THE REAL BOTTLENECK IS NOT REWARD NOISE: GRADIENT SIGNAL DEGENERACY IN LONG-HORIZON AGENT RL

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning with verifiable rewards depends on an automatic verifier, and on single-turn tasks a considerable amount of verifier error has proven tolerable: noise rates of up to 15 percent leave peak accuracy within two points of a clean baseline. However, a long-horizon trajectory receives a single terminal reward while giving a faulty verifier dozens of opportunities to be wrong about it, so that tolerating the same rate per trajectory would demand a per-turn rate an order of magnitude lower, and long-horizon training should on this reasoning be the more fragile of the two. Contrary to that expectation, we measure no accuracy cost in a multi-turn search agent at false-positive rates up to 0.5, across a fourfold range of realized horizons and a sixfold range of training length. In stark contrast to what accuracy reports, however, the fraction of sampled groups contributing no reward-driven gradient rises from 0.57 to 1.00 over that same range, so that a policy whose batch contains no such gradient anywhere remains indistinguishable, on accuracy, from one trained against a clean verifier. Subsequently, we trace the discrepancy to verifier error that attaches to the task rather than to the answer and therefore removes whole groups at once, and we obtain a parameter-free account of the collapse which transfers across two environments, three model families, false negatives and KL regularization. Our work shows that a verifier cannot be certified by the performance of the policy it trains, and that the fraction of groups producing no reward-driven gradient, obtainable at no cost during training, should be monitored in its place.

1 INTRODUCTION

Training an agent by reinforcement learning from verifiable rewards requires a stand-in for correctness. In place of a human judgement, an automatic verifier decides whether a trajectory succeeded, and every gradient the policy receives is mediated by that decision. Such verifiers are imperfect by construction: a test suite admits solutions it was never designed to admit, substring matching accepts an answer that merely contains the reference, and a model-based judge inherits the failure modes of the model behind it. How much verifier error this form of training can absorb is therefore a question about the foundation of the method rather than about one of its hyperparameters.

On single-turn tasks the question has received a reasonably consistent answer. Plesner et al. (2026) inject controlled noise into code and science tasks and report that peak validation accuracy stays within two points of a clean baseline at noise rates up to 15 percent, on which basis they recommend that practitioners prioritize moderate verifier accuracy at high precision over exact verification. Mitsuhashi et al. (2026) ask whether additional compute can be substituted for supervision quality and find that it cannot, although the resulting shortfall is bounded. Zhang (2026) examine the false positives that arise naturally from weak test suites, which recur on a given problem rather than being redrawn independently, and find their average effect on held-out performance to be small even though 47.6 percent of the programs they reward are incorrect. Two features are common to this literature: the tasks are single-turn, and tolerance is assessed through task accuracy.

Neither feature survives the move to agents. A single trajectory spans several to dozens of tool calls while the reward arrives once at the end, so tolerating 15 percent error at the trajectory level would

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

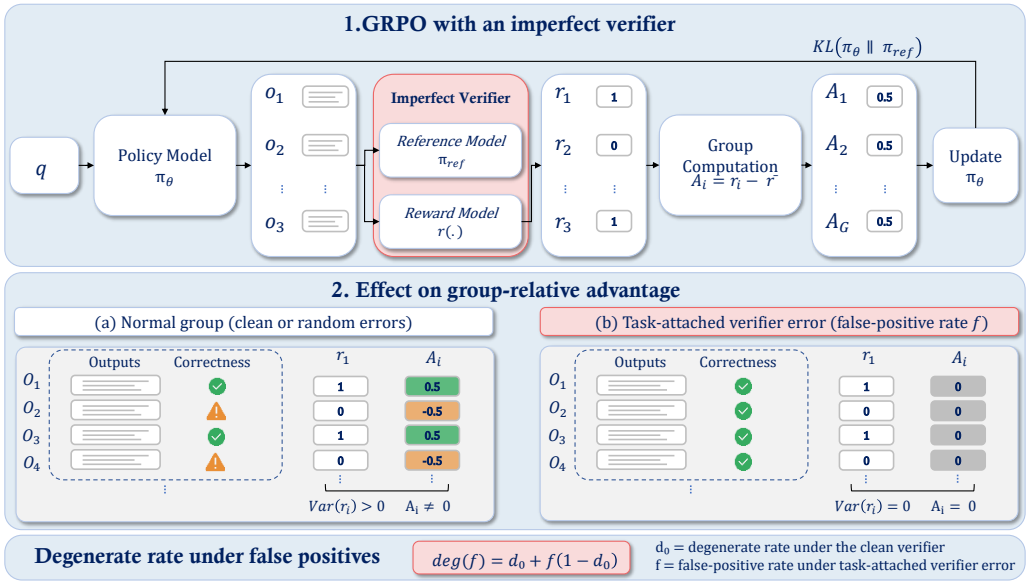


Figure 1: **Task-attached verifier errors collapse group-relative learning signals.** Top: GRPO computes relative advantages from verifier rewards within each group. Bottom: a normal group retains reward variation and nonzero advantages, whereas task-attached false positives can make all rewards identical, yielding zero advantage for every sample. This mechanism implies $\deg(f) = d_0 + f(1 - d_0)$.

require a per-turn error rate near 1 percent at horizon 16, were the turns independent. Whether verifier error in fact compounds this way is the question we take up, and we take it up within group-relative policy optimization, which is what agentic RLVR predominantly uses (Guo et al., 2025; Zhang et al., 2026). A learned value function would require a second model of comparable size trained alongside the policy, and a terminal reward over trajectories of uneven length makes such a function difficult to fit; sampling a verifiable reward several times for one question is inexpensive by comparison, and consequently an advantage taken relative to the remaining samples of that question replaces the critic (Shao et al., 2024; Hu et al., 2025a).

Measured by accuracy, the single-turn conclusion transfers without qualification. We sweep the false-positive rate from 0 to 0.50, the realized number of tool calls from 1.90 to 7.33, and training from 50 to 300 steps, and observe no trend in accuracy beyond seed-to-seed variation. A second quantity recorded in the same runs, however, behaves otherwise. A group whose sampled rewards are identical has zero advantage on each of its samples and therefore contributes nothing to the gradient; the fraction of such groups increases monotonically with the false-positive rate and attains exactly 1.000 on every seed at a rate of 0.50 (Figure 2). Accordingly, at that rate no sample anywhere in the batch carries a reward-driven gradient, and yet its accuracy of 0.547 is that of the clean baseline, 0.542. Tolerance to verifier error, in other words, has been established on a variable that does not move when training stops.

