Fixed-Budget Accounting Can Mislead Reinforcement Learning for Reasoning

Anonymous Author(s)
Affiliation
Address
email

Code: https://anonymous.4open.science/r/fixed-budget-rlvr-B29D

Abstract

Reinforcement learning with verifiable rewards (RLVR) has made reasoning performance increasingly depend on how compute is allocated. At a fixed budget, should training spend compute on more optimizer updates, more parallel rollouts, or deeper refinement passes? This question is easy to state but hard to answer because the planned token budget may not match the number of tokens actually generated. We study this problem for Low-Rank Adaptation (LoRA) with Group Relative Policy Optimization (GRPO) under a strict fixed-budget audit across several LLMs. Our audit compares the three allocation axes under scheduled-token accounting, instruments realized-token usage, and then tests any accounting-driven reversal with direct realized-budget reruns. This exposes an accounting inversion: refinement can look stronger after realized-token normalization because its rollouts often terminate early, while the controlled scheduled-budget comparison favors spending compute on optimizer updates. The direct reruns resolve the ambiguity: the optimizer-update allocation still achieves better benchmark performance while using fewer generated tokens. We also run a matched inference-time audit comparing refinement depth with parallel sampling on reasoning benchmarks. Depth helps in some settings, but loses or ties in others, and out-of-distribution (OOD) evaluations show no consistent depth advantage. Our results turn a potentially misleading fixed-budget comparison into a reporting rule: state the budget being fixed, treat realized-token normalizations as diagnostics, and verify apparent reversals with direct reruns before making training claims.

1 Introduction

Reinforcement learning with verifiable rewards (RLVR) has become a central tool for improving mathematical reasoning in open language models [10, 34, 24]. As these systems improve, the next question is no longer just whether RLVR helps, but where a fixed compute budget should be spent. A training run can buy more optimizer updates, more rollouts per prompt, or more refinement passes inside a single rollout. At inference time, the analogous choice is whether to spend extra compute on more independent samples or on deeper revision of one answer.

This paper studies a failure mode in that comparison: the same experiment can support different conclusions depending on the accounting unit. A scheduled-token budget counts the tokens a protocol permits; a realized-token budget counts the tokens the model actually emits before stopping. Depth refinement makes this distinction especially important because later refinement chunks often terminate early. We call the resulting phenomenon an *accounting inversion*: the allocation that looks worse under the controlled schedule can look better after normalizing by realized tokens.

We expose this inversion using Low-Rank Adaptation (LoRA) with Group Relative Policy Optimization (GRPO) across three DeepSeek-R1-Distill checkpoints. With the scheduled-token budget fixed

Submitted to 40th Conference on Neural Information Processing Systems (NeurIPS 2026). Do not distribute.

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.

at $3{,}276{,}800$ tokens, the optimizer-update allocation (Arm A) wins $11/12$ greedy MATH500 cells, while the d-axis allocation (Arm B) wins $0/12$. But because Arm B realizes fewer response tokens, a post-hoc realized-token normalization changes the apparent winner counts to A: $2/12$, B: $10/12$, C: $0/12$. This reversal is too large to ignore, but it is not a counterfactual training result: it assumes a linear return curve that the original grid did not run.

The risk is practical. RLVR papers often use "fixed budget" as a shorthand for several different controls: prompts, rollouts, token limits, generated tokens, or wall-clock compute. These controls coincide only when response lengths and early stopping behavior are similar. Depth refinement breaks that coincidence. One run can receive the same scheduled allowance as another run but emit fewer tokens because later passes stop after a short edit. Reporting only the scheduled budget can hide this efficiency difference; reporting only realized tokens can re-rank the same trained checkpoints without showing that the re-ranked allocation would train better if given that budget prospectively. Our audit keeps these two roles separate.

The main contribution is not a slogan about whether depth wins or loses. It is an audit protocol for what to do when the budget definition changes the answer. We run prospective Ashort/Blong reruns that test the realized-token reversal directly. These reruns still favor the optimizer-update allocation on MATH500 at 1.5B, 7B, and L8B while using only 0.83–$0.85\times$ as many realized response tokens as the d-axis reruns. We then use the same fixed-compute lens at inference time and find a bounded, model-dependent crossover: depth helps at $4\times$ compute for 1.5B and 7B, sampling wins at $8\times$ for 1.5B and L8B, and the remaining null cells are bounded with cluster-corrected equivalence tests.

Contributions.

1. We define a fixed-budget audit protocol for LoRA-GRPO that separates optimizer updates, parallel rollouts, and refinement depth while reporting both scheduled-token and realized-token accounting.
2. We identify a definition-sensitive accounting inversion: scheduled-token accounting favors optimizer updates, while realized-token normalization would appear to favor depth refinement on the same MATH500 grid.
3. We show that the inversion should be treated as a diagnostic rather than a training claim: prospective realized-budget reruns reject the tempting reversal and still favor the optimizer-update allocation.
4. We bound the scope of depth refinement with matched inference-time and out-of-distribution (OOD) audits, showing that depth has model- and multiplier-dependent benefits rather than a consistent advantage.

Take-away. A fixed-budget result should be reported as a verdict together with the knob that was fixed. When scheduled-token and realized-token accounting disagree, the next step is a prospective audit, not a stronger abstract claim.

2 Related Work

RLVR for reasoning models. RLVR follows earlier evidence that language models can benefit from spending tokens on intermediate reasoning: chain-of-thought prompting, zero-shot chain-of-thought, and self-bootstrapped rationales all improved reasoning without changing the test question itself [41, 20, 46]. RL from human or verifiable feedback has become a standard route for improving instruction following and reasoning [29, 10, 34, 24, 19, 44, 16]. Recent work also asks whether RLVR creates new reasoning behavior or mainly changes the sampling distribution of solutions [45], and 2025 mechanism studies trace gains to data efficiency, negative samples, and high-entropy decision tokens [39, 48, 37]. Our setting follows the verifiable math-reasoning line: the policy is trained with a binary correctness reward on boxed answers rather than with learned preference labels. We do not propose a new RL objective; we use LoRA-GRPO as a controlled substrate for a compute allocation audit.

Compute allocation in train-time and test-time reasoning. Compute-optimal scaling work already shows that a fixed budget is meaningful only after the cost model is specified [14, 32, 7]. Prior work on test-time compute shows that extra inference budget can be spent through sampling, search,

verification, or budget forcing [36, 38, 43, 28, 18]. Training-time RLVR has its own allocation knobs. Longer reasoning is not uniformly beneficial: inverse-scaling and overthinking studies find settings where extra reasoning length lowers accuracy [8, 47]. The IsoCompute line studies how on-policy RL should divide sampling compute across rollouts, problems per batch, and update steps [3]; training objectives can also interact with later pass@N search budgets, where pass@N is the fraction of problems solved by at least one of N sampled answers [2]. Depth-breadth RLVR work studies how exploration changes when difficulty and batch breadth are adapted [42]. Repeated-sampling scaling also has strong per-problem heterogeneity, so aggregate gains can hide whether a budget helps broadly or only on a hard tail [33]. Our audit treats updates, rollouts, and refinement depth as axes in the same budget decomposition and asks whether depth remains competitive when the scheduled token budget is held fixed.

Depth refinement and self-correction. Self-refinement methods ask a model to revise its own answer, either without additional training [26, 43] or through RL objectives designed for self-correction [21]. Process reward models provide another route to reasoning-time feedback [25]. Agent-memory and tool-feedback variants add external or linguistic feedback loops [35, 9], while negative self-correction results show that unguided revision can fail or even hurt reasoning [17]. These methods motivate depth as a plausible compute axis. The allocation question remains open: whether training budget spent on multiple refinement passes beats the same scheduled budget spent on more updates or more independent rollouts.

Budget accounting and evaluation methodology. Fixed-budget comparisons are only interpretable when the accounting unit is explicit. Scheduled-token accounting matches the configuration the experimenter sets; realized-token accounting measures the work actually emitted by the model. Rollout counts, token limits, and generated token totals are not interchangeable once response lengths differ. This paper shows that scheduled-token and realized-token views can disagree sharply for refinement training, then pairs the diagnostic with prospective reruns. This emphasis is aligned with prior calls for careful RL comparisons, fuller compute reporting, reproducibility checklists, and adequately powered statistical tests [12, 5, 4, 1, 30]. For inference and OOD comparisons we use cluster-by-problem bootstrap [6] and two one-sided tests (TOST) for equivalence [23]; the per-question depth analysis follows the flip-event lens of Zhou et al. [47].

3 Fixed-Budget Audit Protocol

LoRA-GRPO setup. We post-train DeepSeek-R1-Distill-Qwen-1.5B/7B and DeepSeek-R1-Distill-Llama-8B [10] with LoRA [15] (rank 32) via Group Relative Policy Optimization [34]. Training uses the `lighteval`-derived MATH split [13]; the reward is binary verifier-checked correctness of the answer inside `\boxed{}`. Evaluation uses vLLM [22] with matched prompts, decoding settings, and verifiers across all compared cells.

Three allocation axes. We factor the scheduled training budget as

$$\text{tokens} \;=\; M \cdot B_p \cdot n \cdot d \cdot \text{max_new}, \tag{1}$$

where M is the number of optimization steps, B_p is prompts per step, n is rollouts per prompt, d is refinement passes per rollout, and max_new is the maximum new tokens per pass. In the main grid, we keep B_p and max_new fixed and hold Equation 1 at 3,276,800 scheduled tokens. Figure 1 summarizes the resulting arms: allocating the fixed product toward optimizer steps M, refinement depth d, or rollout count n.

This setup makes the comparison an allocation audit rather than a schedule search. The prompts, per-pass token cap, and total scheduled budget are shared across arms, so differences in MATH500 performance are read as the effect of shifting budget among updates, self-revision, and independent samples. These choices are often mixed together when a paper says that it spent more compute on reasoning.

