ACQUISITION OR REDISTRIBUTION? AUDITING EVIDENCE BEYOND BASE SUPPORT

Anonymous authors
Paper under double-blind review

ABSTRACT

Improved answer accuracy after reinforcement learning does not by itself distinguish acquired capability from a redistribution of probability over solutions already accessible to a base model. We examine this distinction through an artifact audit of a Qwen3-8B mathematical-reasoning study. Historical score vectors identify ten problems with zero base credits and 75 trained credits at 128 samples per problem. A subsequent evaluation preserves 20,480 generations from these problems. We exactly reproduce its 19 base and 12 trained automatic credits, but every generation reaches the 1024-token limit. Response inspection exposes coefficient matches, scalar/root-pair normalization collisions, and an uncredited prefix that derives the correct value. Consequently, neither the automatic counts nor an audit restricted to credited responses yields a reliable estimate of answer correctness. A separate forked-update evaluation records differences of one credit in each of two probe groups, but its implementation is not verified standard GRPO, and adapter identity and train–probe separation remain insufficiently established for causal interpretation. We provide deterministic replay, traceable counterexamples, and an evidence checklist for beyond-observed-support claims. This case establishes the existence, not the population prevalence, of failures in the measurement chain; it leaves acquisition versus redistribution unresolved.

1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) can increase the probability of a correct answer. Whether this increase reflects acquired reasoning capability or more frequent use of existing solutions is a different question. GRPO (Shao et al., 2024) makes this distinction particularly relevant: a higher mean reward is directly useful, but does not identify how the distribution of reasoning changed. Recent work studies RL-trained reasoning relative to base-model sampling coverage (Yue et al., 2025). Interpreting a change in observed coverage requires both a definition of support and a trustworthy way to measure it.

An observed zero is conditional on a protocol. A model might produce no credited answer because the required reasoning is rare, because the generation ends before the answer, or because the scorer fails to recognize the answer. Conversely, a scorer may credit a number that occurs in unfinished work without being the requested answer. Increasing the sample count addresses sampling uncertainty under the measurement procedure; it does not establish that the procedure measures the intended outcome.

We audit experiments developed for this submission using Qwen3-8B (Yang et al., 2025) and LoRA (Hu et al., 2022); this is a self-audit, not a replication audit of a published external study. At the population level, retained score vectors contain apparent upward movement on problems with no base credits. For one run, ten such problems account for 75 trained credits among 128 samples per problem. At the update level, a fork design compares checkpoints immediately before and after a purported training update. Both settings invite an interpretation about capability. Our question is narrower: *which parts of that interpretation can the retained artifacts actually support?*

The strongest evidence comes from a later evaluation that saved every generation on the ten selected problems. Replaying its scorer exactly reproduces all stored labels, yet exposes a distinction between computational reproducibility and measurement validity. All 20,480 outputs end at the token limit. Among the 31 credited outputs, some credits come from an unfinished coefficient or a scalar conflated

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.

with a root pair. An uncredited output, meanwhile, derives the correct value before continuing into a truncated expression. Inspecting only positive labels would miss this second failure and could turn a scoring audit into an unjustified accuracy estimate.

The forked-update records illustrate another break in the evidence chain. We can count the recorded credits, inspect the implemented objective, and identify adapter-ID reuse. We cannot infer from these artifacts alone that distinct intended adapter states were loaded at every evaluation call or that the probes were held out from the training trajectory. These are limitations of causal identification, even if the score arithmetic is correct.

Our contributions are three concrete outputs. First, we separate score movement from a claim about acquired capability, using a protocol-conditional support statistic as an exploratory summary rather than a new calibrated estimator. Second, we provide an exact replay of the retained response-level labels and trace failures to identifiable generations and scorer operations. Third, we audit a forked-update implementation and specify the missing evidence needed to interpret its contrasts. The resulting paper is a single-study diagnostic case, not an estimate of failure prevalence across RLVR. It neither establishes acquisition nor establishes its absence.

2 Related Work

RLVR and reasoning. GRPO was introduced in DeepSeekMath (Shao et al., 2024); PPO (Schulman et al., 2017) provides a broader policy-optimization context. DeepSeek-R1 (Guo et al., 2025) and Qwen3 (Yang et al., 2025) describe large-scale reasoning-model training. Our audit concerns a small-scale implementation and its retained evidence. The name of a training method does not certify that a particular implementation executes that method or reproduces the training conditions of these systems.

Acquisition and search-accessible support. Yue et al. (2025) investigate RL reasoning beyond base-model coverage. Matsutani et al. (2026) compare reasoning changes under RL and supervised fine-tuning. Inference-time scaling studies (Snell et al., 2025; Brown et al., 2024) emphasize that search budget changes observed performance. Our work does not adjudicate these studies' broader conclusions. It examines whether the measurements in one study warrant a capability interpretation.