Both measurements follow from a single property of the error, illustrated in Figure 1: in a group-relative estimator, verifier error of the kind that occurs in practice does not degrade the update but removes it, and since such error attaches to the task rather than to the answer, it removes entire groups at once. A group in which every wrong answer is accepted has uniform rewards and drops out, whereas a group left alone contributes exactly what it would have contributed under a clean verifier. Neither outcome moves the update in a wrong direction, which is why the policy survives although its training signal has disappeared. Verifier tolerance is therefore a property of the training signal rather than of the policy, and accuracy will report it most confidently in precisely the regime where no learning occurs.

Our main contributions can be summarized as follows. (i) We show that the degenerate-group rate is a determinate function of the verifier error rate, $d_0 + f(1 - d_0)$ with no fitted quantity, which predicts a held-out condition to 0.002 and deviates from measurement by at most 0.015 across every setting

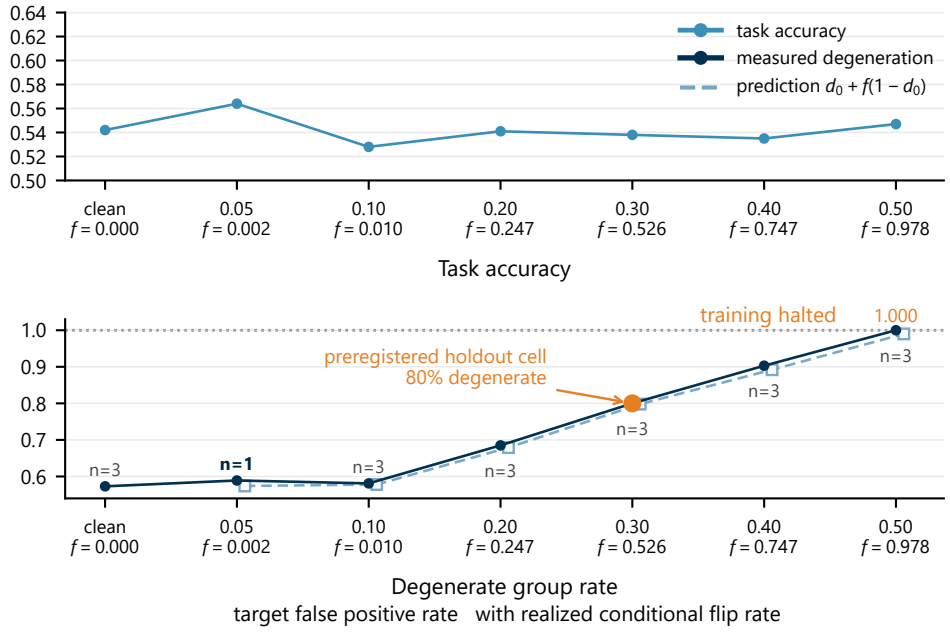


Figure 2: **A verifier can stop training with no sign of it in accuracy.** Horizontal axis: target false-positive rate, with the realized conditional rate f beneath each condition. Top: accuracy under the strict evaluation verifier. Bottom: the degenerate-group rate, with the parameter-free prediction $d_0 + f(1 - d_0)$ dashed; the 0.30 condition is a preregistered holdout. The flat left end reflects a near-zero realized conditional rate, not a threshold.

we test. (ii) We report that verifier error is harmless to the policy because of where it attaches, not how much of it there is: error drawn independently per sample retains usable variation within groups and, unlike task-attached error, costs accuracy. This is the one condition our account overpredicts, by 0.16. (iii) We demonstrate that the null result on accuracy reflects how little was available to lose: where reinforcement learning contributes more than twice as much, the same error rate leaves 29% of the clean gain.

2 Related Work

Verifier quality has recently been treated as a variable rather than a given: systematic verification error produces delay, a plateau, or collapse (Egashira et al., 2026), an incomplete verifier limits what inference-time scaling can deliver (Dorner et al., 2026), and LLM judges drift toward leniency over the course of an evaluation (Mazaheri & Mazaheri, 2026); all of this work, like the three studies above, is single-turn and reports task accuracy. A parallel line accepts the reward as trustworthy and asks how its credit should be distributed over many turns, densifying a terminal reward with temporal-difference signals from a frozen reference model (Tao et al., 2026), clues recovered backward from the reference answer (Lu et al., 2026), bidirectional process rewards (Huang et al., 2026), or evidence anchors supplied as privileged information (Wu et al., 2026); we hold outcome-only GRPO fixed throughout so that this choice cannot enter our conclusions. Group-relative advantages became the default for RLVR following Guo et al. (2025), with leave-one-out and globally normalized variants in the same family (Shao et al., 2024; Hu et al., 2025a), and that choice is now being re-examined for multi-turn settings, where Li et al. (2026) find turn-level PPO the more robust alternative, Liu et al. (2025) reintroduce a small critic, and Dong et al. (2025) move the unit of optimization to the agent step; all three adjudicate between methods by final task performance, which is the quantity we show to be uninformative. A further line treats gameability as a static property of an environment (Bercovich et al., 2026; Zhong et al., 2026; Zhao et al., 2026), whereas our object is the causal effect of verifier error on the trained policy. Closest to us is the work on advantage collapse, where zero-variance groups are filtered by dynamic sampling (Yu et al., 2025), their rate

is shown to predict training stalls (He et al., 2026), the resulting gradient sparsification is analysed (Mishra et al., 2026), and mitigations have followed (Zhang et al., 2025; Agrawal et al., 2026; Deng et al., 2026; Hu et al., 2025b); on the noise side, reward corruption has been modelled as Bernoulli flips with a debiased gradient (Mansouri et al., 2026), a phase boundary at Youden's $J = 0$ has been derived from replicator dynamics (Rad et al., 2026), and capability has been reported to survive the flipping of 40% of rewards (Lv et al., 2025). Two gaps remain. Within-group correlation of verifier error has been measured directly in training rollouts (**?**), but the false-positive rate has not been treated as the independent variable with the collapse rate as the dependent one, and although He et al. (2026) find collapse visible in final accuracy under a clean verifier, a regime exists in which it is not. The mean-field treatment of Rad et al. (2026), moreover, assumes errors drawn independently per sample, whereas errors that attach to a task are correlated within a group and fall outside that framework, which bounds where the analysis applies rather than contradicting it, since converting our rates to theirs places our 0.50 condition at $J \approx 0$, where a stall is likewise predicted.