Scheduled and realized budgets. The scheduled-token budget is the control variable in the training configuration: it counts the tokens that would be generated if every pass used the full max_new allowance. The realized-token budget counts the tokens actually emitted before `<eos>`. Each run records realized response tokens and realized fraction in a structured training summary. We use

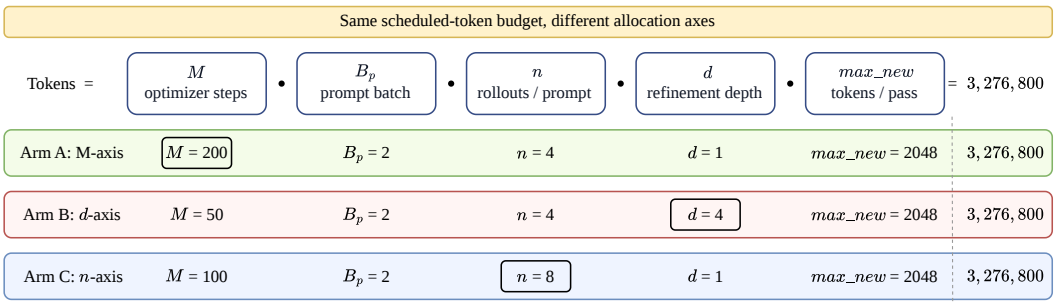


Figure 1: **The fixed-token-budget decomposition and our three allocation arms.** The training-token budget factorizes as $\text{tokens} = M \cdot B_p \cdot n \cdot d \cdot \text{max_new}$. All three arms (A, B, C) hold the *scheduled-token* budget at 3,276,800; they differ only in how they allocate it across optimization steps M, rollouts per prompt n, and refinement passes per rollout d. **Take-away:** this lets us isolate the d-axis (Arm B) against the M-axis (Arm A) and n-axis (Arm C) at a single fixed budget.

pass@1 divided by realized fraction as a sensitivity diagnostic, not as an achievable accuracy estimate, because it assumes a linear return curve.

Formally, if r_j is the number of response tokens emitted by generation call j, the realized fraction is

$$f_{\text{real}} = \frac{\sum_j r_j}{M \cdot B_p \cdot n \cdot d \cdot \text{max_new}}. \tag{2}$$

The scheduled score is the ordinary evaluation accuracy of the trained checkpoint. The realized-token diagnostic rescales this score by f_{real} to ask how sensitive the ranking is to unused token allowance. By design, the rescale is a screening diagnostic rather than a rerun: it flags rankings that depend on the accounting unit, without estimating the accuracy of a reallocated training run or modeling diminishing returns.

Prospective rerun protocol. When scheduled-token and realized-token accounting disagree, we test the disagreement directly with matched Ashort/Blong reruns. Ashort is an optimizer-update allocation with a shorter planned budget; Blong is a d-axis allocation trained under the corresponding realized-budget audit. Matched seeds are compared on MATH500 in greedy and sampled modes, and we report both downstream accuracy deltas and realized-token ratios. The prospective comparison is the evidence used to decide whether the realized-token diagnostic should become a training recommendation.

The rerun protocol is intentionally narrow. Its target is the specific reversal suggested by the diagnostic, not a full accuracy curve over every possible realized-token budget: whether a d-axis allocation remains preferable once the optimizer-update allocation is shortened to the realized-token scale. Using matched seeds removes one source of training variance, while evaluating the same checkpoints in greedy and sampled modes checks that the verdict is not an artifact of a single decoder. A reversal is promoted only if it survives this direct comparison.

Statistical testing. For inference-time and OOD comparisons, the cluster is the problem rather than the sampled completion. We resample clusters 10,000 times, report cluster-by-problem confidence intervals, and use TOST [23] at margins ± 3, ± 4, and ± 5 pp for null cells. Full hyperparameters, seed lists, and bootstrap definitions are in Appendix A–B.

Clustering by problem matches the dependency structure of pass@N evaluation. Multiple samples from the same problem are not independent evidence that the allocation works on more tasks; they are repeated attempts on one task. The bootstrap resamples problems, carries all paired completions for each resampled problem, and recomputes the paired depth-minus-sampling or arm-minus-A delta. For null cells, TOST is used as a bounded-null statement rather than as a claim of exact equality. This matters for the inference audit, where several cells are neither significant wins nor practically large losses.

4 Experiments

The experiments follow the audit decision tree in Section 3. We first ask what wins under the scheduled budget, because that is the budget fixed by the training protocol. We then ask whether the verdict changes after realized-token instrumentation. If it does, we run a prospective comparison instead of treating the rescaled table as a new result. The inference-time and OOD evaluations are separate boundary tests: they check whether the training-axis conclusion would be contradicted by a clear advantage for depth at evaluation time or after leaving the math-training distribution.

Main 36-cell MATH500 audit. The primary training-time audit evaluates $\{1.5\text{B}, 7\text{B}, \text{L}8\text{B}\} \times \{\text{Arm A}, \text{Arm B}, \text{Arm C}\} \times d_{\text{eval}} \in \{1, 2, 4, 8\}$ on full MATH500 [25]. All cells use greedy decoding with $\text{max_new}{=}2048$ and the same verifier. The comparison asks which allocation arm is best when the scheduled training-token budget is held fixed.

We use greedy MATH500 for the main result grid because it is the cleanest readout of the trained policy under a single answer attempt. Sampling is useful for estimating pass@N, but it introduces an inference-time breadth knob that would be confounded with the training-time breadth arm. Greedy evaluation keeps the main grid tied to the checkpoint produced by each allocation. The sampled MATH500 and pass@4 results are used later as checks on whether the scheduled verdict is brittle to decoding mode.

Realized-token diagnostic. For each trained arm we aggregate realized-fraction training summaries and compute two diagnostic views: the scheduled-token pass@1 score and the corresponding per-realized-token efficiency. The diagnostic is intentionally post-hoc: it reveals whether the fixed-budget conclusion is sensitive to early termination, not what a separately trained model would score under a different realized budget.

Prospective Ashort/Blong reruns. To test the realized-token disagreement directly, we run matched Ashort/Blong reruns at all three scales. The matched sets contain 3 seeds at 1.5B, 9 at 7B, and 8 at L8B. We evaluate each matched pair on MATH500 in greedy and sampled modes, average over evaluation depths, and record the ratio of realized response tokens consumed by Ashort relative to Blong.

The comparison is stricter than choosing the better-looking row from the rescaled diagnostic. Ashort must beat Blong after both checkpoints are trained under the rerun protocol, not after an algebraic normalization of earlier checkpoints. The matched-pair design also makes the sign of the delta easier to interpret: each seed contributes one Ashort-minus-Blong contrast after the same evaluation sweep.

Inference-time depth versus parallel sampling. For each checkpoint, we evaluate the $d{=}4$ trained LoRA on AIME$\{24, 25, 26\}$ [27] and HMMT$\{24, 25, 26\}$ [11] with three sampling seeds. We compare $(d{=}4, n{=}1)$ against $(d{=}1, n{=}4)$ at $4\times$ compute and $(d{=}8, n{=}1)$ against $(d{=}1, n{=}8)$ at $8\times$ compute. The metric is paired pass@1 difference, depth minus sampling, with cluster-by-problem bootstrap intervals. The final 7B/L8B rows use the on-budget 2048-token LoRAs; the 1.5B $8\times$ row has fewer paired problems than the other grand-pool rows, as documented in Appendix A.

OOD evaluation. Math-only training may over-specialize differently across allocation arms. For that boundary check, we rerun all checkpoint-by-arm cells under three sampling seeds with $n{=}4$ samples on GPQA-Diamond [31] and a 500-item stratified subset of MMLU-Pro [40]. We report cluster-corrected pass@4 deltas for Arm B and Arm C relative to Arm A.

The OOD tests are not meant to establish broad transfer. They serve a narrower role in the audit. If the d-axis training arm lost on MATH500 but gained consistently on GPQA-Diamond or MMLU-Pro, the scheduled-budget verdict would be too narrow to report without qualification. Conversely, a tie or loss outside the training distribution supports a bounded claim: in this experimental regime, depth refinement does not recover through a clear transfer advantage.

5 Results

Scheduled-budget comparison favors optimizer updates. Under the budget fixed by the training protocol, the optimizer-update axis is the strongest allocation. Figure 2 reports the MATH500 greedy

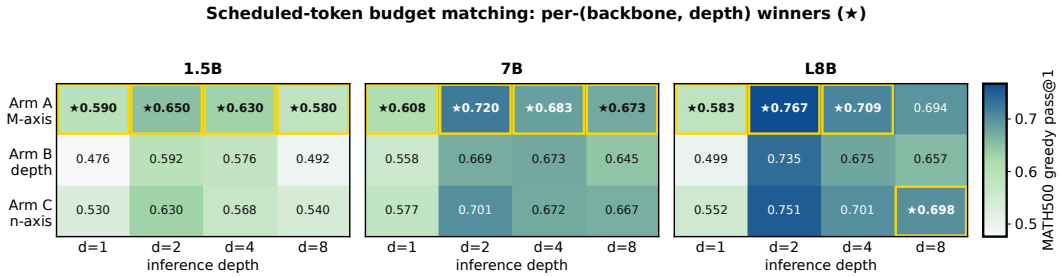


Figure 2: **Scheduled-token budget-matched verdict, MATH500 greedy pass@1.** Three checkpoints, three allocation arms, four inference depths ($d \in \{1, 2, 4, 8\}$). Stars mark the per-(checkpoint, depth) winner; the M-axis (Arm A) wins $11/12$ cells, the n-axis (Arm C) wins $1/12$, and the d-axis (Arm B) wins $0/12$. **Take-away:** under the scheduled-token budget the d-axis is never the per-cell best.

pass@1 grid. Arm A wins $11/12$ (checkpoint, d_{eval}) cells, Arm C wins the remaining L8B $d{=}8$ cell, and Arm B wins none. The 12-cell means show the same ordering: $\bar{\text{A}}{=}0.657$, $\bar{\text{C}}{=}0.633$, $\bar{\text{B}}{=}0.604$. The completed high-scale three-seed scan preserves the A-vs-B ordering at every 7B/L8B cell and every seed (Appendix Table 3).