Multiple-generation evaluation. Pass@k estimation (Chen et al., 2021), best-of-N selection (Nakano et al., 2021), and self-consistency (Wang et al., 2023) use multiple outputs in different ways. Their outcomes depend on sampling and answer selection. We evaluate artifacts for MATH (Hendrycks et al., 2021), using the MATH-500 subset associated with process-supervision evaluation (Lightman et al., 2023). A large number of draws is informative only about the outcome that the evaluator actually records.

Uncertainty versus validity. The rule of three (Hanley & Lippman-Hand, 1983) approximates a confidence bound after zero successes. Such a bound concerns repeated draws under its assumptions; it cannot turn a misspecified score into a measure of semantic correctness. Likewise, uncertainty about a treatment effect presupposes a well-defined treatment. We make no priority claim for these statistical principles. The contribution is their application to inspectable artifacts in an acquisition-versus-redistribution study.

3 Measurement Framework and Audit Design

3.1 A Protocol-Conditional Support Statistic

Let a protocol specify the model and adapter, problem set, prompt and chat template, thinking configuration, sampler, token budget, answer extraction, and answer equivalence rule. For problem i, let b_i and r_i denote empirical rates of *automatic credit* for the base and trained model. We use this term throughout because score validity is itself under investigation. With P problems, define

$$G_{\text{emp}} = \sum_{i=1}^{P} (r_i - b_i)_+, \qquad L_{\text{emp}} = \sum_{i=1}^{P} (b_i - r_i)_+, \tag{1}$$

where $(x)_+ = \max(x, 0)$. The mean net score change is $100(G_{\text{emp}} - L_{\text{emp}})/P$ percentage points. Positive-part aggregation separates upward and downward movements but can be nonzero under sampling noise.

For $G_{\text{emp}} > 0$, define the empirical beyond-observed-base-support share

$$\text{BOSS}_{\tau}^{\text{emp}} = \frac{\sum_i (r_i - b_i)_+ \mathbf{1}[b_i \leq \tau]}{G_{\text{emp}}}. \tag{2}$$

At $\tau = 0$, the numerator uses problems with no observed base credits. At $\tau > 0$, it includes low-rate problems, not just observed zeros. For example, $\tau = 0.05$ measures the share of upward movement assigned to that low-rate subset. It is not the fraction of solutions newly acquired or the fraction of trained-model successes outside base capability. We retain 0.05 as the historical exploratory cutoff, not an independently justified capability boundary; Appendix F reports a deterministic threshold grid.

An empirical zero does not establish zero underlying probability. Under independent Bernoulli outcomes with probability p, the probability of n failures is $(1 - p)^n$. After zero successes, the one-sided 95% upper bound is $1 - 0.05^{1/n}$, approximately $3/n$. At $n = 128$ it is about 2.31%. This calculation is a per-problem statement under a fixed, valid outcome definition, not a simultaneous guarantee over a post-selected set. We report deterministic summaries of existing records; we do not add synthetic outcomes or use a simulated calibration as empirical evidence.

3.2 Evidence Levels

The audit distinguishes four levels. *Recorded scores* establish what the saved evaluator credited. *Exact replay* establishes that the supplied extraction and comparison code reproduces those records. *Response inspection* checks what a particular text supports: an extracted answer, an intermediate result, or a completed response. *Mechanistic interpretation* additionally requires comparable generation protocols and, for an intervention, verified treatment identity. Success at an earlier level does not establish a later one.

We retain sample IDs rather than collapsing outputs to an aggregate. The response audit covers every credited output and uses explicit counterexamples among uncredited outputs. It is not a comprehensive semantic adjudication of all negative labels. We therefore do not divide the number of accepted positive examples by the total sample count and call the result corrected accuracy.

3.3 Artifact Scope and Selection

The historical population summaries contain per-problem score vectors for 500 problems, with 128 samples per problem, for four records labelled GRPO and one labelled SFT. The base vector is shared. Here, *base* denotes the starting Qwen3-8B checkpoint before the local updates, not a claim that it is a pretrained-only model. These summaries lack the full corresponding generations. We use their vectors to reconstruct descriptive statistics, not to validate their original accuracy or all recorded training settings.

The response-level evaluation covers the ten indices selected from the first historical GRPO record by $b_i = 0$ and $r_i > 0$. Each model has 1024 saved outputs per selected problem. The JSON records raw text, parsed answer, credit, token count, and finish reason. The protocol metadata records a chat prompt, temperature 0.8, top-p 0.95, and a 1024-token generation limit. It does not preserve a chat-template hash or an explicit thinking-state record. We call this a *common recorded protocol*, rather than a verified replay of the historical evaluation.