3 METHOD

3.1 ENVIRONMENT

Our environment is a multi-turn retrieval agent over HotpotQA (Yang et al., 2018) in the distractor setting. Each question comes with ten paragraphs of which two are gold, so that genuine multi-hop retrieval is possible without a Wikipedia-scale index. A tool call is a local BM25 query over one question's candidate paragraphs; because it takes microseconds and touches no network, a matrix over error rates, horizons and seeds becomes affordable. Trajectories follow the Search-R1 format (Jin et al., 2025), with explicit reasoning, search and answer segments. Reasoning-tuned models require two adjustments before they produce this format reliably, since their preambles can exhaust the token budget before any tag appears and their chain of thought reuses one of the format tags. We therefore apply the chat template with the model's thinking mode disabled, and we supply one worked example of the format in the system prompt.

3.2 VERIFIER AND INJECTED ERROR

Training uses the weak verifier that practitioners actually deploy, substring matching against the reference answer. Evaluation uses a strict verifier that extracts the single committed answer, requires an exact match and rejects answers hedged across several candidates; it is never visible during training. The weak verifier produces false positives of its own, as an answer that lists many candidate entities is scored correct whenever the gold string appears among them. These are precisely the naturally occurring, task-attached and asymmetric errors described by Zhang (2026).

On top of them we inject further false positives at a controlled rate. Whether a question is affected is decided by a keyed hash of the question identifier, the epoch and the seed, and does not depend on the answer text; each question is seen once per epoch. Injected error is consequently perfectly correlated within a group: either every wrongly-answered sample of a question is accepted, or none is. This differs from the per-sample independent corruption assumed in single-turn work (Plesner et al., 2026), and it matches the behaviour of weak test suites, which admit the same class of wrong solution for a given problem on every attempt (Zhang, 2026). Section 4.7 isolates the consequences of the difference by adding the answer text to the hash key, which renders the injection independent across samples while holding everything else fixed.

Two rates must be distinguished. Our target is a rate over all judgements, whereas the theoretical literature parameterizes by the conditional rate $f = P(\text{accept} \mid \text{wrong})$; the two are related by $f \approx (\text{target} - \text{natural})/(\text{fraction wrong})$. We report results against the realized f, measured within each run, because the injection budget is reduced by the verifier's own natural false positives so that the total rate remains comparable across horizons. Acceptance follows from a keyed hash rather than a random number generator, and a configuration therefore reproduces exactly.

3.3 CONTROLLING THE HORIZON

The horizon is set at the environment level by requiring a minimum number of retrievals. An answer offered before that count is rejected with a message stating how many searches remain, and

the requirement also appears in the system prompt. Because questions, corpus and answers are unchanged, the parameter moves the horizon without touching task difficulty. One implementation detail matters: a rejected answer must not count as an invalid turn, as the first rejection would otherwise end the episode and leave the requirement without effect. A longer horizon would confound the experiment if it also made evidence harder to obtain. We therefore record the realized number of retrievals together with the fraction of supporting facts covered by retrieved paragraphs, both of which appear in Section 4.1.

3.4 Training

Since credit assignment is the object under study, we implement a minimal GRPO loop rather than adopt an existing framework, which gives us control over which tokens receive credit. Each turn is a separate (prompt, completion) pair, and we train with LoRA (Hu et al., 2021). Log probabilities are computed one sample at a time, without padded batching. The model mixes full and linear attention with state-space and causal-convolution components, and on this architecture padding perturbs a sample's log probability by an amount that grows with sequence length. Since length tracks the number of turns, and the quantity under study is credit assigned across turns, such a perturbation would bias the measurement systematically. Generation is batched, shares one model object with training so that updates take effect immediately, and stops at the closing tag.

3.5 Metrics

The primary metric is accuracy under the strict evaluation verifier, averaged over the final ten steps. Alongside it we record the within-group standard deviation of rewards and, more importantly, the *degenerate-group rate*: the fraction of groups whose sampled rewards are all equal, and for which every sample therefore has zero relative advantage. This is the collapse rate of He et al. (2026) with its threshold set to zero. Such a group contributes nothing through the reward term; since our main sweep disables the reference-policy penalty, it contributes nothing at all in that configuration, whereas under a retained penalty the rate characterises the reward-driven component of the update.

4 Experiments

4.1 The horizon parameter behaves as intended

Raising the required number of retrievals from 0 to 6 raises the realized number of tool calls from 1.90 to 7.33. Over the same range, moreover, the fraction of supporting facts retrieved rises from 0.532 to 0.776, and accuracy under a clean verifier rises from 0.520 to 0.605. Longer horizons therefore give the agent more of the evidence it needs rather than less, and the parameter does not make the task harder (Table 3 in Appendix D, six seeds per row).

4.2 Accuracy shows no cost, at any horizon or training length

In a matrix of three horizons, three error rates and two seeds trained for 50 steps, the accuracy difference between a noisy and a clean verifier is -0.030, -0.022 and -0.020 at a false-positive rate of 0.10 for horizons 0, 3 and 6, and -0.003, -0.011 and -0.038 at a rate of 0.30. Were the error compounding along the trajectory, these differences would grow with horizon; instead they are flat, and their median magnitude of 0.021 matches the within-seed standard deviation of 0.019. At this sample size a difference of 0.05 is detectable with three seeds and one of 0.03 with seven, so that differences below about 0.05 remain possible. Extending the horizon-6 runs from 50 to 300 steps does not change the picture either. Compared window by window, the difference between clean and noisy training is -0.006, $+0.020$, -0.030, $+0.032$ and $+0.029$, which is to say within ± 0.03 throughout and, if anything, positive late in training.

4.3 What the error destroys is the gradient, not the policy

A flat accuracy curve would be unremarkable if the training signal were scarce to begin with, since error would then have little left to remove. Two quantities recorded during training rule that reading out, however: the signal is not merely scarce but is destroyed in proportion to the injected error.