The result is stronger than a single best cell. Arm A's advantage appears across checkpoints and across evaluation depths, including depths that give Arm B additional chances to revise at test time. The lone Arm C win at L8B $d{=}8$ does not change the training-axis ordering: the d-axis arm is never the scheduled-budget winner, and the arm with more optimizer updates is the only allocation that wins across nearly the full grid. That is the verdict that a scheduled-token audit would report.

Realized-token normalization induces an accounting inversion. Scheduled-token accounting leaves one important diagnostic open: depth refinement uses less of its token allowance. Averaged over available training summaries, Arm B realizes 0.720 of scheduled response tokens at 1.5B, 0.691 at 7B, and 0.640 at L8B, compared with Arm A's 0.842, 0.847, and 0.782. Normalizing pass@1 by realized fraction changes the apparent winner counts from A: $11/12$, B: $0/12$, C: $1/12$ to A: $2/12$, B: $10/12$, C: $0/12$. We call this the accounting inversion. It is a real sensitivity of the comparison, but the normalized entries are rates, not achievable accuracies; several exceed 1.0 in the full table (Appendix Table 4).

The point of this table is disclosure, not counterfactual training. A normalized score above 1.0 cannot be a pass@1 accuracy; it shows that the denominator changed more than the evaluation score. The diagnostic asks whether the ranking is sensitive to charging unused scheduled tokens against each allocation. The training question is left for direct reruns: how would the allocation perform after the schedule is redesigned around a realized-token budget?

Prospective reruns reject the tempting reversal. The prospective realized-budget reruns put the post-hoc normalization to a direct training test. The reversal does not survive. Ashort beats Blong on MATH500 at all three scales in both greedy and sampled evaluation, while consuming only 0.83–$0.85\times$ as many realized response tokens as Blong. Across all benchmark/mode/depth cell means, Ashort also wins more cells than Blong at every scale, but the main claim is the direct in-domain MATH500 comparison summarized in Table 1 and Figure 3.

The direction matters more than the exact gain. If realized-token efficiency were enough to make d-axis training preferable, Blong should close the gap once Ashort is shortened to the same audit scale. Instead, Ashort remains ahead. The result separates two facts that the post-hoc table merges: Arm B emits fewer tokens per scheduled budget, and Arm A still produces the better MATH500 checkpoint when the comparison is rerun directly. The prospective test resolves the accounting ambiguity without denying that early termination is a real property of refinement training.

Inference-time depth has a model-dependent crossover. The training-axis result leaves room for a narrower inference-time role for depth. At inference time, depth is beneficial in some fixed-compute cells and harmful in others. On the AIME+HMMT 2024+2025+2026 grand pool, depth beats parallel sampling at $4\times$ compute for 1.5B by $+4.7$ pp (cluster CI $[+1.8, +7.8]$, $p{=}0.001$) and for 7B by $+4.2$ pp (CI $[+1.1, +7.5]$, $p{=}0.011$). At $8\times$, sampling beats depth for 1.5B (-5.2 pp) and L8B

Table 1: Prospective realized-budget reruns. Positive deltas mean Ashort−Blong on MATH500; uncertainty is the standard deviation over matched training seeds after averaging over evaluation depths.

Checkpoint	matched seeds	greedy Δ	sample Δ	A/B realized tokens
1.5B	3	$+6.7\pm2.6$ pp	$+4.5\pm1.5$ pp	$0.85\times$
7B	9	$+2.7\pm0.6$ pp	$+1.7\pm0.9$ pp	$0.83\times$
L8B	8	$+3.3\pm1.2$ pp	$+3.1\pm1.4$ pp	$0.85\times$

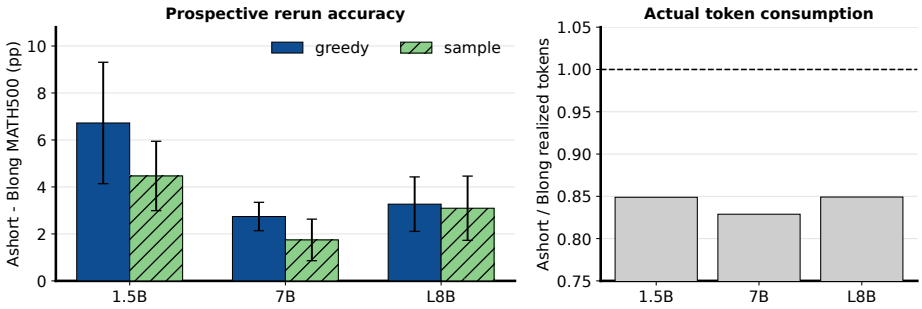


Figure 3: **Accounting inversion is a diagnostic, not a training recommendation.** Left: matched-seed MATH500 Ashort–Blong deltas averaged over evaluation depths, with seed-level standard-deviation bars. Ashort wins on 1.5B ($+6.7$ pp greedy, $+4.5$ pp sample), 7B ($+2.7$ pp, $+1.7$ pp), and L8B ($+3.3$ pp, $+3.1$ pp). Right: Ashort uses only 0.83–$0.85\times$ as many realized response tokens as Blong in the matched training runs. **Take-away:** the per-realized-token rescale is a useful audit warning, but direct reruns still favor the optimization-step axis on in-domain MATH500.

(-4.6 pp), while 7B is a TOST-bounded null at ±4 and ±5 pp under cluster correction. The crossover is model- and multiplier-dependent rather than a stable advantage for either axis; Figure 4 summarizes the paired inference deltas.

The inference result also guards against an overbroad reading of the training grid. The training arms ask where to spend compute while updating the policy; the inference comparison asks how to spend compute after a checkpoint is fixed. Those are different interventions. A model can benefit from revising an answer at test time even if training with many refinement passes is not the best use of the scheduled training budget. The crossover in Figure 4 sets the scope of the claim.

OOD tests show no consistent depth advantage. On GPQA-Diamond and MMLU-Pro, the d-axis training allocation does not show a consistent out-of-distribution advantage. At 1.5B, Arm B is statistically worse than Arm A on both GPQA (-8.6 pp, CI $[-12.6, -4.7]$) and MMLU-Pro (-4.7 pp, CI $[-7.4, -2.1]$). At 7B and L8B, all B$-$A and C$-$A cells are tied under cluster correction. The OOD evidence is best read as a boundary on the training-axis claim: depth is not rescued by a clear transfer benefit.

The 1.5B losses are the clearest OOD signal, but the larger models are just as relevant for the paper's scope. They show ties rather than hidden wins for the d-axis arm. The pattern supports the reporting rule used throughout the paper: the scheduled-budget MATH500 verdict should be stated with its task and checkpoint scope, and the OOD audit should be reported as a failure to find a consistent transfer advantage for depth.

6 Discussion

Reporting rule for RLVR compute audits. The scheduled-token and realized-token views answer different questions. Scheduled-token matching asks where to allocate the budget the training protocol is allowed to spend. Realized-token accounting asks how much work the model actually emitted. In refinement training these two can diverge because later passes often terminate early. The safe reporting unit is a verdict-and-knob: state the budget being fixed, report the verdict under that budget, and treat any realized-token reversal as a diagnostic that requires a prospective check.

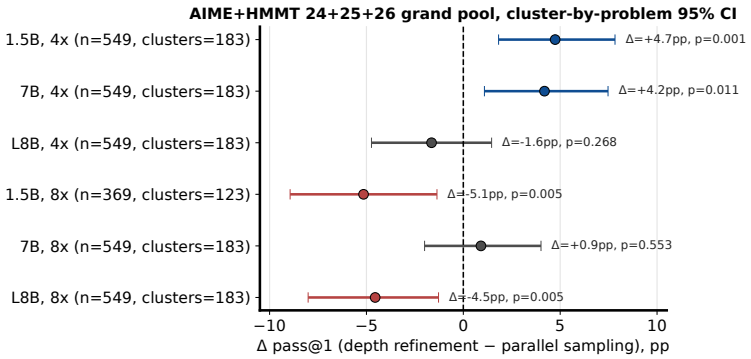


Figure 4: **Compute-matched inference: depth refinement vs. parallel sampling.** Paired cluster bootstrap on the AIME+HMMT 24+25+26 grand pool (3 sampling seeds; clusters by problem). At $4\times$ compute, depth refinement beats sampling on the 1.5B trained model by $+4.7$ pp (p=0.001) and on 7B by $+4.2$ pp (p=0.011); L8B is statistically null at $4\times$. At $8\times$ compute, sampling beats depth on 1.5B and L8B; 7B remains null and is bounded-equivalent at ±4 pp and ±5 pp. **Take-away:** the depth-vs-sampling crossover is model- and multiplier-dependent, not a universal advantage for either axis.

For this paper, the rule would read as follows. Under a fixed scheduled-token training budget, the optimizer-update arm is best on the MATH500 LoRA-GRPO grid. After realized-token normalization, the same grid raises a depth-favoring diagnostic because the refinement arm leaves more of its scheduled allowance unused. The diagnostic is then tested with matched realized-budget reruns, and those reruns favor the optimizer-update allocation. That phrasing keeps the three statements in their proper order. It avoids both mistakes: hiding the early-stopping efficiency of refinement, or turning a post-hoc rate into a claim about which training schedule should be chosen.