The forked-update file contains three fork records, with supported and uncovered probe groups, ten probes per group, and 128 outputs per probe and arm. It saves parsed answers and finish metadata but not raw response text. The supplied code defines the intended operations; without a complete execution trace it does not prove every historical runtime event. Our deterministic replay and code inspection are deliberately separate.

Provenance boundary. The source records and code were frozen for this audit on September 26, 2026. They originate from the authors' experiment workspace. Original runtime timestamps, model-weight hashes, and template hashes are not consistently present, so the capture date is not

presented as an experiment date. Appendix G identifies the frozen objects by hash and distinguishes verified replay links from unverified historical links. In particular, exact scorer replay establishes agreement with the saved labels, not proof of which scorer revision executed during generation.

4 Population Scores and Response-Level Audit

4.1 What the Historical Vectors Establish

Table 1 reconstructs the saved vectors using Equation (2). All records have a mean base score of 67.353%, and 78 problems have zero base credits. The labels in the first column identify files; they do not certify that the intended optimization settings were executed. Recorded trained means exceed the recorded base mean by 0.680–1.655 percentage points. This is an arithmetic fact about saved labels, not a validated estimate of capability change.

Table 1: Reconstructed historical score summaries on 500 problems, 128 samples per problem. BOSS is the *empirical* statistic at $\tau = 0.05$. Zero-base credits count trained credits on all 78 problems with no base credits. Values are descriptive; full historical responses and protocol equivalence are not verified.

Record	Trained score (%)	Change (pp)	BOSS (%)	Zero-base credits
GRPO-A	68.033	+0.680	11.77	75
GRPO-B	68.070	+0.717	13.19	74
GRPO-C	68.634	+1.281	7.58	67
GRPO-D	68.178	+0.825	8.00	69
SFT	69.008	+1.655	6.87	118

The complement of BOSS at 0.05 assigns 86.81–93.13% of gross upward score movement to problems with base rate above 0.05. It does not establish that this movement is redistribution, nor that the remaining movement is acquisition. The threshold, finite sampling, selection, and scoring all affect the statistic. We make no significance or equivalence claim from these values.

Figure 1 plots the cumulative share of gross upward movement against problem rank for each record: the ten largest per-problem movements contribute 30.5–42.9% of G_{emp}, so the summaries in Table 1 are dominated by a small number of problems. This is a descriptive property of the saved vectors, not an estimate of problem-level acquisition.

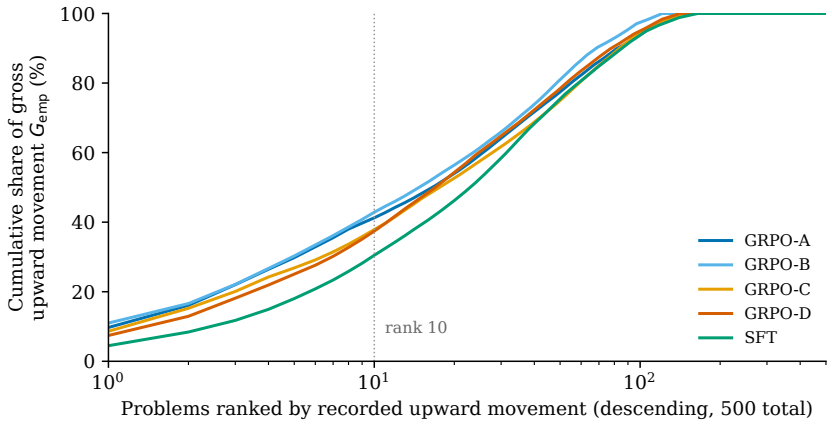

Figure 1: Cumulative share of gross upward movement G_{emp} against problems ranked by recorded upward movement, recomputed from the saved historical vectors (500 problems, 128 samples each). The ten largest per-problem movements contribute 30.5–42.9% of G_{emp} across the five records. Counts describe fixed artifacts, so no inferential error bars are shown.

For GRPO-A, the 75 trained credits on zero-base problems fall on ten indices. Two problems, 326 and 478, contribute 41 and 24 credits, respectively. This concentration motivates response-level

inspection, but also makes selection explicit: the set was chosen using trained success. Results on this set cannot estimate the frequency of support expansion on an independent benchmark sample.

Historical comparability is not certified. The population evaluation script loads a cached base evaluation named `base_eval_2048.json` while permitting the trained token limit to be set separately. Another training script changes the model directory's tokenizer template when its non-thinking option is used. A cache filename alone does not prove a token budget, and a mutable model path does not establish a tokenizer state. The retained summaries therefore do not certify equal base and trained generation protocols. We do not reconstruct missing historical runtime settings from filenames or intended defaults.