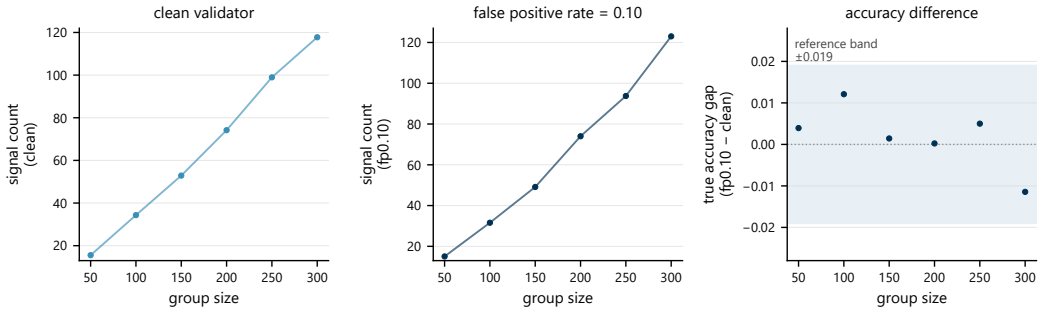


Figure 3: **Degeneracy accumulates during training under both clean and noisy verification.** The degenerate-group rate rises from about 0.59 to about 0.70 over 300 training steps under both the clean verifier and a false-positive rate of 0.10. The two trajectories remain closely aligned, which indicates that policy convergence and verifier error contribute separately to the loss of signal. Faint markers are individual seeds.

Table 1: Larger groups carry more signal, and accuracy does not respond. Trajectories per step are fixed at 32, so that only the composition of the groups changes; the accuracy difference is comparable to seed-level variation throughout.

group size	degenerate-group rate	within-group reward s.d.	Δ accuracy
4	0.571	0.195	-0.026
8	0.412	0.235	$+0.005$
16	0.292	0.278	-0.004

Figure 2 plots both quantities against the false-positive rate at horizon 6. Accuracy varies by 0.036 in total, less than two standard deviations, neither monotonically nor with any trend. The degenerate-group rate, by contrast, rises monotonically throughout and moves sharply between rates of 0.20 and 0.40, from 0.685 to 0.903 to 1.000. At 0.50 every group on every seed has uniform rewards, so that no sample anywhere in the batch carries a nonzero advantage; accuracy in that condition is nevertheless 0.547 against 0.542 for the clean baseline. Table 4 in Appendix E lists the values together with the run selection. Degeneracy grows with training as well, and independently of the verifier: by step 300, roughly 70% of groups contribute no reward-driven gradient even when the verifier is clean (Figure 3). Accordingly, the rate is informative only against a clean baseline measured under the same configuration.

Enlarging the group supplies more signal without changing what accuracy reports: raising the group size from 4 to 8 to 16 lowers the degenerate-group rate and raises the within-group standard deviation of rewards, while the accuracy difference between noisy and clean training shows no trend (Table 1). Although a larger group gives verifier error more signal to destroy, and although more signal is accordingly destroyed, accuracy registers none of the difference.

4.4 A PARAMETER-FREE ACCOUNT OF THE COLLAPSE

The shape of the curve follows from how realistic verifier errors are distributed. Since the decision to accept a wrong answer attaches to the question rather than to the answer text, a group is either released as a whole, with all rewards equal to one, or else left untouched. A released group has uniform rewards, hence zero advantage on every sample, and contributes nothing; an untouched group contributes exactly what it would have contributed under a clean verifier. Error of this kind removes training signal and never redirects it, which is why the policy is unharmed. Writing d_0 for the degenerate-group rate under a clean verifier and f for the realized conditional rate at which wrong answers are accepted, a fraction f of the $(1 - d_0)$ groups that would have carried signal are converted to uniform ones, so

$$\deg(f) \;=\; d_0 + f\,(1 - d_0). \tag{1}$$

Table 2: Equation 1 against measurement at horizon 6, three seeds per condition. The row at a target rate of 0.30 is held out: it was not used in deriving the expression, and the interval $[0.72, 0.82]$ was our prediction for it.

target rate	realized f	predicted	measured	deviation
0.10	0.010	0.578	0.581	+0.003
0.20	0.247	0.679	0.685	+0.006
0.30	0.526	0.798	**0.800**	**+0.002**
0.40	0.747	0.892	0.903	+0.011
0.50	0.978	0.990	1.000	+0.010

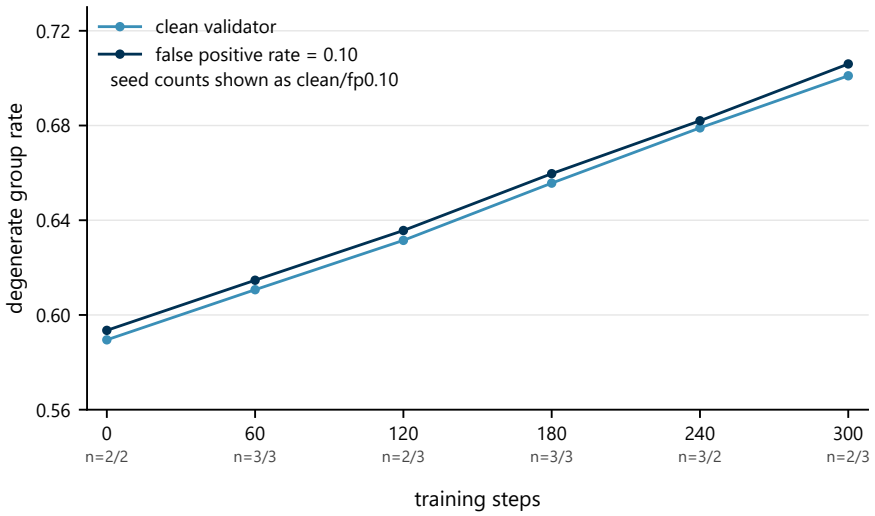


Figure 4: **Verifier-induced degeneration does not steepen materially with horizon.** $S(H)$ is the degenerate-group rate at a false-positive rate of 0.50 minus the clean-verifier rate at the same horizon; the means are 0.387, 0.396 and 0.427 for $H = 0, 3, 6$, a spread of 0.040. Open markers show all seed-pair gaps and filled markers the means. The shaded band is the preregistered uncertainty region and the dashed line the prewritten threshold $S(6) - S(0) \geq 0.15$, which the observed increase does not approach.

Nothing in Equation 1 is fitted, and its substantive content lies in the constancy of d_0: were the error redirecting the update rather than removing it, the policy's answer distribution would shift as training proceeded, and the clean baseline would move along with it.

Predictions are accurate to 0.015 across the sweep (Table 2), and two features of the fit bear on its interpretation. All five deviations are positive, so that the expression slightly underestimates degeneracy, by about 0.008 on average; although the bias is far smaller than seed-level variation, its sign is consistent, and the natural explanation is that Equation 1 accounts only for error while training contributes degeneracy of its own. At target rates of 0.10 and below, separately, the realized conditional rate is close to zero, because the verifier's natural false-positive rate of about 0.09 exceeds the 0.06 assumed when the budget was set and is subtracted from it. The flat left end of Figure 2 is therefore a near-zero rate rather than a threshold.