Why the rule is conservative. The rule deliberately favors direct evidence over a more convenient normalization. That makes the conclusion less sweeping, but it also makes it harder to misread. A realized-token rate is useful because it exposes wasted allowance and early stopping. It becomes unsafe when it is treated as if the unused allowance could be converted into accuracy at the same slope. The prospective rerun is the minimal check that avoids this assumption. If a future method shows a realized-token reversal and also wins after matched reruns, the same rule would support the d-axis allocation. Our result is not a preference for scheduled accounting in all cases. It is a preference for matching the claim to the experiment that was actually run.

Implications for depth refinement. The evidence narrows the role of depth refinement rather than dismissing it. In our scheduled-token LoRA-GRPO grid, depth is not the best training-axis allocation, and its realized-token efficiency is insufficient grounds for a training recommendation without direct reruns. At inference time, depth can help at $4\times$ compute for some trained models, but the benefit disappears or reverses at other multipliers and scales. The pattern is consistent with depth sometimes repairing local errors and sometimes revising within a narrow attractor while parallel samples explore more diverse solutions. That mechanism remains a hypothesis. The evidence here is comparative and accounting-focused.

Scope of the claims. The realized-token inversion is a rate diagnostic, not an achievable Arm B accuracy table. The scheduled-token result is tied to this LoRA-GRPO setting. Extrapolation to all self-correction methods, search procedures, or larger models would go beyond the evidence. The inference crossover is a matched-compute observation, not a universal depth-vs-sampling law. The OOD checks are boundary tests on two benchmarks, not a broad claim about transfer, coding, or instruction following.

Auxiliary diagnostics. Two secondary analyses support cautious interpretation and remain in the appendix. The per-question flip-event analysis shows that aggregate accuracy can hide substantial churn across refinement depths (Appendix H). The generation-budget sweep shows that absolute MATH500 scores are sensitive to `max_new`, so cross-paper comparisons must match decoding budget before attributing gaps to training quality (Appendix D).

7 Limitations

Realized-budget coverage. The realized-token normalization is a post-hoc sensitivity analysis, and the prospective Ashort/Blong rerun is a targeted audit rather than a fully crossed realized-token return curve. The scheduled-token MATH500 scan has three training seeds at high scale, but the realized-fraction instrument has two high-scale summaries per arm. This is enough to expose the accounting disagreement, but not enough to characterize the entire accuracy-vs-realized token frontier.

Inference coverage. The final 7B/L8B inference rows use on-budget 2048-token LoRAs and cover the full 549 paired-problem grand pool at both $4\times$ and $8\times$. The older 1.5B $8\times$ row has 369 paired problems. We report the paired count for every row and use cluster correction, but the row-count asymmetry should be kept in mind when comparing effect sizes across checkpoints.

Checkpoint and task scope. The audit covers three DeepSeek-R1-Distill checkpoints and math-centered training. The OOD checks cover GPQA-Diamond and MMLU-Pro; they do not cover coding, tool use, instruction following, or long-context reasoning. The depth-at-$4\times$ inference win holds for 1.5B and 7B but not L8B, so it should be written as a bounded regime rather than as a scale-invariant property.

Cost-model scope. Tokens are the accounting unit throughout the main audit. They are the right unit for comparing scheduled allowances and realized generation, but they do not capture every cost a practitioner pays. Refinement passes can change wall time, KV-cache reuse, batching efficiency, and memory pressure in ways that a token count does not show. We avoid making a throughput claim. A deployment study should report latency and hardware utilization alongside tokens before turning this audit into a serving recommendation.

Metric scope. The main metrics are verifier-checked answer accuracy, pass@1, and pass@4. These metrics leave rationale quality, chain-of-thought faithfulness, and safety outside the evaluation. This choice is deliberate for math RLVR because the reward and evaluation target are boxed-answer correctness. It also limits the claim. A setting that rewards explanations, tool traces, or human preferences could value refinement for reasons that are invisible to our verifier.

Mechanism scope. Our evidence is comparative and accounting-focused. It leaves the mechanisms behind early termination, question-level flips, and the $8\times$ depth-sampling crossover for separate study. The mechanistic hypotheses in the discussion are interpretations to test, not claims established by this paper.

8 Conclusion

Within this LoRA-GRPO audit, a fixed scheduled training budget favors the update-axis allocation. The d-axis arm gives a different kind of benefit: it can be more economical in emitted tokens because refinement often uses less than its scheduled allowance. These two observations should not be collapsed into a single winner. One asks which schedule produces the better checkpoint under a fixed allowance. The other describes how much of that allowance a completed run actually used.

The result is a reporting rule rather than a new preference for one axis in all settings. Fixed-budget RLVR studies should state which budget is controlled, report realized-token use as a diagnostic, and check apparent reversals with direct reruns before turning them into training claims. The inference and out-of-distribution audits set the boundary: depth can help in some settings, but it is not a general substitute for more updates or more samples. Future work should move from auditing fixed arms to designing schedules that plan with both budgets from the start, across broader model families, tasks, and cost measures beyond tokens.

References

[1] Dallas Card, Peter Henderson, Urvashi Khandelwal, Robin Jia, Kyle Mahowald, and Dan Jurafsky. With little power comes great responsibility. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9263–9274, Online, 2020. Association for Computational Linguistics.

[2] Feng Chen, Allan Raventós, Nan Cheng, Surya Ganguli, and Shaul Druckmann. Rethinking fine-tuning when scaling test-time compute: Limiting confidence improves mathematical reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[3] Zhoujun Cheng, Yutao Xie, Yuxiao Qu, Amrith Setlur, Shibo Hao, Varad Pimpalkhute, Tongtong Liang, Feng Yao, Zhengzhong Liu, Eric Xing, Virginia Smith, Ruslan Salakhutdinov, Zhiting Hu, Taylor Killian, and Aviral Kumar. Isocompute playbook: Optimally scaling sampling compute for LLM RL, 2026. Presented at the ICLR 2026 Workshop on Scaling Post-Training for LLMs (SPOT).

[4] Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. Show your work: Improved reporting of experimental results. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2185–2194, Hong Kong, China, 2019. Association for Computational Linguistics.

[5] Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia, 2018. Association for Computational Linguistics.

[6] Bradley Efron and Robert Tibshirani. Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. *Statistical Science*, 1(1):54–75, 1986.

[7] Preston Fu, Oleh Rybkin, Zhiyuan Zhou, Michal Nauman, Pieter Abbeel, Sergey Levine, and Aviral Kumar. Compute-optimal scaling for value-based deep RL. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[8] Aryo Pradipta Gema, Alexander Hägele, Runjin Chen, Andy Arditi, Jacob Goldman-Wetzler, Kit Fraser-Taliente, Henry Sleight, Linda Petrini, Julian Michael, Beatrice Alex, Pasquale Minervini, Yanda Chen, Joe Benton, and Ethan Perez. Inverse scaling in test-time compute. *Transactions on Machine Learning Research*, 2025. Also available as arXiv:2507.14417.

[9] Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *International Conference on Learning Representations (ICLR)*, 2024.

[10] Daya Guo, Dejian Yang, Haowei Zhang, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025.

[11] Harvard-MIT Mathematics Tournament. HMMT problems and results archive. https://www.hmmt.org/www/archive/problems, 2026.

[12] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, pages 3207–3214, 2018.

[13] 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. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2021.

[14] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. An empirical analysis of compute-optimal large language model training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[15] Edward 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 (ICLR)*, 2022.

[16] Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-Reasoner-Zero: An open source approach to scaling up reinforcement learning on the base model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. Poster.

[17] Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *International Conference on Learning Representations (ICLR)*, 2024.

[18] Yuichi Inoue, Kou Misaki, Yuki Imajuku, So Kuroki, Taishi Nakamura, and Takuya Akiba. Wider or deeper? scaling LLM inference-time compute with adaptive branching tree search. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[19] Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, et al. Kimi k1.5: Scaling reinforcement learning with LLMs, 2025. arXiv:2501.12599.

[20] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[21] Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D. Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal Behbahani, and Aleksandra Faust. Training language models to self-correct via reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2025.

[22] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Symposium on Operating Systems Principles (SOSP)*, 2023.

[23] Daniël Lakens. Equivalence tests: A practical primer for t tests, correlations, and meta-analyses. *Social Psychological and Personality Science*, 8(4):355–362, 2017.

[24] Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James Validad Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Christopher Wilhelm, Luca Soldaini, Noah A. Smith, et al. Tülu 3: Pushing frontiers in open language model post-training. In *Conference on Language Modeling (COLM)*, 2025.

[25] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations (ICLR)*, 2024.

[26] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[27] Mathematical Association of America. MAA invitational competitions. https://maa.org/maa-invitational-competitions/, 2026.

[28] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 20275–20321. Association for Computational Linguistics, 2025.

[29] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[30] Joelle Pineau, Philippe Vincent-Lamarre, Koustuv Sinha, Vincent Larivière, Alina Beygelzimer, Florence d'Alché-Buc, Emily Fox, and Hugo Larochelle. Improving reproducibility in machine learning research: A report from the NeurIPS 2019 reproducibility program. *Journal of Machine Learning Research*, 22(164):1–20, 2021.

[31] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof Q&A benchmark. In *Conference on Language Modeling (COLM)*, 2024.

[32] Nikhil Sardana, Jacob Portes, Sasha Doubov, and Jonathan Frankle. Beyond chinchilla-optimal: Accounting for inference in language model scaling laws. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 43445–43460. PMLR, 2024.

[33] Rylan Schaeffer, Joshua Kazdan, John Hughes, Jordan Juravsky, Sara Price, Aengus Lynch, Erik Jones, Robert Kirk, Azalia Mirhoseini, and Sanmi Koyejo. How do large language monkeys get their power (Laws)? In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 53132–53176. PMLR, 2025.

[34] 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.

[35] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[36] 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 *International Conference on Learning Representations (ICLR)*, 2025.

[37] Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for LLM reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. Poster.

[38] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

[39] Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Liyuan Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, Weizhu Chen, Shuohang Wang, Simon Shaolei Du, and Yelong Shen. Reinforcement learning for reasoning in large language models with one training example. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[40] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2024.