4.2 Exact Replay of the Common-Protocol Records

The later response files contain 10,240 outputs per model. Applying the supplied scorer to every saved string reproduces every parsed answer and every credit, with zero mismatches. The automatic totals are 19 base credits and 12 trained credits. Table 2 reports these original labels without recasting them as adjudicated correctness.

Table 2: Response-level automatic credits. Every problem has 1024 outputs per model; the last problem row groups seven problems. All 20,480 outputs have finish reason `length`, 1024 generated tokens, and no `\boxed{` marker. The scores are exactly replayable but do not measure completed-answer accuracy.

Problem	Base credits	Trained credits	Credited extraction
104	4	0	Coefficient in unfinished algebra
326	5	6	Correct value visible in prefix
456	10	6	Scalar/root-pair collision
100, 239, 340, 355, 466, 478, 483	0	0	No automatic credits
Total	19	12	10,240 outputs per model

The historical base truncation summary is 22.448%, whereas the selected response-level evaluation is entirely truncated. Because selection and incompletely recorded generation conditions differ, this contrast does not identify a single cause. It does establish that the later low score cannot simply replace the historical score as a measurement under verified equivalent conditions. Symmetry of the recorded token limit also does not guarantee that both models have equal opportunity to finish.

4.3 Three Response-Level Counterexamples

A coefficient is not an answer. For problem 104, the target is 40. Base sample 84 ends while manipulating two conic equations, at an expression containing $2y^2 - 40y+$. The last-number fallback extracts `40`, which matches the target even though this occurrence is a coefficient. Four base credits on this problem have this unfinished-algebra pattern. They do not establish that the requested geometric quantity was computed.

Normalization destroys answer type. Problem 456 asks for integer roots, with gold representation `-2,1`. Removing the comma converts that representation into `-21`. Consequently, the scorer accepts a parsed scalar `-21`. In base sample 217, this scalar occurs while testing a candidate root: the response has reached an arithmetic term $-7 - 14 = -21$. This is not the requested pair. The same prefix has already found roots 1 and -2, however, so an invalid extraction is not evidence that all preceding reasoning is wrong. Across both arms, 16 credits exhibit the scalar/root-pair collision.

A negative label can hide a correct derivation. For problem 326, the gold value is $9/100$. Base sample 34 derives

$$\frac{\cos(2\alpha) - \cos(2\beta)}{2} = \frac{0.68 - 0.5}{2} = 0.09, \qquad \alpha = \arcsin(0.4), \quad \beta = \arcsin(0.5).$$

It then checks the computation again and truncates at `9/1`. The fallback extracts `1`, so the saved credit is zero. The derivation is visible, but this is not a completed final answer. The example demonstrates that auditing only credited responses cannot enumerate all prefixes containing correct reasoning.

As a discovery aid, 65 base and 63 trained *uncredited* outputs on problem 326 contain the literal string `0.09`. These are string-match candidates, not counts of correct solutions; some may be tentative, copied, or contradicted. We do not convert this search into a semantic success total.

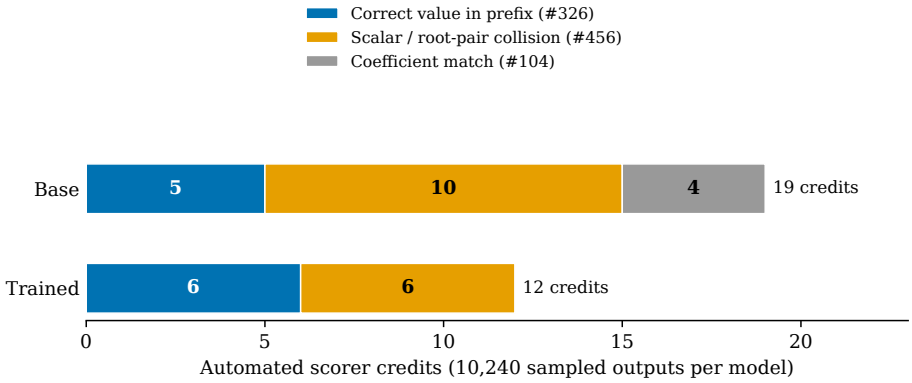

Figure 2: Inspection of the 31 automatic credits, grouped by the extraction patterns in Section 4.3. The 11 credits on problem 326 contain the correct value in truncated reasoning; they are not an estimate of all correct prefixes, because uncredited responses were not exhaustively adjudicated. Counts describe these fixed artifacts, so no inferential error bars are shown.

