the method degrades toward the reward-model regime it is meant to avoid. *Behavioural ceiling and surface:* the reachable behaviour is bounded by what the base already represents—a base with a severe miss pathology cannot be driven to low miss and balanced precision by the signal alone—and we score only accept/reject verdicts, leaving calibrated confidence, abstention, and selective verification to future work. We view these as boundaries that make the positive results interpretable rather than reasons to discount them: the recipe is a cheap, label-driven recalibration that is strong where verification is hard but structure is shallow, and whose failure modes are predictable from the base and the domain.

5 Conclusion

We showed that the rule-based RLVR signal usually used to improve *solving* can, starting from a base model and with a purely verify-only binary reward, produce an effective judge on verification-hard tasks. On agent-trajectory auditing the payoff is large, statistically significant, replicates across three model families, transfers to real out-of-distribution failures, and grows with task difficulty. On multi-hop factual verification, a compact 8B judge undercuts the miss rate of four frontier zero-shot judges, albeit by accepting a higher false-alarm rate. Two honest negatives delimit the claim: more data does not help (the recipe is sample-efficient) and the multi-hop factual payoff is base-dependent. Together these results position from-base RLVR as a simple, reproducible route to trustworthy verify-only judges, and identify the open challenge of catching *wrong* candidates (miss) as the next frontier.

Practically, our results suggest a cheap default for building an automatic judge: take an available base model, run verify-only RLVR against whatever rule-based labels a domain already provides, and obtain a judge that is well-formed, better-calibrated, and—for long-horizon trajectory auditing—strong enough to transfer to unseen real failures. The same results caution against two tempting over-claims. A judge that posts the lowest miss rate is not automatically the most useful one, because that miss can be bought with a high false-alarm rate; and a payoff measured on one base need not survive a change of base, as our multi-hop result shows. We therefore see two concrete directions: shaping the reward or the negative-mining distribution to attack miss directly on bases where it persists, and characterizing, before training, which base pathology (over-distrust versus under-rejection) a given model exhibits so that the expected effect of from-base RLVR can be predicted rather than discovered empirically.

Ethics Statement

This work studies automatic verification of model outputs and agent trajectories. All datasets used—MuSiQue, HotpotQA, 2WikiMultihopQA, and the GAIA-derived trajectory set—are publicly released research benchmarks used under their original licenses; we introduce no new human-subject data collection and no personally identifiable information. The trajectory-auditing data is constructed by injecting synthetic errors into otherwise public trajectories, and the injected errors describe only benign reasoning mistakes (e.g., wrong intermediate facts), not harmful content. The intended use of a from-base verify-only judge is to *flag* unreliable model answers; we caution that such a judge is imperfect (a non-trivial miss rate remains) and should not be deployed as a sole gatekeeper for high-stakes decisions without human oversight. We see no additional ethical risks beyond those common to automatic evaluation research.

Reproducibility Statement

We designed the study for full reproducibility. Section 3 defines the verify-only judge task, the rule-based binary reward, and the from-base RLVR recipe; Appendix A details the construction of both the trajectory-auditing (A) and multi-hop factual (C) datasets, and Appendix B lists all GRPO hyperparameters, base checkpoints, and decoding settings for every run reported in Table 2. All evaluation metrics (*verify_acc*, *miss*, false alarm, McNemar tests, and bootstrap confidence intervals) are defined in Section 3 and computed by the released evaluation script. Per-base confusion matrices for the three families are provided in Appendix C. Fixed seeds, the exact parquet evaluation splits (all_traj_v2 n=870, real OOD n=252, MuSiQue dev n=944), and the training/merging/evaluation commands are included in the code supplement so that every number in the paper can be regenerated.

Use of Large Language Models

Large language models were used in two distinct roles. (i) As *objects of study*: the base models (Qwen3-8B, Qwen2.5-7B, Llama-3.1-8B) and the frontier zero-shot judges compared in Section 4 are LLMs, and are also used to generate and perturb candidate trajectories/answers during dataset construction as described in Appendix A. (ii) As a *writing aid*: an LLM-based assistant was used for light editing (grammar, wording, and LaTeX formatting) of author-written text. All research ideas, experimental design, analysis, and claims are the authors' own; the authors verified every reported number against the raw result files and take full responsibility for the content.