[41] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[42] Zhicheng Yang, Zhijiang Guo, Yinya Huang, Yongxin Wang, Dongchun Xie, Hanhui Li, Yiwei Wang, Xiaodan Liang, and Jing Tang. Depth-breadth synergy in RLVR: Unlocking LLM reasoning gains with adaptive exploration, 2025. arXiv:2508.13755.

[43] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[44] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, Lingjun Liu, Xin Liu, Haibin Lin, et al. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[45] 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 *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[46] Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[47] Shu Zhou, Rui Ling, Junan Chen, Xin Wang, Tao Fan, and Hao Wang. When more thinking hurts: Overthinking in LLM test-time compute scaling, 2026. arXiv:2604.10739.

[48] Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. The surprising effectiveness of negative reinforcement in LLM reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Appendix

Contents

A Hyperparameters and Training Details **15**

B Statistical Methodology **16**

C Full Result Tables **16**

C.1 36-cell MATH500 scheduled-token grid 16

C.2 Realized-token rescale diagnostic 16

C.3 Prospective realized-budget artifacts 17

C.4 Inference-iso grand pool 17

C.5 TOST equivalence 17

C.6 OOD pass@4 multi-seed results 17

D Generation-Budget Sanity Check **18**

E Realized-Fraction Instrumentation **18**

F Off-Budget Runs and Remediation **19**

G Sampled-MATH500 Sub-experiment **20**

H Per-Question Flip Events **20**

I Negative and Deferred Checks **21**

J Reproducibility **22**

A Hyperparameters and Training Details

Checkpoints. Three open distill models from the DeepSeek-R1 family [10], all under the `deepseek-ai/` Hugging Face prefix: `DeepSeek-R1-Distill-Qwen-1.5B`, `DeepSeek-R1-Distill-Qwen-7B`, and `DeepSeek-R1-Distill-Llama-8B`. We refer to these throughout as 1.5B, 7B, and L8B respectively. The L8B row is a Llama-architecture distill, not a Qwen-architecture distill; we keep it as a third scale because it gives a $\sim 2.6\times$ jump over 1.5B and a different base architecture for the OOD comparison.

LoRA. We adapt every checkpoint with LoRA [15]: rank r=32, α=64, dropout 0.0, target modules $\{q_{\text{proj}}, k_{\text{proj}}, v_{\text{proj}}, o_{\text{proj}}, \text{up_proj}, \text{gate_proj}, \text{down_proj}\}$. The base parameters are frozen; only the LoRA adapters are trained. Adapter merging is *not* performed: at evaluation we serve the base via vLLM and load the LoRA dynamically.

GRPO objective. We use Group Relative Policy Optimization [34] with group size g=n (the number of rollouts per prompt), advantage normalization within the group, and no critic. Reward is binary verifier-checked correctness on the boxed answer (1 if the verifier accepts the predicted answer, 0 otherwise). The verifier parses LaTeX, normalizes fraction/parenthesis forms, and falls back to symbolic equivalence for non-numeric answers.

Optimizer. AdamW with learning rate 1×10^{-5}, β_1=0.9, β_2=0.95, weight decay 0.0, gradient accumulation steps 1, linear warmup over the first 5% of optimization steps followed by cosine decay, and gradient clipping at $\|\nabla\|$=1.0.

Training data. The `lighteval`-derived MATH split [13]: a subset of `open-r1/MATH-lighteval` with the AIME competition years held out for the inference-iso evaluation pool (Section 3). Each prompt is wrapped in the chat template of its base model (the DeepSeek-R1-Distill chat template for all three checkpoints); we ask the model to think step-by-step and put its final answer in `\boxed{}`.

Refinement-prompt format (Arm B at d>1). For Arm B with d=4, each rollout is a sequence of four `(prompt + answer)` chunks separated by an explicit refinement turn (*"Please review your previous answer and revise if you find an error."*); the policy gradient is computed across the full concatenation, giving the model credit for the final correctness of the rollout.

Eval (in-distribution greedy MATH500). The in-distribution evaluator uses max_new=2048, temperature T=0, top_p=1.0, and $d_{\text{eval}} \in \{1, 2, 4, 8\}$ inference-depth passes per problem. The verifier and chat template match training. Each LoRA-arm cell is evaluated on the full 500-problem MATH500 set [25]; the trained-LoRA scan records per-problem correctness at every d_{eval}.

Eval (inference-iso AIME+HMMT grand pool). T=0.6, top_p=0.95, max_new=4096, sampling seeds $\{42, 43, 44\}$. Five $(d_{\text{eval}}, n_{\text{eval}})$ configurations: $(1, 1)$ at $1\times$ compute, $(1, 4)$ and $(4, 1)$ at $4\times$, $(1, 8)$ and $(8, 1)$ at $8\times$. The grand pool merges AIME$\{2024, 2025, 2026\}$ [27] and HMMT$\{2024, 2025, 2026\}$ [11] for a total of 549 paired problems at $4\times$ for all checkpoints. At $8\times$, 1.5B has 369 paired problems because the older HMMT2024 + HMMT2025 pool does not have an $8\times$ run; the on-budget 7B/L8B reruns cover the full 549 paired problems at $8\times$.

Eval (OOD pass@4). GPQA-Diamond [31] (the non-gated mirror `fingertap/GPQA-Diamond`; n=198 problems) and a stratified 500-problem subset of MMLU-Pro [40]. T=0.6, top_p=0.95, max_new=4096, n_{samples}=4, sampling seeds $\{42, 43, 44\}$. Both benchmarks use a multiple-choice question (MCQ) format; the model is asked to put a letter answer in `\boxed{}`, and the reward/metric is letter-match.

Hardware and compute. Experiments used NVIDIA H100 (94 GB) and H200 (141 GB) GPUs. Approximate per-run wall times vary by arm: 1.5B trainings took about 2.0–4.0 GPU-h, 7B about 2.7–4.7 GPU-h, and L8B about 3.1–5.9 GPU-h. Greedy MATH500 evaluation takes 5–30 GPU-min per cell. Total compute is order 10^2 GPU-h across training, scans, inference-iso, and OOD pass@4 runs; we do not compute an exact total because runs were re-submitted and re-evaluated multiple times during the on-budget remediation cycle described in Appendix F.

Random seeds. Training seeds: $\{42, 43, 44, 45\}$ (1.5B has 3 seeds per arm out of $\{43, 44, 45\}$; the $\{42\}$ sweep is the original baseline; 7B/L8B draw from $\{42, 43, 44\}$). Sampling seeds at evaluation: $\{42, 43, 44\}$ for AIME+HMMT and OOD pass@4. The 14B-distill flip-event analysis uses seed 42 at the trained-LoRA scan.

B Statistical Methodology

Cluster-by-problem bootstrap. The unit of resampling is the cluster, taken as $(\text{bench}, \text{problem})$ for the AIME+HMMT inference-iso pool and as the problem alone for the OOD pass@4 analysis (the latter has a single benchmark per call). Within each cluster we average across the three sampling seeds $\{42, 43, 44\}$, producing a single cluster-mean difference Δ_c between the two compared cells. We then sample n_{clusters} clusters with replacement, B=10,000 bootstrap iterations, and report the 95% CI as the 2.5%/97.5% quantiles of the bootstrap cluster-mean distribution. The two-sided p-value is $2 \cdot \min(P(\Delta_b < 0), P(\Delta_b > 0))$ over the B bootstrap means.

Naive paired bootstrap (for completeness). For comparison only, we also report a naive paired bootstrap that resamples $(\text{problem}, \text{seed})$ pairs as if independent. This over-narrows the CI when seed correlation is non-trivial, as it is for sampled generations of the same problem. The cluster CI is the authoritative report; the naive CI is shown alongside the cluster-corrected CI in the appendix tables.

TOST equivalence. Following the standard two-one-sided-tests procedure [23], we declare equivalence at margin δ and significance α iff the $100 \cdot (1 - 2\alpha)\%$ confidence interval of the effect lies strictly inside $(-\delta, +\delta)$. We use α=0.05 and the 90% cluster CI; we evaluate three margins $\delta \in \{3, 4, 5\}$ pp without preregistration (the values were chosen during the audit cycle). The cluster-corrected 90% CI is the authoritative report.

Linear-extrapolation rescale (sensitivity check, not a counterfactual). For each cell with measured pass@1 and realized fraction $f_{\text{real}} \in [0, 1]$, we report two derived quantities:

$$\begin{aligned} \text{pass1}_{\text{realized}} &= \text{pass1} \;/\; f_{\text{real}}, \\ \text{pass1}_{\text{rescaled}} &= \text{pass1} \;\cdot\; \frac{\max_{\text{arm} \in \{A,B,C\}} f_{\text{real,arm}}}{f_{\text{real}}}. \end{aligned}$$

The first is interpreted as an efficiency rate (units of *accuracy per realized token*, with the same denominator across arms within a checkpoint after subsequent rescaling). The second is a linear projection to the largest realized fraction observed in that checkpoint. These quantities are diagnostic rates, not achievable accuracies: pass@1 need not scale linearly with realized response tokens near saturation, and several projected entries exceed 1.0.

Prospective realized-budget reruns. The direct answer to "what happens if we train this comparison directly?" is a prospective budget audit rather than a post-hoc linear projection. We run matched Ashort/Blong reruns and report both realized response tokens and downstream MATH500 deltas. These reruns are not a dense return curve, but they are the relevant check for the paper's main claim: Ashort beats Blong on MATH500 at 1.5B, 7B, and L8B while consuming only 0.83–0.85× as many realized response tokens. The resulting framing is that the linear realized-token inversion is a *disclosure and diagnostic*; the prospective rerun is the training evidence.