4.5 Why the horizon does not matter

Equation 1 gives $S(H) = f\,(1 - d_0(H))$ for the increase in degeneracy that error causes at horizon H, with no interaction between the two. Since the intuition motivating this study was that errors compound along a trajectory, we tested the prediction on two horizons the sweep above does not

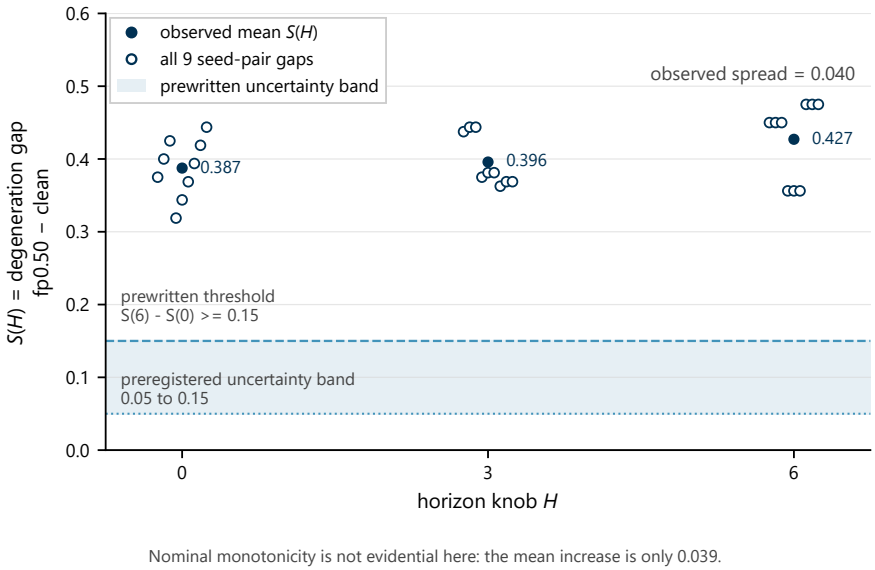


Figure 5: **The parameter-free account transfers across models, environments and error types, and fails only where the error is not attached to the task.** Deviation of the measured degenerate-group rate from the prediction $d_0 + f(1-d_0)$; points at zero agree with it. Task-attached error agrees across model families, a second environment, false negatives and KL regularization. Independent per-sample error deviates by 0.16 at matched rate, which marks the boundary of the account; at the highest error rate both kinds saturate and the deviation vanishes.

cover. The measured values of S are 0.388, 0.396 and 0.427 at horizons 0, 3 and 6, against predictions of 0.42, 0.39 and 0.42 (Figure 4). Such variation as there is derives from the clean baselines, which are themselves non-monotone at 0.569, 0.602 and 0.573. The single-turn tolerance result therefore carries over to agents, although not for the reason the compounding argument would suggest: a task-attached error removes an entire group at once, and that operation is indifferent to the length of the trajectory.

4.6 Other models, another environment, and the size of what is lost

The clean baseline d_0 grows with the model, from 0.573 for a 4B model to 0.631 for a 9B model of the same family and 0.694 for a 4B-class model of another family, and Equation 1 matches each to within 0.023 of the condition means (Figure 5). Within one condition of the third family, however, seeds disagree more than they do elsewhere, at -0.053 and $+0.078$. In a second environment, 2WikiMultihopQA (Ho et al., 2020), where the natural false-positive rate is 0.082, every qualitative feature reappears, and it does so with the lower clean baseline of about 0.51 that harder questions and more dispersed answers would predict.

Whether the loss is visible depends on how much reinforcement learning contributes in the first place. In the retrieval setting, where 300 steps of clean training add about $+0.07$ accuracy, nothing shows. On a mathematics subset chosen for moderate difficulty, by contrast, clean training adds $+0.146$, and there the same false-positive rate of 0.50 leaves 29% of that gain (Figure 6, Appendix C). Evaluated on 300 unseen questions the surviving fraction is 60.8%, close to the 60% measured during training, so the difference reflects what the policy learned rather than what it memorized. The mechanism therefore does not change with the size of the gain, although its visibility does, and this reconciles our null result with the accuracy-visible collapse reported by He et al. (2026). In the retrieval setting the corresponding fraction is 60% rather than the 50% or less we had predicted. The likely reason is that the acceptance decision includes the epoch, so training that passes over the data many times redraws which questions are removed and no question is removed permanently, whereas a single pass removes them for good. We have not tested this.

4.7 Independent error behaves differently

Adding the answer text to the hash key makes each sample's corruption independent while leaving everything else unchanged. At matched rate the degenerate-group rate is 0.67 against 0.82 for task-attached error, so independent corruption does leave usable variation inside groups. Unlike task-attached error, however, it also costs accuracy, by 0.042 on average and with individual seeds at -0.14 and -0.15. Equation 1 overpredicts this condition by 0.16, seven times its largest deviation elsewhere (Figure 5), which marks the boundary of its applicability. The contrast isolates the mechanism: error that attaches to a task removes the learning signal, whereas error drawn per sample retains the signal and misdirects it. Two further conditions behave as Equation 1 requires. False negatives obey the symmetric form to within 0.014, and enabling KL regularization changes neither d_0 nor the fit to three decimal places.

4.8 Which repair works, and why the other cannot

The usual response to a degenerate group is to draw more samples, and the mechanism distinguishes two versions of that response. Drawing fresh samples of the same question cannot help, since acceptance was decided for the question: every new wrong answer is accepted as well, and the group stays uniform. Drawing a different question can help; because a fresh question yields a usable group only with probability $(1 - d_0)(1 - f)$, however, the sampling cost relative to a clean verifier is $1/(1-f)$. Simulation over 20,000 questions per rate, using the injection code of the experiments, bears this out (Figure 7, Appendix C): across more than 45,000 affected groups, each given up to 200 attempts, resampling the same question recovers not one. The same procedure does recover about 0.29 of the groups that are uniform because the question is too hard, independently of rate. This is the part that misleads in practice: the degenerate rate visibly falls and can be read as evidence that the error has been addressed. The cost of replacing the question follows $1/(1 - f)$ to within 0.12 up to $f = 0.80$. In the mathematics setting a target rate of 0.50 corresponds to $f = 0.80$ and a fivefold cost, which is affordable, whereas in the retrieval setting the same target gives $f = 0.99$ and a hundredfold cost, which is not. Once almost every wrong answer is accepted, therefore, no amount of resampling helps, and the verifier itself has to be fixed.