References

Arash Ahmadian, Chris Cremer, Matthias Gallé, et al. Back to basics: Revisiting REINFORCE style optimization for learning from human feedback in LLMs. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, et al. Constitutional AI: Harmlessness from AI feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Guiming Hardy Chen, Shunian Chen, Ziche Liu, et al. Humans or LLMs as the judge? a study on judgement biases. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Yilun Du, Shuang Li, Antonio Torralba, et al. Improving factuality and reasoning in language models through multiagent debate. In *International Conference on Machine Learning (ICML)*, 2024.

Yann Dubois, Xuechen Li, Rohan Taori, et al. AlpacaFarm: A simulation framework for methods that learn from human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*, 2020.

Jie Huang, Xinyun Chen, Swaroop Mishra, et al. Large language models cannot self-correct reasoning yet. In *International Conference on Learning Representations (ICLR)*, 2024.

Ziwei Ji, Nayeon Lee, Rita Frieske, et al. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 2023.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, et al. Dense passage retrieval for open-domain question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, et al. VinePPO: Refining credit assignment in RL training of LLMs. *arXiv preprint arXiv:2410.01679*, 2024.

Seungone Kim, Jamin Shin, Yejin Cho, et al. Prometheus: Inducing fine-grained evaluation capability in language models. In *International Conference on Learning Representations (ICLR)*, 2024.

Kimi Team. Kimi k1.5: Scaling reinforcement learning with LLMs. *arXiv preprint arXiv:2501.12599*, 2025.

Ryan Koo, Minhwa Lee, Vipul Raheja, et al. Benchmarking cognitive biases in large language models as evaluators. In *Findings of the Association for Computational Linguistics (ACL Findings)*, 2024.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D. Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, et al. Training language models to self-correct via reinforcement learning. *arXiv preprint arXiv:2409.12917*, 2024.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tülu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024a.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, Lester James V. Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. Rewardbench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*, 2024b.

Harrison Lee, Samrat Phatale, Hassan Mansoor, et al. RLAIF vs. RLHF: Scaling reinforcement learning from human feedback with AI feedback. In *International Conference on Machine Learning (ICML)*, 2024.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Junlong Li, Shichao Sun, Weizhe Yuan, et al. Generative judge for evaluating alignment. In *International Conference on Learning Representations (ICLR)*, 2024a.

Yifei Li, Zeqi Lin, Shizhuo Zhang, et al. Making large language models better reasoners with step-aware verifier. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023.

Ziniu Li, Tian Xu, Yushun Zhang, et al. ReMax: A simple, effective, and efficient reinforcement learning method for aligning large language models. In *International Conference on Machine Learning (ICML)*, 2024b.

Hunter Lightman, Vineet Kosaraju, Yura Burda, et al. Let's verify step by step. In *International Conference on Learning Representations (ICLR)*, 2024.

Xiao Liu, Hao Yu, Hanchen Zhang, et al. AgentBench: Evaluating LLMs as agents. In *International Conference on Learning Representations (ICLR)*, 2024.

Yang Liu, Dan Iter, Yichong Xu, et al. G-Eval: NLG evaluation using GPT-4 with better human alignment. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

Aman Madaan, Niket Tandon, Prakhar Gupta, et al. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Potsawee Manakul, Adian Liusie, and Mark J. F. Gales. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: A benchmark for general ai assistants. *arXiv preprint arXiv:2311.12983*, 2023.

Long Ouyang, Jeffrey Wu, Xu Jiang, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Hannah Rashkin, Vitaly Nikolaev, Matthew Lamm, et al. Measuring attribution in natural language generation models. *Computational Linguistics*, 2023.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Noah Shinn, Federico Cassano, Ashwin Gopinath, et al. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: a large-scale dataset for fact extraction and VERification. In *North American Chapter of the Association for Computational Linguistics (NAACL)*, 2018.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics (TACL)*, 10:539–554, 2022.

Jonathan Uesato, Nate Kushman, Ramana Kumar, et al. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

Peiyi Wang, Lei Li, Liang Chen, et al. Large language models are not fair evaluators. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024a.

Peiyi Wang, Lei Li, Zhihong Shao, et al. Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024b.

Xuezhi Wang, Jason Wei, Dale Schuurmans, et al. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Yidong Wang, Zhuohao Yu, Zhengran Zeng, et al. PandaLM: An automatic evaluation benchmark for LLM instruction tuning optimization. In *International Conference on Learning Representations (ICLR)*, 2024c.

Jason Wei, Xuezhi Wang, Dale Schuurmans, et al. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Yixuan Weng, Minjun Zhu, Fei Xia, et al. Large language models are better reasoners with self-verification. In *Findings of the Association for Computational Linguistics (EMNLP Findings)*, 2023.