C Full Result Tables

C.1 36-cell MATH500 scheduled-token grid (per-cell pass@1)

Table 2 shows the seed-42 baseline (single training seed for 7B/L8B); Table 3 keeps the 1.5B single-run row in the same layout and reports the 3-seed mean for 7B and L8B (all arms at seeds 42, 43, 44).

C.2 Realized-token rescale diagnostic (full 36 rows)

Figure 1 lists the arm tuples. The realized fractions used for this rescale are reported in Table 8.

Table 2: Full 36-cell scheduled-token MATH500 greedy pass@1 grid at training seed 42. Bold: the per-(checkpoint, d_{eval}) winner. Numbers use greedy mode.

	d_{eval}=1			d_{eval}=2			d_{eval}=4			d_{eval}=8		
checkpoint	A	B	C	A	B	C	A	B	C	A	B	C
1.5B	**.590**	.476	.530	**.650**	.592	.630	**.630**	.576	.568	**.580**	.492	.540
7B	**.614**	.564	.580	**.722**	.696	.706	.662	.652	**.688**	**.680**	.648	.674
L8B	**.588**	.492	.538	**.776**	.738	.734	**.726**	.668	.702	.698	.662	**.704**

Table 3: Scheduled-token MATH500 grid with the 1.5B single-run row and 7B/L8B 3-seed means (seeds 42, 43, 44) for all arms.

	d_{eval}=1			d_{eval}=2			d_{eval}=4			d_{eval}=8		
checkpoint	A	B	C	A	B	C	A	B	C	A	B	C
1.5B	.590	.476	.530	.650	.592	.630	.630	.576	.568	.580	.492	.540
7B (3s)	.608	.558	.577	.720	.669	.701	.683	.673	.672	.673	.645	.667
L8B (3s)	.583	.499	.552	.767	.735	.751	.709	.675	.701	.694	.657	.698

Table 4 reproduces every row of the realized-token rescale summary, sorted by checkpoint, arm, depth. The table is a sensitivity analysis, not an achievable accuracy table; the prospective rerun in the main text is the direct training test.

Per-cell winner counts: **scheduled-token (column** $\text{pass1}_{\textbf{maxtok}}$**)**: A = 11, B = 0, C = 1 out of 12 (checkpoint, d_{eval}) cells. **realized-token efficiency (column** $\text{pass1}_{\textbf{realtok}}$**)**: A = 2, B = 10, C = 0. **linear rescale (column** $\text{pass1}_{\textbf{rescaled}}$**)**: A = 2, B = 10, C = 0. The two realized-token columns agree on the inversion. Several Arm B realized-token efficiency entries exceed 1.0 because the metric is a rate, not an accuracy.

C.3 Prospective realized-budget audit artifacts

The prospective rerun summary used in Table 1 is computed from deterministic analysis summaries. The matched sets contain 3 seeds for 1.5B, 9 for 7B, and 8 for L8B. Across all benchmark/mode/depth cell means, Ashort wins 26 cells versus Blong's 8 at 1.5B (one tie), 23 versus 12 at 7B, and 27 versus 8 at L8B.

C.4 Inference-iso grand pool (per-pool, full 18 rows)

Table 5 reproduces the full cluster-corrected inference-iso grand-pool summary.

Cross-pool concordance. The 1.5B 4× depth-vs-sampling effect at the per-pool level is +5.19 pp on AIME, +4.30 pp on HMMT, and +4.74 pp on the grand pool: the three numbers agree in direction and within 0.5 pp of each other, suggesting the grand pool is a tight aggregate, not a cherry-picked merge.

C.5 TOST equivalence (full 18 rows, three margins each)

Table 6 reproduces the full TOST equivalence summary.

Cluster vs naive. After the W1 rerun, 7B 4× is not an equivalence case: depth is significantly better. The 7B 8× null is equivalent at δ=4 and δ=5 under both naive and cluster correction. The L8B 4× row shows why we report the cluster correction: naive paired would certify δ=4, but the cluster-corrected interval crosses that margin.

C.6 OOD pass@4 multi-seed (full 12 rows)

Table 7 reproduces the full multi-seed cluster-corrected summary on GPQA-Diamond and MMLU-Pro.

The naive paired CI on 7B MMLU-Pro B−A is $[-3.93, -0.07]$ which just excludes zero; the cluster CI $[-4.20, +0.27]$ does not. We treat this borderline cell as a tie under cluster correction, consistent with the rest of the table.

Table 4: Full realized-token rescale. $\mathrm{pass1}_{\mathrm{maxtok}}$: scheduled-token pass@1. $\mathrm{pass1}_{\mathrm{realtok}}$: pass@1 divided by mean realized fraction. $\mathrm{pass1}_{\mathrm{rescaled}}$: linear projection to the largest realized fraction in the checkpoint. Bold: the per-(checkpoint, d_{eval}) realized-token winner; star: the scheduled-token winner.

checkpoint	arm	d	f_{real}	max f	$\mathrm{pass1}_{\mathrm{maxtok}}$	$\mathrm{pass1}_{\mathrm{realtok}}$	$\mathrm{pass1}_{\mathrm{rescaled}}$
1.5B	A	1	0.842	0.859	⋆0.590	**0.701**	**0.602**
1.5B	A	2	0.842	0.859	⋆0.650	0.772	0.663
1.5B	A	4	0.842	0.859	⋆0.630	0.748	0.642
1.5B	A	8	0.842	0.859	⋆0.580	**0.689**	**0.591**
1.5B	B	1	0.720	0.859	0.476	0.661	0.567
1.5B	B	2	0.720	0.859	0.592	**0.822**	**0.706**
1.5B	B	4	0.720	0.859	0.576	**0.800**	**0.686**
1.5B	B	8	0.720	0.859	0.492	0.683	0.586
1.5B	C	1	0.859	0.859	0.530	0.617	0.530
1.5B	C	2	0.859	0.859	0.630	0.734	0.630
1.5B	C	4	0.859	0.859	0.568	0.662	0.568
1.5B	C	8	0.859	0.859	0.540	0.629	0.540
7B	A	1	0.847	0.856	⋆0.608	0.718	0.614
7B	A	2	0.847	0.856	⋆0.720	0.850	0.727
7B	A	4	0.847	0.856	⋆0.683	0.806	0.690
7B	A	8	0.847	0.856	⋆0.673	0.794	0.680
7B	B	1	0.691	0.856	0.558	**0.808**	**0.691**
7B	B	2	0.691	0.856	0.669	**0.969**	**0.829**
7B	B	4	0.691	0.856	0.673	**0.975**	**0.834**
7B	B	8	0.691	0.856	0.645	**0.934**	**0.799**
7B	C	1	0.856	0.856	0.577	0.674	0.577
7B	C	2	0.856	0.856	0.701	0.819	0.701
7B	C	4	0.856	0.856	0.672	0.785	0.672
7B	C	8	0.856	0.856	0.667	0.779	0.667
L8B	A	1	0.782	0.801	⋆0.583	0.746	0.597
L8B	A	2	0.782	0.801	⋆0.767	0.982	0.786
L8B	A	4	0.782	0.801	⋆0.709	0.907	0.726
L8B	A	8	0.782	0.801	0.694	0.888	0.711
L8B	B	1	0.640	0.801	0.499	**0.780**	**0.624**
L8B	B	2	0.640	0.801	0.735	**1.148**	**0.919**
L8B	B	4	0.640	0.801	0.675	**1.055**	**0.844**
L8B	B	8	0.640	0.801	0.657	**1.026**	**0.821**
L8B	C	1	0.801	0.801	0.552	0.690	0.552
L8B	C	2	0.801	0.801	0.751	0.938	0.751
L8B	C	4	0.801	0.801	0.701	0.876	0.701
L8B	C	8	0.801	0.801	⋆0.698	0.872	0.698

D Generation-Budget Sanity Check

A natural concern about the main fixed-budget grid is that $\mathrm{max_new}{=}2{,}048$ is below the generation budget used in some reference reports. We ran a base-model MATH500 sweep over $\mathrm{max_new} \in \{2048, 4096, 8192\}$, summarized in Figure 5. Moving from $2{,}048$ to $8{,}192$ closes 22/30/27 pp at 1.5B/7B/L8B. The sanity check does not change the cross-arm conclusion because every main-grid cell uses the same decoding budget, but it cautions against comparing absolute scores across papers without matching the decoding protocol.

E Realized-Fraction Instrumentation

Each training run records a structured summary with the fields needed for budget accounting: depth, optimization steps, prompts per step, rollouts per prompt, maximum new tokens, LoRA settings, seed, scheduled tokens, realized response tokens, realized fraction, and wall time. In a representative Arm C run, $M{=}100$, $B_p{=}2$, $n{=}8$, $d{=}1$, and $\mathrm{max_new}{=}2{,}048$ give $3{,}276{,}800$ scheduled tokens; the run emits $2{,}811{,}238$ response tokens, for a realized fraction of 0.8579. The realized fraction is the ratio between emitted response tokens and the scheduled budget $M \cdot B_p \cdot n \cdot d \cdot \mathrm{max_new}$. Per-arm averaged realized fractions are shown in Table 8.