4.4 What This Audit Can and Cannot Correct

Replaying the scorer validates label provenance. The counterexamples invalidate the interpretation of these labels as a clean completed-answer measure. Filtering the 31 positives leaves 11 credits on problem 326 with a visible correct value, but reporting 5/10,240 versus 6/10,240 as corrected accuracy would additionally assume that every excluded or uninspected output is wrong. The uncredited counterexample shows why that assumption is not justified.

A scorer test suite is useful only for the cases it covers. The supplied suite accepts decimal/fraction equivalence, including 0.09 versus $9/100$. It nevertheless misses the root-pair collision and the last-number problems above. A separate deterministic probe also shows that normalizing `+42` yields `4`, because the leading-plus branch retains one character rather than the remaining substring. This establishes a code defect; we do not claim that it caused any observed response-level credit here.

The valid conclusion is therefore a measurement diagnosis. Neither the 19-versus-12 automatic count nor the 5-versus-6 positive-only inspection answers whether training acquired capability. A reliable re-evaluation needs a stated outcome rubric, appropriate completion budgets, and inspection that can discover both false positives and false negatives.

5 Forked Updates: Scores Without a Validated Causal Contrast

A fork design aims to compare an unchanged checkpoint (SHAM) with a checkpoint after one update (TREAT). If both evaluations load the intended states and use the same protocol, their contrast can probe the local effect of that update. This identification depends on execution provenance as well as sampling.

5.1 Recorded Outcomes

The retained file contains three forks. Each arm contributes $3 \times 10 \times 128 = 3840$ outputs per support group. The supported group records 600 SHAM credits and 601 TREAT credits; the uncovered group records 4 and 3. Thus the recorded TREAT-minus-SHAM differences are $+1/3840$ and $-1/3840$,

or approximately $+0.026$ and -0.026 percentage points. These are score differences, not certified update effects.

Of 15,360 outputs, 15,356 have finish reason `length`. All 7680 supported-group outputs are truncated; 7676 of 7680 uncovered-group outputs are truncated. The file contains parsed answers, not full generations, so the response-level semantic checks of Section 4.3 cannot be repeated on these records. Three forks also do not constitute three independent training seeds. We report no confidence interval or hypothesis test for a general training effect.

5.2 Adapter Identity Must Be Verified

The evaluator creates `LoRARequest("adapter", 1, lora_path)` for each adapter and reuses one inference engine. The integer ID is unchanged across paths. The vLLM documentation consulted for this audit specifies a globally unique adapter ID.[1] This creates an adapter-caching risk: a requested path change is not itself evidence that a distinct adapter was evaluated. The retained artifacts do not include a loaded-adapter hash trace, and we cannot establish from them that the historical runtime did or did not reuse a cached adapter. Accordingly, a small observed difference cannot be treated as evidence that the intended update had a small effect.

5.3 The Implemented Objective Matters

The inspected trajectory code accumulates gradients over eight completions. It divides each policy term by the group size but adds the full KL term for each completion before one optimizer step. Writing the completion policy loss as ℓ_j, advantage as a_j, and KL term as K_j, its accumulated objective is

$$\sum_{j=1}^{8}\left(\frac{a_j\ell_j}{8}+0.04K_j\right)=\frac{1}{8}\sum_{j=1}^{8}a_j\ell_j+0.32\left(\frac{1}{8}\sum_{j=1}^{8}K_j\right). \tag{3}$$

Relative to a mean KL term, the coefficient is 0.32, not 0.04. This is an algebraic property of the supplied code, not a measurement of the realized optimizer trajectory. The implementation uses centered rewards and weighted negative log-likelihood; it does not implement the clipped probability-ratio objective described for GRPO in DeepSeekMath. We therefore avoid attributing its recorded behavior to standard GRPO.

5.4 Probe Separation and Update Provenance

The trajectory script draws training prompts from the MATH-500 pool that also supplies the probes. Without the executed training sequence, this does not establish which particular probes were used for updates, but it prevents a claim of verified held-out evaluation. By contrast, comparing the separate 746-record population training file with the 500-record evaluation file yields zero exact-text overlaps, including after trimming or collapsing whitespace. This limited text check concerns those files, not pretraining contamination or the fork's training sequence.

Finally, the fork evaluation file identifies forks but does not itself record the optimizer-step ledger linking each evaluated adapter to its update. A directory name or loop index is not such a ledger. A valid local intervention requires both the executed update record and the loaded checkpoint identity. Until those are established, the fork counts remain descriptive observations.