4.9 What to measure instead

Downstream accuracy should not be used to certify a verifier. The degenerate-group rate is the better instrument: it costs nothing to compute during training, increases monotonically with the false-positive rate, saturates at a value with an unambiguous meaning, and coincides with an established metric (He et al., 2026) at threshold zero. It should nonetheless be read against a clean baseline measured under the same configuration, since convergence alone drives most groups to uniformity late in training. Because the rate moves sharply over a narrow range, finally, the return on hardening a verifier is uneven: below that range hardening changes the training signal very little, whereas above it training does not proceed at all.

5 Conclusion

Verifier error in long-horizon agent training removes the gradient rather than degrading the policy. Across false-positive rates up to 0.5, a fourfold range of horizons and a sixfold range of training length, task accuracy shows no measurable cost; in the same runs, however, the fraction of groups producing no reward-driven gradient rises from 0.573 to 1.000. The reason is structural, in that realistic verifier mistakes attach to the task, so that every sample of an affected question is accepted at once, the group's advantage vanishes, and the group leaves the gradient without changing its direction. The parameter-free expression this implies predicts a held-out condition to 0.002 and holds across two environments, three model families, false negatives, KL regularization, and two training regimes. Where reinforcement learning contributes more, the same error rate costs 71% of the gain, so what changes with the setting is not the mechanism but whether accuracy can see it.

AI use statement

(This section is **required** and does not count toward the page limit.)

We used generative AI tools substantially in this work. An AI assistant wrote the experiment code, the cluster scheduling scripts, and the analysis code; ran the training and evaluation jobs; drafted the analysis plan and the prose of this paper; and reconciled the numbers reported here against the raw training logs. The human authors set the research question and hypotheses, decided which experiments to run and which to decline, judged what the results supported, produced the figures, verified the citations, and approved the final text and every claim in it. Generated code was reviewed by the human authors, and self-tests were written for the parts of the implementation that cannot be checked by reading, in particular the numerical equivalence of the log-probability path and the behaviour of the error injection. We take responsibility for the final content of this work, including all text, claims, and artifacts.

Ethics statement

This work trains small open-weight language models on public question-answering corpora. It involves no human subjects, no personal data, and no new capability for misuse. The failure mode we report is invisible to standard evaluation, so we regard reporting it as reducing risk: practitioners who certify a verifier by downstream accuracy may be training on a signal that has already collapsed.

Reproducibility statement

Every acceptance decision in the error injection comes from a keyed hash rather than a random number generator, so a configuration reproduces exactly; the key structure and its consequence for within-group correlation are given in Section 3.2. Section 3.4 records the implementation constraints that most affect reproduction. The run selection, seed composition, and averaging window behind Figure 2 are given in Appendix E. The analysis code, the self-tests for the log-probability path and the error injection, and the per-seed data will be released with the paper.

References

Aakriti Agrawal, Minghui Liu, and Furong Huang. Verigate: Verifier-gated step-level supervision for grpo, 2026. URL https://arxiv.org/abs/2605.30451.

Ivan Bercovich, Ivgeni Segal, Kexun Zhang, Shashwat Saxena, Aditi Raghunathan, and Ziqian Zhong. Terminal wrench: A dataset of 331 reward-hackable environments and 3,632 exploit trajectories, 2026. URL https://arxiv.org/abs/2604.17596.

Zeyun Deng, Yuzhe Lu, Yawei Wang, Linbo Liu, Qing Ping, Han Ding, Guande Wu, Panpan Xu, and Jun Huan. Prism-grpo: Faster vla policy optimization via splitting same-outcome groups, 2026. URL https://arxiv.org/abs/2608.17423.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. Agentic reinforced policy optimization, 2025. URL https://arxiv.org/abs/2507.19849.

Florian E. Dorner, Yatong Chen, André F. Cruz, and Fanny Yang. Roc-n-reroll: How verifier imperfection affects test-time scaling, 2026. URL https://arxiv.org/abs/2507.12399.

Kazuki Egashira, Mark Vero, Jasper Dekoninck, Florian E. Dorner, Robin Staab, and Martin Vechev. Delay, plateau, or collapse: Evaluating the impact of systematic verification error on rlvr, 2026. URL https://arxiv.org/abs/2605.02909.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean

Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://dx.doi.org/10.1038/s41586-025-09422-z.

Xixiang He, Qiyao Sun, Ao Cheng, Xingming Li, Xuanyu Ji, Hailun Lu, Runke Huang, and Qingyong Hu. Advantage collapse in group relative policy optimization: Diagnosis and mitigation, 2026. URL https://arxiv.org/abs/2605.21125.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In Donia Scott, Nuria Bel, and Chengqing Zong (eds.), *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.580. URL https://aclanthology.org/2020.coling-main.580/.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

Jian Hu, Jason Klein Liu, Haotian Xu, and Wei Shen. Reinforce++: Stabilizing critic-free policy optimization with global advantage normalization, 2025a. URL https://arxiv.org/abs/2501.03262.

Zengjie Hu, Jiantao Qiu, Tianyi Bai, Haojin Yang, Binhang Yuan, Qi Jing, Conghui He, and Wentao Zhang. Vade: Variance-aware dynamic sampling via online sample-level difficulty estimation for multimodal rl, 2025b. URL https://arxiv.org/abs/2511.18902.

Yibin Huang, Bin Xu, Hailong Cao, and Conghui Zhu. Bicaa: Bidirectional credit assignment for search-augmented agent, 2026. URL https://arxiv.org/abs/2608.01321.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning, 2025. URL https://arxiv.org/abs/2503.09516.

Junbo Li, Peng Zhou, Rui Meng, Meet P. Vadera, Lihong Li, and Yang Li. Turn-PPO: Turn-level advantage estimation with PPO for improved multi-turn RL in agentic LLMs. In Vera Demberg, Kentaro Inui, and Lluís Marquez (eds.), *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 6227–6243, Rabat, Morocco, March 2026. Association for Computational Linguistics. ISBN 979-8-89176-386-9. doi: 10.18653/v1/2026.findings-eacl.328. URL https://aclanthology.org/2026.findings-eacl.328/.