Table 5: Full inference-iso grand pool with cluster-by-problem 95% CIs. Each cell reports $\Delta = \text{pass1}_{\text{depth}} - \text{pass1}_{\text{sample}}$ on the $4\times$ or $8\times$ compute multiplier; cluster CIs from 10,000 bootstrap resamples.

checkpoint	pool	cmp	n_{clusters}	n_{pairs}	Δ (pp)	cluster 95% CI	cluster p
1.5B	AIME	$4\times$	90	270	$+5.19$	$[+0.37, +10.37]$	0.038
1.5B	AIME	$8\times$	90	270	-5.19	$[-10.00, -0.74]$	0.021
1.5B	HMMT	$4\times$	93	279	$+4.30$	$[+1.08, +7.89]$	0.003
1.5B	HMMT	$8\times$	33	99	-5.05	$[-12.12, +1.01]$	0.064
1.5B	grand	$4\times$	183	549	$+4.74$	$[+1.82, +7.83]$	0.001
1.5B	grand	$8\times$	123	369	-5.15	$[-8.94, -1.36]$	0.005
7B	AIME	$4\times$	90	270	$+6.67$	$[+1.48, +11.85]$	0.010
7B	AIME	$8\times$	90	270	-1.11	$[-7.04, +4.44]$	0.686
7B	HMMT	$4\times$	93	279	$+1.79$	$[-1.79, +5.38]$	0.315
7B	HMMT	$8\times$	93	279	$+2.87$	$[-0.36, +6.81]$	0.099
7B	grand	$4\times$	183	549	$+4.19$	$[+1.09, +7.47]$	0.011
7B	grand	$8\times$	183	549	$+0.91$	$[-2.00, +4.01]$	0.553
L8B	AIME	$4\times$	90	270	-1.85	$[-7.41, +3.33]$	0.486
L8B	AIME	$8\times$	90	270	-4.07	$[-9.26, +1.11]$	0.114
L8B	HMMT	$4\times$	93	279	-1.43	$[-4.30, +1.43]$	0.312
L8B	HMMT	$8\times$	93	279	-5.02	$[-9.32, -1.08]$	0.007
L8B	grand	$4\times$	183	549	-1.64	$[-4.74, +1.46]$	0.268
L8B	grand	$8\times$	183	549	-4.55	$[-8.01, -1.28]$	0.005

Table 6: Full TOST equivalence verdicts at α=0.05, three margins, both naive paired (n) and cluster-corrected (c). "y" = the corresponding 90% CI lies entirely inside $(-\delta, +\delta)$.

checkpoint	cmp	Δ (pp)	naive 90% CI	cluster 90% CI	δ=3		δ=4		δ=5	
					n	c	n	c	n	c
1.5B	$4\times$	$+4.74$	$[+2.73, +6.74]$	$[+2.37, +7.29]$	n	n	n	n	n	n
1.5B	$8\times$	-5.15	$[-8.13, -2.17]$	$[-8.40, -2.17]$	n	n	n	n	n	n
7B	$4\times$	$+4.19$	$[+1.82, +6.74]$	$[+1.64, +6.92]$	n	n	n	n	n	n
7B	$8\times$	$+0.91$	$[-1.46, +3.28]$	$[-1.64, +3.46]$	n	n	y	y	y	y
L8B	$4\times$	-1.64	$[-3.83, +0.73]$	$[-4.19, +0.91]$	n	n	y	n	y	y
L8B	$8\times$	-4.55	$[-6.92, -2.19]$	$[-7.47, -1.82]$	n	n	n	n	n	n

Why does Arm B realize fewer tokens? Refinement at d=4 feeds the previous draft into the next chunk. Because the model has already seen its earlier answer, later chunks reach <eos> faster: the second pass tends to be a short revise rather than a fresh draft. The first chunk fills max_new=2,048 about as often as it does for Arm A; the later chunks emit short "no-revision-needed" or short patch outputs and stop early. The effect is systematic and grows with model scale, plausibly because the larger model is more confident in early chunks and patches less.

F Off-Budget Runs and Remediation

Issue. Early in our training cycle, the 7B Arm B and L8B Arm B runs were submitted at max_new=1,536 rather than the intended 2,048. The audit found that these runs used 75% of the intended scheduled-token budget (2.46M instead of 3.28M), so they did not satisfy the strict fixed-budget match. We replaced them with on-budget runs before finalizing the reported results.

On-budget replacement. We retrained 7B Arm B and L8B Arm B at max_new=2,048 and replaced the off-budget rows before the final analysis. All Section 5 scheduled-token grid numbers use these on-budget LoRAs. The Section 5 inference-iso grand pool also uses the on-budget 7B/L8B replacements; those reruns completed by 2026−04−29 and are included in the final grand-pool analysis.

Effect on the final claims. The off-budget inference rows are excluded from the final tables. The on-budget rerun changes the inference interpretation: 7B at $4\times$ is now a significant depth-favoring

Table 7: Full multi-seed cluster-corrected 95% CIs of (Arm X − Arm A) on per-problem pass@4, three sampling seeds, $n_{\text{clusters}} \in \{198, 500\}$, $n_{\text{pairs}} \in \{594, 1500\}$. "verdict" = whether cluster CI excludes 0 ("→A" if the CI is wholly negative, "tie" otherwise).

checkpoint	bench	cmp	n_{cluster}	Δ (pp)	cluster 95% CI	cluster p	verdict
1.5B	GPQA	B−A	198	−8.59	[−12.63, −4.71]	0.0002	→A
1.5B	GPQA	C−A	198	−5.05	[−9.09, −1.01]	0.014	→A
1.5B	MMLU-Pro	B−A	500	−4.73	[−7.40, −2.07]	0.0008	→A
1.5B	MMLU-Pro	C−A	500	−1.00	[−3.60, +1.60]	0.448	tie
7B	GPQA	B−A	198	−1.85	[−6.06, +2.36]	0.386	tie
7B	GPQA	C−A	198	+1.68	[−2.19, +5.72]	0.409	tie
7B	MMLU-Pro	B−A	500	−2.00	[−4.20, +0.27]	0.084	tie
7B	MMLU-Pro	C−A	500	−0.87	[−3.07, +1.33]	0.448	tie
L8B	GPQA	B−A	198	−3.37	[−7.41, +0.67]	0.099	tie
L8B	GPQA	C−A	198	+1.68	[−2.36, +5.72]	0.418	tie
L8B	MMLU-Pro	B−A	500	−0.20	[−2.33, +1.87]	0.851	tie
L8B	MMLU-Pro	C−A	500	−0.20	[−2.40, +2.07]	0.878	tie

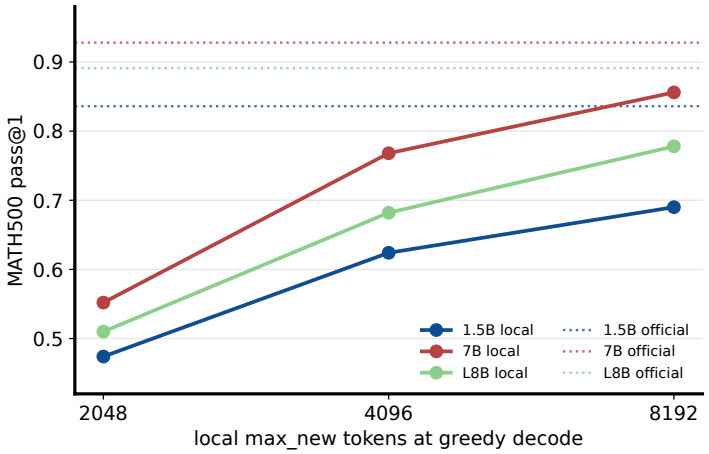


Figure 5: **Generation budget and the local-vs-reference MATH500 gap.** Each solid curve is the base-model local greedy accuracy at max_new $\in \{2048, 4096, 8192\}$; each dotted line is the corresponding DeepSeek-Distill reference number reported under 32K tokens × 64 samples. Going from 2,048 to 8,192 closes about 30 pp (7B), 22 pp (1.5B), and 27 pp (L8B). **Take-away:** most of the local-vs-reference gap closes as the generation budget increases, so absolute scores should not be compared without the decoding protocol.

cell (+4.19 pp, cluster CI [+1.09, +7.47], p=0.011), 7B at 8× remains null/equivalent, and L8B at 8× remains sampling-favoring but with a smaller magnitude (−4.55 pp).

G Sampled-MATH500 Sub-experiment (Arm B 7B d=2 Single-Cell Flip)

We also evaluated the same trained-LoRA grid under sampling T=0.6, top_p=0.95 on MATH500 (single-pass, no d_{eval} sweep). The motivation was the natural question: "does Arm B do better under sampling?". The 9-cell sampled (checkpoint ×d_{eval}) grid in Table 9 is overall dominated by Arm A; one stochastic cell flips:

We do *not* elevate this single-cell flip into a "d-axis is good for sampling" claim; the rest of the 9-cell sampled grid still shows Arm A winning the majority. The 7B Arm B d=2 cell is reported here as a single observation for transparency.

H Per-Question Flip Events (Extended)

Source. The deduplicated flip-event table contains 1,332 data rows across 446 unique source records. Each row records one $(d \to d{+}1)$ transition for one (run, model, bench, seed, max d). We

Table 8: Per-arm mean realized fraction, averaged across the available training seeds in each cell.

checkpoint	arm A	arm B	arm C	arm B vs. arm A
1.5B	0.842 (n_{seeds}=3)	0.720 (3)	0.859 (3)	−14.4%
7B	0.847 (2)	0.691 (2)	0.856 (2)	−18.5%
L8B	0.782 (2)	0.640 (2)	0.801 (2)	−18.1%

Table 9: Sampled MATH500 (T=0.6) by checkpoint $\times d_{\text{eval}}$. The 7B Arm B d=2 cell is the only cell where Arm B exceeds Arm A under sampling; the gap (+1.4 pp) is within the sampling-seed variance probe of $\sim$ 2.6 pp standard deviation across seeds $\{42, 43, 44\}$.

checkpoint	d=1 A/B/C	d=2 A/B/C	d=4 A/B/C
1.5B	0.580/0.494/0.518	0.700/0.672/0.642	0.610/0.554/0.534
7B	0.620/0.566/0.584	0.742/**0.756**/0.708	0.668/0.654/0.682
L8B	0.568/0.488/0.546	0.748/0.730/0.736	0.716/0.660/0.700

canonicalize on the highest available $\max d$ JSON Lines file per run for the quoted 14B-distill MATH500 trajectory in this appendix section.

Convention. "positive flip" = the problem was incorrect at d=d and correct at d=d+1; "negative flip" = the inverse. "churn" = positive + negative. Aggregate Δpass@1 at the transition is $(\text{pos} - \text{neg})/n_{\text{problems}}$.