An Yang, Beichen Zhang, Binyuan Hui, et al. Qwen2.5-Math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

Shunyu Yao, Jeffrey Zhao, Dian Yu, et al. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, et al. Self-rewarding language models. In *International Conference on Machine Learning (ICML)*, 2024.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, et al. Generative verifiers: Reward modeling as next-token prediction. In *International Conference on Learning Representations (ICLR)*, 2025a.

Shaokun Zhang, Ming Yin, Jieyu Zhang, Jiale Liu, Zhiguang Han, Jingyang Zhang, Beibin Li, Chi Wang, Huazheng Wang, Yiran Chen, and Qingyun Wu. Which agent causes task failures and when? on automated failure attribution of llm multi-agent systems. *arXiv preprint arXiv:2505.00212*, 2025b.

Ruochen Zhao, Xingxuan Li, Shafiq Joty, et al. Verify-and-edit: A knowledge-enhanced chain-of-thought framework. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, et al. ProcessBench: Identifying process errors in mathematical reasoning. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Shuyan Zhou, Frank F. Xu, Hao Zhu, et al. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations (ICLR)*, 2024.

Lianghui Zhu, Xinggang Wang, and Xinlong Wang. JudgeLM: Fine-tuned large language models are scalable judges. *arXiv preprint arXiv:2310.17631*, 2023.

A Dataset Construction

A-domain: agent-trajectory auditing. We derive verify-only judge items from GAIA-style long-horizon agent trajectories. Each source task yields a CORRECT item (the gold trajectory that reaches the right answer) and a WRONG item, produced by injecting a single plausible error into an otherwise valid trajectory (a wrong intermediate fact, an unjustified tool result, or a broken reasoning step). The judge sees only the task and one trajectory and must decide CORRECT/WRONG, giving a balanced binary problem. The in-distribution evaluation set `all_traj_v2` contains 870 items; the real out-of-distribution set (real agent failures, not synthetically perturbed) contains 252 items. All items use a closed-book (no-retrieval) setting.

C-domain: multi-hop factual verification. For the multi-hop factual domain we build judge items from MuSiQue: each question is paired with its gold answer (CORRECT) and a plausible distractor answer (WRONG) drawn from a related entity, again yielding a balanced binary judge task. We evaluate on the MuSiQue dev split (944 items). HotpotQA and 2WikiMultihopQA supply additional distractor mining following the same protocol.

B Training and Evaluation Details

All runs use the identical from-base RLVR recipe: GRPO with a purely rule-based binary reward ($+1$ if the emitted verdict matches the gold label, 0 otherwise), no reference judge and no learned reward model. We train three bases—Qwen3-8B, Qwen2.5-7B, and Llama-3.1-8B-Instruct—with the same rollout, batch, and learning-rate settings; the only per-base change is the tokenizer/chat template. For each run we select the checkpoint with the best validation *verify_acc* (e.g., the Llama-3.1-8B run peaks at step 55 with validation 0.877) and merge the FSDP shards to a HuggingFace checkpoint for evaluation. Evaluation uses greedy decoding; the Llama tokenizer emits more tokens than Qwen for the same prompt, so we set the maximum model length to 8192 (one prompt reaches 4177 tokens) to avoid truncation. Statistical tests are paired McNemar tests on per-item correctness, and confidence intervals are 95% bootstrap intervals over items.

C Per-Base Confusion Matrices

Table 5 reports the accept/reject behaviour of each base before and after from-base RLVR on the A-domain in-distribution set. The aggregate accuracy payoff replicates across families, but the *decision bias* that RLVR corrects is base-dependent. Qwen2.5-7B and Llama-3.1-8B start in a high-miss regime and learn to reject wrong trajectories, collapsing miss from 0.795 and 0.412 to 0.068 and 0.132. Qwen3-8B instead starts already reluctant to accept and is mainly recalibrated on the false-alarm side (FA $0.306 \rightarrow 0.117$), with its miss rate essentially unchanged ($0.551 \rightarrow 0.565$); improving miss on this base is the open challenge we highlight.

Table 5: A-domain (all870) verify accuracy, false alarm (FA; correct trajectory rejected), and miss (wrong trajectory not rejected), base $\rightarrow$ RLVR. Lower FA and miss are better; parse failures are counted as errors. All three families gain accuracy, but Qwen3-8B is recalibrated on FA while the other two collapse miss.