Jiashun Liu, Johan Obando-Ceron, Han Lu, Yancheng He, Weixun Wang, Wenbo Su, Bo Zheng, Pablo Samuel Castro, Aaron Courville, and Ling Pan. Asymmetric proximal policy optimization: mini-critics boost llm reasoning, 2025. URL https://arxiv.org/abs/2510.01656.

Yijun Lu, Rui Ye, Jiajun Wang, Yuwen Du, Tian Jin, Songhua Liu, and Siheng Chen. Abseeker: Training long-horizon search agents via answer-backtracked credit assignment, 2026. URL https://arxiv.org/abs/2608.05102.

Ang Lv, Ruobing Xie, Xingwu Sun, Zhanhui Kang, and Rui Yan. The climb carves wisdom deeper than the summit: On the noisy rewards in learning to reason, 2025. URL https://arxiv.org/abs/2505.22653.

Omar El Mansouri, Fathinah Asma Izzati, Mohamed El Amine Seddik, and Salem Lahlou. Noise-corrected grpo: From noisy rewards to unbiased gradients, 2026. URL https://arxiv.org/abs/2510.18924.

Parsa Mazaheri and Kasra Mazaheri. Prior audit-repair context shifts llm verifier thresholds toward leniency, 2026. URL https://arxiv.org/abs/2608.16003.

Amritansh Mishra, Supriyo Chakraborty, and Berkcan Kapusuzoglu. On the policy gradient foundations of group relative policy optimization: Credit assignment, gradient sparsity, and rank collapse, 2026. URL https://arxiv.org/abs/2606.29238.

Ryo Mitsuhashi, Patrick Chen, Isabelle Tseng, Jasin Cekinmez, and Addison J. Wu. Quantifying empirical compute-supervision tradeoffs in rlvr, 2026. URL https://arxiv.org/abs/2605.25252.

Andreas Plesner, Francisco Guzmán, and Anish Athalye. An imperfect verifier is good enough: Learning with noisy rewards, 2026. URL https://arxiv.org/abs/2604.07666.

Ali Rad, Khashayar Filom, Darioush Keivan, Peyman Mohajerin Esfahani, and Ehsan Kamalinejad. Rate or fate? rlv$^{\varepsilon}$r: Reinforcement learning with verifiable noisy rewards, 2026. URL https://arxiv.org/abs/2601.04411.

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, 2024. URL https://arxiv.org/abs/2402.03300.

Leitian Tao, Baolin Peng, Wenlin Yao, Tao Ge, Hao Cheng, Mike Hang Wang, Jianfeng Gao, and Sharon Li. Trace: Turn-level reward assignment via credit estimation for long-horizon agents, 2026. URL https://arxiv.org/abs/2607.13988.

Haoze Wu, Chuqiao Kuang, Tianyi Zhuang, and Xiaoguang Li. Beyond outcome rewards: Step-level self-distilled policy optimization for deep search agents, 2026. URL https://arxiv.org/abs/2608.12764.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL https://aclanthology.org/D18-1259/.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Chuyifei Zhang. When the reward suite is leaky: A preregistered causal contrast of natural verifier false positives in rlvr, 2026. URL https://arxiv.org/abs/2607.11022.

Guibin Zhang, Hejia Geng, Xiaohang Yu, Zhenfei Yin, Zaibin Zhang, Zelin Tan, Heng Zhou, Zhongzhi Li, Xiangyuan Xue, Yijiang Li, Yifan Zhou, Yang Chen, Chen Zhang, Yutao Fan, Zihu Wang, Songtao Huang, Francisco Piedrahita-Velez, Yue Liao, Hongru Wang, Mengyue Yang, Heng Ji, Jun Wang, Shuicheng Yan, Philip Torr, and Lei Bai. The landscape of agentic reinforcement learning for llms: A survey, 2026. URL https://arxiv.org/abs/2509.02547.

Xingjian Zhang, Siwei Wen, Wenjun Wu, and Lei Huang. Edge-grpo: Entropy-driven grpo with guided error correction for advantage diversity, 2025. URL https://arxiv.org/abs/2507.21848.

Bingchen Zhao, Dhruv Srikanth, Yuxiang Wu, and Zhengyao Jiang. Specbench: Measuring reward hacking in long-horizon coding agents, 2026. URL https://arxiv.org/abs/2605.21384.

Ziqian Zhong, Ivgeni Segal, Ivan Bercovich, Shashwat Saxena, Kexun Zhang, and Aditi Raghunathan. Hardening agent benchmarks with adversarial hacker-fixer loops, 2026. URL https://arxiv.org/abs/2606.08960.

A Scope and limitations

The mechanism requires that advantages be computed relative to other samples of the same prompt, which covers GRPO together with the leave-one-out and globally normalized estimators built on the same idea. It does not cover methods that learn a value function. There a corrupted reward enters the temporal-difference target and moves the update in a wrong direction rather than removing it, so the damage should appear in capability rather than hide from it, and our conclusion should not be carried over. That difference is testable, and it bears on the current argument for returning to critics in long-horizon settings (Li et al., 2026; Liu et al., 2025): should the failure described here account for the brittleness of group-relative estimation in this regime, a critic would trade an invisible loss of signal for a visible loss of accuracy, which is a different and more tractable problem.

Absence of a measurable accuracy cost is not absence of a cost. With a seed-level standard deviation of 0.019 and three seeds in each condition of the main sweep, effects below about 0.05 remain possible, although this does not weaken the main result, whose effect size is tens of standard deviations. Our coverage extends to two environments, three model families and two training regimes, whereas models above 10B, agent environments outside retrieval, and settings in which verifier error is not task-attached remain untested. Since realized tool calls reach only 7.3, moreover, the absence of an interaction with horizon should not be extrapolated to trajectories of dozens of turns. Nor can the verifier's natural false positives be separated from the injected ones. Because we sample sparsely between target rates of 0.20 and 0.40, the rate at which degeneracy accelerates is bracketed rather than located. Finally, the curve in Figure 2 was measured at a single horizon, and the horizon comparison that follows from it was run afterwards on conditions the curve does not contain.

B Future work

Task-attached and independent error are two ends of an axis rather than two categories, and flipping a fraction of the samples within a question would fill in the interval between them. Replicating the high-gain result in agent environments that support larger gains would then test the claim that the mechanism is fixed while its visibility is not. Finally, the two repairs analysed above should be run end to end, at a rate where the damage is visible in accuracy.