14B-distill MATH500 trajectory ($\max d$=8, seed 42). Reproduced from Figure 6 for completeness, and extended with churn in Table 10:

Table 10: Per-transition flip counts and aggregate change. One canonical row is used per transition.

transition	pos	neg	churn	Δ aggregate (pp)
d=1 → 2	104	72	176	+6.4
d=2 → 3	34	57	91	−4.6
d=3 → 4	55	34	89	+4.2
d=4 → 5	35	36	71	−0.2
d=5 → 6	42	38	80	+0.8
d=6 → 7	41	32	73	+1.8
d=7 → 8	38	31	69	+1.4

Aggregate across d=1 → 8: +10 pp net (the 14B-distill MATH500 trajectory ends at higher accuracy than it starts), but with 649 total problem-flips along the way. The d=2 → 3 transition is the only single transition where the aggregate *regresses*; the d=4 → 5 transition is the only transition where the aggregate is near-zero despite 71 problems flipping.

I Negative and Deferred Checks

We record negative and deferred checks to reduce selection bias.

Arm C at M=50, n=16 (out of memory). Our original Arm C plan was M=50, n=16 to match the $n{\cdot}d$=16 count of Arm B. This configuration was not used in the final grid because 32 simultaneous rollouts exceeded the KV-cache budget at max_new=2,048 on a single H100. We rebudgeted to M=100, n=8 (the Arm C in the paper), which preserves total tokens and reduces simultaneous rollouts to 16.

Excluded off-budget runs at max_new=1,536. The 7B/L8B Arm B initial runs were submitted at max_new=1,536, 25% below the fixed-token schedule (Appendix F). We exclude these runs from the final inference-iso grand pool after the on-budget reruns were completed. They remain documented here because they explain why earlier reports had a weaker 7B inference signal and a larger L8B sampling win.

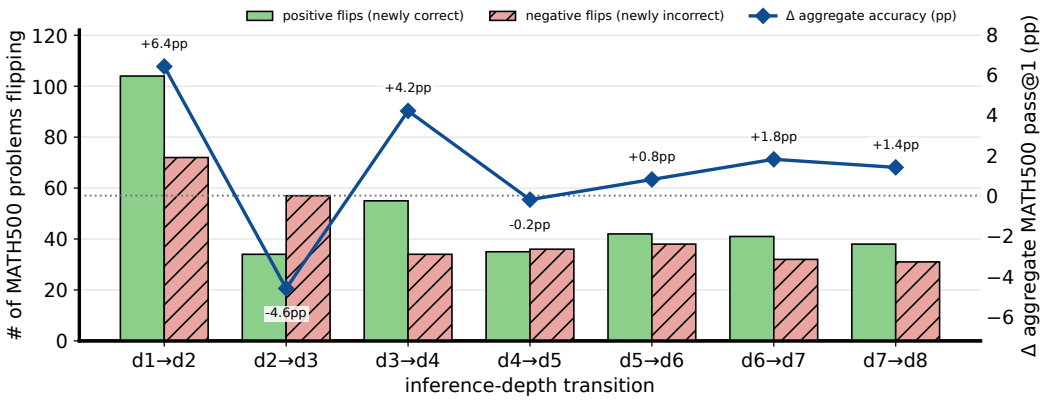


Figure 6: **Per-question correctness churns at every depth transition, even when aggregate accuracy looks small.** 14B-distill MATH500 (seed 42, $\max d{=}8$). Bars are positive vs. negative flips at each $d{\to}d{+}1$ transition; the diamond curve shows the corresponding aggregate change. The $d{=}2{\to}d{=}3$ step regresses by 4.6 pp on aggregate *and* on per-question; the $d{=}4{\to}d{=}5$ step is near-zero in aggregate but 71 problems flip. **Take-away:** aggregate accuracy is a lossy summary of what depth refinement is doing per problem.

Sampled MATH500 cell variance probes. The single-stochastic-flip cell at 7B Arm B $d{=}2$ (Appendix G) had a sampling-seed variance probe across seeds $\{42, 43, 44\}$ giving $0.756/0.714/0.776$ respectively, mean 0.745, standard deviation ~ 2.6 pp. The point estimate of 0.756 at seed 42 is within seed-noise of the mean. We do *not* treat this as evidence that Arm B trains better for sampling; the 9-cell sampled grid still has Arm A winning a majority of cells.

LoRA-rank ablations excluded from realized-token rescale. Two earlier LoRA-rank ablations, one rank-8 Arm A run (realized_frac=0.860) and one rank-64 Arm B run (realized_frac=0.759), were inadvertently swept into an earlier realized-fraction table by a broad inclusion rule. The final analysis uses a strict seed-run inclusion list and a scheduled-budget check at $3{,}276{,}800$ tokens. Per-arm means tightened slightly: 1.5B Arm A from 0.847 to 0.842; Arm B from 0.736 to 0.720. The body reports this cleaned table.

Deferred coverage. LiveCodeBench and IFEval remain outside this audit because LiveCodeBench requires sandboxed code execution, which we did not use under the offline-compute protocol. The 14B Arm C training run is also deferred (estimated ~ 25 GPU-h on H200) because the $1.5\text{B}/7\text{B}/\text{L}8\text{B}$ cross-scale story does not need a fourth scale to motivate the accounting-sensitivity claim.

J Reproducibility

Code archive contents.

- Training entry point for LoRA-GRPO.
- Evaluation entry point for greedy and sampled decoding; the evaluator can optionally dump per-problem correctness records.
- Benchmark loaders for MATH, AIME, HMMT, GPQA-Diamond, MMLU-Pro, and chat-template construction.
- Verifier utilities for math equivalence and MCQ letter matching.
- Released result summaries for the realized-token rescale, prospective realized-budget reruns, clustered bootstrap, and TOST equivalence checks.
- Small example scripts that exercise the training and evaluation paths.

Reproduction recipe.

1. Install the packages in the released `requirements.txt`.

2. Run the example scripts on a machine with at least one H100 (94 GB) or H200 (141 GB), or a comparable GPU for smaller smoke tests.
3. For paper-scale cells, rerun the trainer and evaluator with the hyperparameters in Appendix A. The review archive does not include trained LoRA adapters; they will be released after acceptance.
4. Compare the released CSV summaries for clustered bootstrap, realized-token accounting, prospective reruns, and TOST with the tables in this appendix.

Seeds. The paper-scale analyses used training seeds $\{42, 43, 44, 45\}$, sampling seeds $\{42, 43, 44\}$, and bootstrap seed 42 for the reported summaries.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the four contributions (strict 36-cell fixed-token-budget audit; realized-token diagnostic plus prospective rerun; cluster-corrected inference audit with TOST equivalence; multi-seed cluster-corrected OOD); each is bounded by the scope of the experiments and final on-budget inference reruns.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 7 lists the concrete limitations: realized-budget coverage is targeted rather than a dense return curve; 1.5B has smaller $8\times$ inference coverage; checkpoint and task scope are bounded; and the paper does not claim a mechanism for early termination or depth crossover.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: This is an empirical fixed-budget audit; we make no theoretical claims and prove no theorems.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 3 and Appendix A disclose every hyperparameter, the LoRA rank, the GRPO hyperparameters, the per-arm $(M, B_p, n, d, \text{max_new})$ tuple, the inference $T/\text{top_p}/\text{max_new}$ settings, the bootstrap protocol, the cluster unit, and the random seeds. The repository at `https://anonymous.4open.science/r/fixed-budget-rlvr-B29D` contains the training and evaluation code plus released result summaries.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The training and evaluation code, together with compact result summaries for the realized-token, prospective rerun, clustered-bootstrap, and TOST analyses, is in the repository at `https://anonymous.4open.science/r/fixed-budget-rlvr-B29D`. Training datasets are public (`lighteval` MATH split for training; AIME/HMMT/MATH500/GPQA-Diamond/MMLU-Pro for evaluation). Trained LoRA adapters will be released after acceptance.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Sections 3–5 and Appendix A cover the full setup. Numbers in the main text exactly match the corresponding cells in the released CSVs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: Every inference-time and OOD claim is reported with a cluster-by-problem 95% CI from a 10,000-resample bootstrap; null cells are bounded by TOST equivalence margins (± 3, ± 4, ± 5 pp). Training-time claims are supported by per-cell scheduled-token winner counts, the realized-token diagnostic table, high-scale 3-seed validation, and matched-seed prospective realized-budget reruns.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Appendix A reports the hardware, approximate per-run GPU-hour ranges, evaluation wall-time ranges, and explains that the total compute is order 10^2 GPU-h rather than an exact sum because several runs were re-submitted during remediation.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes]

Justification: We use only public benchmarks and public model weights; no human-subject or biometric data is involved.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This is a methodological audit of training-compute allocation for math reasoning; positive and negative societal impacts are inherited from the underlying base models and benchmarks rather than created or amplified by our audit.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: We release LoRA adapters trained on a public benchmark for a math reasoning task; no high-risk data or models are released.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The DeepSeek-R1-Distill checkpoints (MIT-licensed), MATH, AIME, HMMT, GPQA-Diamond (under their original licenses), MMLU-Pro (MIT), vLLM (Apache 2.0), and the fixed-budget and flip-event anchor papers are credited in Section 2 and the references.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The code repository includes README instructions, training and evaluation scripts, and released result summaries: `https://anonymous.4open.science/r/fixed-budget-rlvr-B29D`. Trained LoRA adapters will be released after acceptance.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: No human-subject experiments.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human-subject experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLM checkpoints are central objects of study in this work: the paper specifies the DeepSeek-R1-Distill models, the LoRA-GRPO training setup, and the evaluation protocol in Sections 3–5 and Appendix A. The work does not use LLMs or agents as a non-standard analysis or evaluation component beyond the studied models themselves.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.