6 Discussion and Limitations

Why more samples are insufficient. Sampling depth reduces uncertainty about the distribution induced by a fixed evaluator. It does not resolve whether a numeric string is the requested answer, whether a prefix is complete, or whether two calls use the intended checkpoints. In this case, exact reproduction of every automatic label coexists with concrete scoring failures. Reproducibility and validity are complementary requirements.

[1] `https://docs.vllm.ai/en/latest/features/lora.html`, accessed September 26, 2026. The historical runtime version is not preserved, so current documentation is not evidence of which cache behavior occurred in that run.

Symmetry and censoring. A common budget is a necessary part of a controlled comparison, but an all-truncation evaluation mainly reveals behavior before that budget is exhausted. It can support a budget-conditional score comparison if the score itself is valid. It cannot establish protocol-independent reasoning capability. A useful follow-up would report completion rates alongside accuracy and evaluate a predeclared range of budgets, keeping the scorer and prompt fixed.

A minimal evidence checklist. For a beyond-support claim, an evaluation should preserve the following chain:

1. **Define the outcome.** Specify whether credit requires a final answer, a correct derivation in a prefix, or successful answer extraction; preserve answer type, including sets and tuples.
2. **Record the protocol.** Hash model, adapter, tokenizer template, data, and scorer; record thinking configuration, sampling parameters, and actual termination metadata.
3. **Audit both label directions.** Inspect credited outputs and a documented sample of uncredited outputs. Preserve response IDs and a rubric before estimating corrected accuracy.
4. **Separate selection from confirmation.** Identify a candidate gain set using one evaluation, then confirm it with fresh draws and explicit selection-aware scope.
5. **Verify interventions.** Record every optimizer step, checkpoint hash, loaded adapter ID, and training/probe intersection before interpreting a fork contrast.

The checklist specifies evidence needed for the inference; it is not a claim that these requirements have all been satisfied here.

Limits of the present evidence. This audit studies one model family, one mathematical benchmark, and selected local artifacts. It provides counterexamples rather than a representative estimate of scorer error rates. Historical population generations, complete runtime manifests, and fork response text are unavailable in the audited set. The common-protocol response files preserve text but do not resolve the thinking-template state or historical comparability. Semantic inspection was not a blinded multi-rater study and did not cover all negative labels. The exact-text split check does not rule out paraphrases or pretraining overlap. No result here estimates the prevalence of these failure modes in other RLVR work.

These limitations determine the scientific scope. The acquisition claim is not established by the retained evidence. The opposite claim, that GRPO only redistributes existing capability, is also not established. The practical result is a reproducible account of why the measurements cannot decide between them.

7 Conclusion

The question of acquisition versus redistribution requires more than a gain in observed answer scores. In the audited artifacts, 20,480 saved responses allow exact label replay, but truncation and answer extraction prevent those labels from supporting the intended capability interpretation. Forked-update counts likewise remain uninterpretable as causal effects without verified checkpoint identity and execution provenance. Our contribution is an inspectable diagnostic record and a concrete evidence checklist. A future controlled evaluation can use these checks to make either a positive or a negative capability conclusion more credible.

Reproducibility Statement

The accompanying artifact contains the response records, historical score vectors, fork records, relevant code snapshots, and a deterministic Python replay. It reconstructs the reported counts and descriptive statistics without model calls or random-number generation. The appendix maps manuscript quantities to files and response IDs. This reproduces the audit of saved artifacts; missing checkpoints and runtime traces prevent a full reproduction of training.

STATEMENT ON LARGE LANGUAGE MODEL USAGE

Large language models, including GPT and Claude, assisted with code inspection, deterministic analysis scripts, manuscript drafting, and critical review. The reported numerical observations are derived from the supplied saved records; no model-generated synthetic performance results are included. LLM assistance is not an independent validation of experimental or semantic correctness. The scope and unresolved evidence gaps are stated in the manuscript.

REFERENCES

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025. doi: 10.1038/s41586-025-09422-z. URL https://doi.org/10.1038/s41586-025-09422-z.

J. A. Hanley and A. Lippman-Hand. If nothing goes wrong, is everything all right? interpreting zero numerators. *JAMA*, 249(13):1743–1745, 1983.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. *arXiv preprint arXiv:2103.03874*, 2021.

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. In *International Conference on Learning Representations*, 2022.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Kohsei Matsutani, Shota Takashiro, Gouki Minegishi, Takeshi Kojima, Yusuke Iwasawa, and Yutaka Matsuo. Rl squeezes, SFT expands: A comparative study of reasoning llms. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 141482–141526, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/e52554a70e0df57a0bea11d1eca0c9b5-Paper-Conference.pdf.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

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.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/1b623663fd9b874366f3ce019fdfdd44-Abstract-Conference.html.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 57654–57689. Curran Associates, Inc., 2025. doi: 10.52202/085713-1933. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/537d5aa768c2d534016a4d06f87bc8fb-Paper-Conference.pdf.