C Gain under a larger loss space, and the two repairs

D Setup validation

E Per-condition values and run selection

Runs are selected by configuration: group size 4, eight prompts per step, minimum six retrievals, one evidence passage per retrieval, 256 new tokens, KL regularization disabled, Qwen3.5-4B. Where a

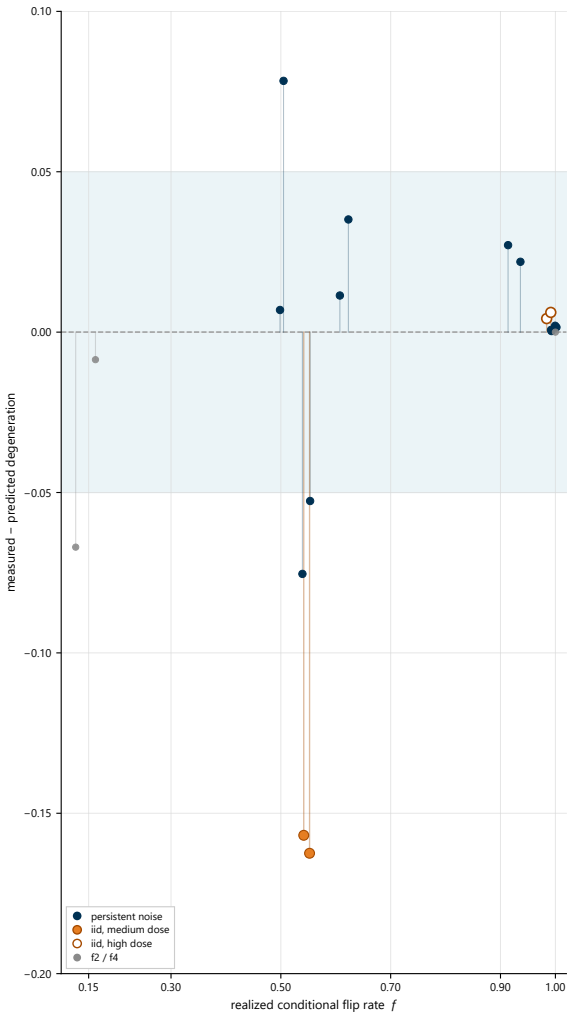


Figure 6: Verifier error becomes visible where reinforcement learning contributes more. At a false-positive rate of 0.50, 29% of the clean gain survives on the mathematics subset and 60% in the retrieval setting. Solid points are means and open points individual seeds.

Table 3: The horizon parameter moves realized tool use without making the task harder. Each row is six seeds.

min. retrievals	realized turns	p90	evidence coverage	clean accuracy
0	1.90	4	0.532	0.520
3	3.86	6	0.702	0.584
6	7.33	10	0.776	0.605

condition has more than one run, the longest is used, and averages are taken over steps 20 to 40. Per-seed values for every condition in the paper, including the cross-model conditions whose seeds disagree most, will be released with the paper.

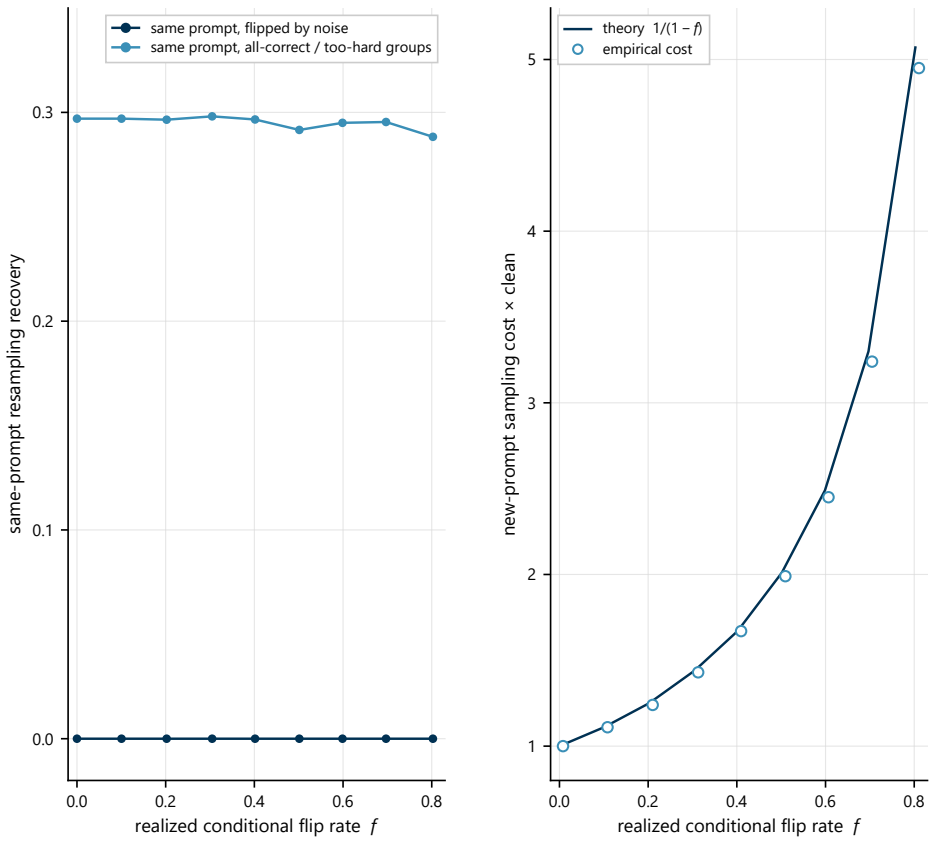


Figure 7: The effective repair depends on the source of degeneracy. Resampling the same question recovers none of the groups made uniform by task-attached error, although it recovers about 0.29 of those uniform because the question is too hard; replacing the question costs $1/(1-f)$. Derived from the injection mechanism and confirmed in simulation, with no training involved.

Table 4: Values behind Figure 2, at horizon 6 over steps 20 to 40.

false-positive rate	accuracy	seeds	degenerate-group rate	seeds
clean	0.542	3	0.573	3
0.10	0.528	3	0.581	3
0.20	0.541	3	0.685	3
0.30	0.538	3	0.800	3
0.40	0.535	3	0.903	3
0.50	0.547	3	1.000	3

F Implementation constraints

On the architecture used here, computing log probabilities one sample at a time without padding agrees exactly with a strictly serial implementation, while padded batching perturbs a sample's log probability by up to 1.4×10^{-1}, with a magnitude that grows with sequence length. Generation uses batched HuggingFace generation rather than vLLM because the available driver predates the CUDA version its current builds require. Stopping generation at the closing search or answer tag avoids charging text beyond that boundary to token-level credit.