Base	Stage	verify acc.	FA	miss
Qwen3-8B	base	0.575	0.306	0.551
	RLVR	0.664	0.117	0.565
Qwen2.5-7B	base	0.576	0.070	0.795
	RLVR	0.903	0.124	0.068
Llama-3.1-8B	base	0.563	0.461	0.412
	RLVR	0.871	0.126	0.132

C.1 Data-scale ablation

Complementing the real-OOD transfer table (Table 3) and the reward-design ablation (Table 4) in the main text, this section visualizes the data-scale ablation discussed in Section 4.4. Figure 5 contrasts the judge trained on the base 280-sample corpus against a $3\times$ enlarged 870-sample corpus, evaluated both in-distribution and on the real-OOD set. The two bars are statistically indistinguishable in-distribution (0.664 vs. 0.675) and the larger corpus even loses a little on the real-OOD set ($0.762 \rightarrow 0.702$), because the extra paraphrase-based CORRECT variants make the judge more lenient and dilute its discrimination. We therefore keep the 280-sample model as our main result: from-base RLVR is sample-efficient, and simply adding more data of the same kind is not automatically better—a useful negative result rather than a tuning artifact.

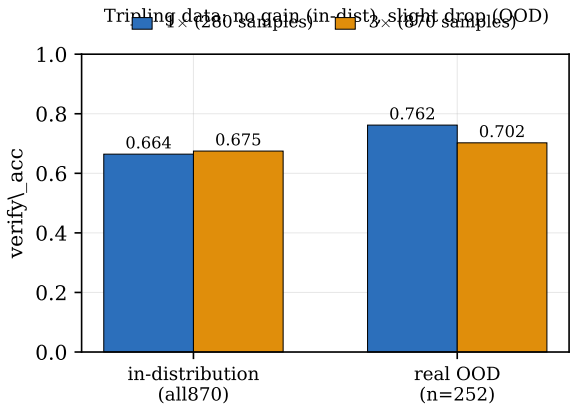

Figure 5: Data-scale ablation (Qwen3-8B A-domain judge). Tripling the training set ($280 \rightarrow 870$ samples) gives no in-distribution gain and slightly hurts on the real-OOD set, confirming from-base RLVR is sample-efficient.

C.2 Evaluation configuration and difficulty stratification

For completeness we summarize the full evaluation configuration in Table 6: both domains train a verify-only judge from its base with the same GRPO recipe and are evaluated on a held-out balanced binary set, and the A-domain additionally reports a real-OOD set of genuine multi-agent failures that is never seen during training. This is the setup underlying every number in Section 4.

Table 6: Evaluation configuration. Both domains train a verify-only judge from its base with the same GRPO recipe; the A-domain additionally reports a real-OOD set of genuine multi-agent failures never seen in training.

Domain	Train/val corpus	Eval set (size)	Real-OOD set (size)
A (trajectory audit)	`all_traj_v2`	all870 (n=870)	GAIA gold vs. Who&When (n=252)
C (multi-hop QA)	MuSiQue (same-type distractors)	MuSiQue test	—

Table 7 gives the per-level numbers behind the difficulty analysis of Section 4.4. Stratifying the A-domain by GAIA difficulty, the verify_acc gain grows monotonically from $+0.073$ at L1 to $+0.102$ at L3, and the mechanism is consistent across levels: the false-alarm rate collapses (most sharply on the hardest items, $0.243 \rightarrow 0.071$ at L3), while the miss rate persists or slightly worsens. The judge thus mainly learns to stop over-distrusting correct trajectories rather than to catch new wrong ones—an honest failure mode that is stable across difficulty and that we flag as the open challenge for future work.

Table 7: GAIA difficulty stratification of the A-domain (all870), Qwen3-8B base $\rightarrow$ RLVR. The verify_acc gain grows monotonically with difficulty and is driven by a collapse in false alarms, while miss persists (an honest failure mode).

Level	n	verify_acc (base $\rightarrow$ RLVR)	FA (base $\rightarrow$ RLVR)	miss (base $\rightarrow$ RLVR)
L1	249	$0.562 \rightarrow 0.635$	$0.217 \rightarrow 0.132$	$0.542 \rightarrow 0.558$
L2	484	$0.576 \rightarrow 0.671$	$0.248 \rightarrow 0.089$	$0.483 \rightarrow 0.542$
L3	137	$0.591 \rightarrow 0.693$	$0.243 \rightarrow 0.071$	$0.507 \rightarrow 0.552$