A Artifact Map and Replay

The supplementary archive separates immutable measurement records from replay code. Run the following command from the extracted archive using Python 3:

```
python reproduce_submission_audit.py --data-root . --out audit.json
```

The script uses only the standard library and does not access the network or call a language model. It hashes its inputs, recomputes every response-level parsed answer and credit, reconstructs population statistics, counts fork outcomes, and performs the limited split text check. Archive checksums cover the distributed files. Machine-local paths in metadata are replaced with anonymous identifiers; measurements and generated text are retained. The archive documents these metadata substitutions.

Record inventory. The archive contains five historical summary vectors (`grpo_summary_seed1.json`, `grpo_summary_seed2.json`, `grpo_summary_beta001.json`, `grpo_summary_beta0001.json`, and `sft_summary.json`); the base and trained response records (`base_1024_unified.json` and `trained_1024_unified.json`, with raw outputs, scores, and termination records); the fork records (`fork_eval_20260926.json`, with parsed answers and finish metadata); the code snapshots (`code/scorer_test.py`, `code/fork_eval.py`, `code/grpo_fixed_trajectory.py`, `code/math_grpo_eval.py`, and `code/math_grpo_train.py`); and the two data files (a separate 746-record population training file and the 500-problem evaluation file). Filenames under `results/` identify saved records, not verified training configurations. Twelve-character SHA-256 prefixes of the distributed evidence appear in Section G, and the replay script recomputes every derived quantity in this paper from these inputs.

B Selection and Denominators

Problem indices are zero-based indices in the distributed MATH-500 file. Applying $b_i = 0$ and $r_i > 0$ to GRPO-A selects

$$\{100, 104, 239, 326, 340, 355, 456, 466, 478, 483\}.$$

The corresponding trained credit counts at 128 draws are

$$(1, 2, 1, 41, 1, 1, 1, 1, 24, 2),$$

which sum to 75. The selection uses no claim that the underlying base probability is exactly zero. The subsequent evaluation has 1024 outputs per problem and model, or $10 \times 1024 = 10{,}240$ per arm. The historical 78-problem zero-base subset and the selected ten-problem set are different denominators.

For population summaries, rates are fractions. The sums G_{emp} and L_{emp} are across problems, while the reported mean change is $100(G_{\text{emp}} - L_{\text{emp}})/500$ percentage points. For example, GRPO-A has $G_{\text{emp}} = 7.234375$ and $L_{\text{emp}} = 3.8359375$, yielding 0.6796875 percentage points. BOSS is a ratio of these per-problem upward movements, not a count of newly solvable problems. The empirical statistic in this paper is not a posterior mean of ratios; no posterior credible interval is attached to it.

C Response Inspection Rubric

We use four separate questions when reading a response:

1. What string did the automatic extractor return, and why?
2. Does that string have the type required by the task?
3. Does the visible reasoning establish a relevant answer or intermediate result?
4. Does the output complete the requested answer before termination?

These questions intentionally do not collapse to a binary correctness label. A root pair and a scalar have different types. A correct derivation can precede unfinished checking. A token-limited response can contain an answer without a final-answer marker. Absence of `\boxed{` is therefore an observable formatting fact, not proof of mathematical failure.

The three worked counterexamples are base problem 104/sample 84, base problem 326/sample 34, and base problem 456/sample 217. Sample IDs are local to each problem and arm. The replay report includes each full response and a SHA-256 hash of its raw text. This makes the examples independently inspectable without relying on our prose excerpts.

Positive-only inspection cannot estimate corrected accuracy. Let C denote automatic credit and Y a separately defined semantic correctness label. An inspection of all $C = 1$ responses can estimate or enumerate $C = 1, Y = 1$ for the inspected rubric. It does not determine $C = 0, Y = 1$. The total number of semantic successes is the sum of both cells. Our uncredited problem-326 example motivates inspection of that second cell, but one example does not determine its size. The literal-string search is a way to find candidates, not a substitute for semantic annotation.

D Deterministic Scorer Probes

The original scorer is preserved so the stored labels remain replayable. Its behavior includes:

Probe	Actual output	Interpretation
Normalize `+42`	`4`	Leading-plus slicing defect
Compare `-21` to normalized `-2,1`	True	Scalar/root-pair collision
Compare `0.09` to normalized $9/100$	True	Decimal/fraction equivalence accepted

These are executions of supplied code on deterministic inputs, not generated experimental outcomes. The leading-plus defect is a code-level counterexample; no prevalence is inferred from it. The last-number fallback is additionally exercised on the actual saved responses, where it can select either an irrelevant coefficient or a later number after a correct derivation. A revised evaluator should preserve typed answers and return an explicit unresolved state when the response does not contain an answer under its declared rubric. We do not retrofit a new scorer and claim that the historical model comparison has thereby been validated.

E Fork Interpretation and Future Confirmation

The fork's supported and uncovered groups are defined using earlier base scores. They are not randomized difficulty strata. Matching a benchmark difficulty label would not equalize model-specific difficulty or baseline probability. Even after treatment identity is verified, a cross-group difference requires this observational qualification.

A confirmatory run should record the actual sequence of optimizer updates rather than reconstruct it from planned steps. For each fork it should preserve pre- and post-update weight hashes, the active objective and gradient normalization, training-example IDs, and loaded adapter hashes for both arms. It should use distinct inference adapter IDs and an explicit unload/reload or process-isolation check. Probe membership should be frozen before training and checked against the executed training sequence. Raw generations are needed for both false-positive and false-negative scoring audits. Independent seeds, with the seed as an inferential unit, are needed before estimating a general training effect. These are proposed checks, not additional completed experiments.

F Deterministic Threshold Sensitivity

The main-text cutoff $\tau = 0.05$ was exploratory. The following table changes only the cutoff in Equation 2, using the same saved rates. This is a deterministic sensitivity calculation, not a simulation, uncertainty interval, or test of acquisition. Per-problem contributions are supplied in `population_problem_contributions.csv` under `figure_data/`. All five comparisons reuse the same base vector, so they are not independent baseline replications. Figure 3 plots the identical values.

At $\tau = 0$, BOSS still attributes positive movement to an empirically zero-base subset, not a proven zero-probability subset. Its increase with the threshold is mechanical for fixed recorded vectors. We attach no capability interpretation or statistical significance to any cutoff.

Table 3: Empirical BOSS (%) across thresholds on the historical recorded-score vectors.

Record	0	0.01	0.02	0.05	0.10
GRPO-A	8.10	8.32	8.53	11.77	15.87
GRPO-B	9.04	9.28	9.77	13.19	17.58
GRPO-C	4.98	5.20	5.20	7.58	18.74
GRPO-D	5.69	6.02	6.10	8.00	12.53
SFT	5.88	5.98	5.98	6.87	14.44

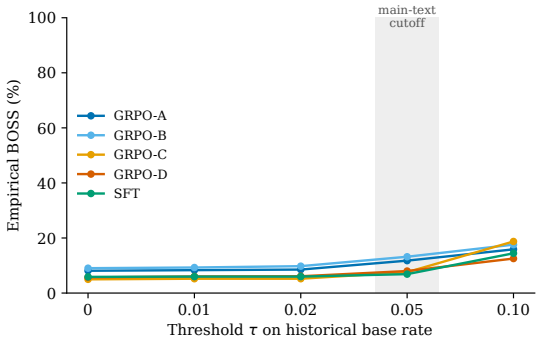

Figure 3: The deterministic threshold sweep from the table above, plotting empirical BOSS against τ for the five records; values are identical to the table. The shaded band marks the main-text cutoff $\tau = 0.05$. The increase with τ is mechanical for fixed recorded vectors.

G Frozen-Artifact Provenance

All objects below were captured for this self-audit on September 26, 2026. The date identifies the audit snapshot, not a recovered runtime date. Hash prefixes identify the distributed anonymized files; full SHA-256 values appear in `SHA256SUMS.json`. The anonymization ledger separately records original and distributed hashes. Only account/path metadata were anonymized; numerical measurements and generated text were preserved.

Table 4: Twelve-character SHA-256 prefixes of the distributed evidence.

Object	SHA-256 prefix
Historical GRPO-A	`c66729783884`
Historical GRPO-B	`af4dd79283f7`
Historical GRPO-C	`4a493a40102d`
Historical GRPO-D	`5873a82eeb31`
Historical SFT	`36ffb6956b61`
Base response records	`116b8faad643`
Trained response records	`52019704c16e`
Fork records	`eac21894cb49`
Scorer snapshot	`0e650d58f128`

Verified links. The replay program maps the historical GRPO-A vectors to the selected ten problem indices, maps each saved response to its parsed answer and credit under the scorer snapshot, and aggregates the fork file's recorded labels. These are direct computations on frozen inputs.

Unverified links. The snapshot does not establish the original generation timestamps, exact model/adapter weight identities, original tokenizer state, or a complete executed training sequence. Paths and configuration labels record intended identities without authenticating loaded states. Historical population vectors cannot be rescored without their original response text. Full fork text is absent. These missing links are retained as missing evidence rather than inferred from matching filenames.