SERA: SCALE-EQUALIZED ROLLOUT ALLOCATION FOR MAXIMUM LIKELIHOOD REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Maximum Likelihood Reinforcement Learning (MaxRL) targets prompt-wise log-success and has shown strong performance on reasoning tasks. Under finite rollout budgets, however, the estimator used by MaxRL attenuates each prompt's likelihood gradient by a factor that depends on its success probability and rollout count. Under uniform rollout allocation, the common rollout count fails to compensate for success-dependent attenuation, leaving low-success prompts more strongly attenuated and distorting their relative contributions to the expected aggregate gradient. We introduce SERA (Scale-Equalized Rollout Allocation), which redistributes a fixed rollout budget to approximately equalize these finite-rollout scaling factors. Building on our theoretical analysis of how finite rollouts distort prompt-wise likelihood gradients, we formulate the allocation as a fixed-budget max–min problem, derive a waterline solution to its continuous relaxation, and introduce a multiplicity correction to remove the additional prompt weighting induced by heterogeneous rollout counts. Experiments show stronger alignment with exact likelihood gradients in a controlled ImageNet setting and improved multi-sample solution coverage over MaxRL on maze navigation and mathematical reasoning under matched training rollout budgets.

1 INTRODUCTION

Reinforcement learning with verifiable rewards has become a standard approach to improving language-model reasoning, as exemplified by GRPO (Shao et al., 2024) and DeepSeek-R1 (Guo et al., 2025). Yet single-sample accuracy does not fully characterize reasoning performance: at inference time, models are often sampled repeatedly, and their ability to produce at least one correct solution is commonly measured by Pass@K (Chen et al., 2021). Improvements in Pass@1 need not translate into better coverage at larger sampling budgets (Yue et al., 2025), motivating objectives that account for multi-sample success. Maximum Likelihood Reinforcement Learning (MaxRL) (Tajwar et al., 2026) targets average prompt-wise log-success, placing greater relative weight on improvements for low-success prompts. Its likelihood gradient can be expressed as a combination of Pass@K gradients across sampling budgets, directly connecting the objective to multi-sample solution coverage.

Realizing this objective with finitely many rollouts, however, introduces a systematic mismatch between the target likelihood gradient and the expected MaxRL update. The practical centered MaxRL estimator attenuates each prompt's log-success gradient by a coefficient that depends jointly on its success probability and rollout count. Under uniform allocation, the same rollout count leaves low-success prompts more strongly attenuated, reducing their relative contributions despite the greater emphasis placed on them by the target log-success objective. When prompt gradients differ in direction, such unequal scaling can also shift the expected batch gradient away from the target likelihood gradient. This raises a central question: *Given a fixed training rollout budget, how should rollouts be distributed across prompts to better preserve their relative log-success gradient contributions?*

Existing rollout allocation methods adapt sampling based on prompt difficulty, learning signals, or objective-induced weighting (Yang et al., 2025b; Xiong et al., 2025; Nguyen et al., 2026; Li et al., 2025). In contrast, we study the prompt-dependent finite-rollout scaling induced by the centered

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.

MaxRL estimator itself. Specifically, we ask how a fixed rollout budget should be allocated to control the coefficients that scale prompt-wise likelihood gradients in the expected MaxRL update.

We introduce SERA (Scale-Equalized Rollout Allocation), which redistributes a fixed rollout budget to approximately equalize the finite-rollout scaling coefficients across prompts. For a fixed retained prompt set, our analysis shows that matching these coefficients preserves the relative likelihood-gradient contributions up to a common factor. We therefore maximize the minimum scaling coefficient across prompts under the budget constraint; in the continuous relaxation, the optimum equalizes these coefficients and admits a waterline solution. The resulting allocation recovers inverse-success behavior in the low-success regime while retaining the full finite-rollout dependence of the centered MaxRL estimator. Unequal rollout counts, however, also induce additional prompt weighting under response-mean aggregation. We therefore pair the allocation with a multiplicity correction that removes this effect, allowing rollout counts to control finite-rollout attenuation without introducing a separate group-size weighting.

We evaluate SERA against MaxRL under matched training rollout budgets on ImageNet (Deng et al., 2009), maze navigation (Tajwar et al., 2026), and mathematical reasoning with SmolLM2-360M-Instruct (Allal et al., 2025), Qwen2.5-Math-1.5B (Yang et al., 2024b), and Qwen3-4B-Base (Yang et al., 2025a). In a controlled ImageNet setting, SERA produces sampled updates that align more closely with the exact likelihood gradient, supporting the predicted effect of scale equalization. Using historical success estimates, SERA also improves held-out maze coverage under both tested loss normalizations and consistently increases multi-sample solution coverage on mathematical reasoning tasks. On SmolLM2-360M-Instruct, Pass@256 on GSM8K-Platinum (Cobbe et al., 2021; Vendrow et al., 2025) improves by 5.13 percentage points under both normalization schemes. On Qwen2.5-Math-1.5B, Pass@512 improves by 7.00 points on BeyondAIME (ByteDance-Seed, 2025) and 6.67 points on AIME 2025 (Zhang & Team, 2025); on Qwen3-4B, it improves by 8.00 points on BeyondAIME and 4.78 points on Minerva Math (Lewkowycz et al., 2022).

Our contributions are:

1. **Finite-rollout analysis of the centered MaxRL estimator.** We characterize the conditional objective and expected batch gradient of centered MaxRL under heterogeneous rollout counts, showing how unequal prompt-wise scaling can alter the aggregate gradient direction and how scale equalization preserves the relative likelihood-gradient contributions of a fixed retained set.
2. **Scale-equalized rollout allocation.** Building on this analysis, we develop SERA, formulating rollout allocation as a fixed-budget max–min problem over the finite-rollout scaling coefficients and deriving a waterline solution to its continuous relaxation. We further introduce a multiplicity correction that removes the additional prompt weighting induced by unequal rollout counts under response-mean aggregation.
3. **Experimental validation across tasks.** We validate the gradient-alignment effect predicted by our analysis in a controlled ImageNet setting and demonstrate improved multi-sample solution coverage over MaxRL on maze navigation and mathematical reasoning with language models under matched training rollout budgets.

2 PRELIMINARIES

We consider generation tasks with binary verifier rewards $R_q(y) \in \{0, 1\}$. For prompt q and response $y \sim \pi_\theta(\cdot \mid q)$, define $p_q(\theta) := \Pr[R_q(y) = 1]$. Following MaxRL (Tajwar et al., 2026), we consider the prompt-wise log-success objective

$$J_{\log}(\theta) := \mathbb{E}_{q\sim\mathcal{D}}[\log p_q(\theta)]. \tag{1}$$

Given N i.i.d. responses $y_{qi} \sim \pi_\theta(\cdot \mid q)$, let $R_{qi} = R_q(y_{qi})$, $\overline{R}_q = \frac{1}{N}\sum_{i=1}^{N} R_{qi}$, and $s_{qi} = \nabla_\theta \log \pi_\theta(y_{qi} \mid q)$. To reduce gradient variance and improve training stability, practical MaxRL uses the following centered estimator:

$$\widehat{g}_q^{(N)} := \begin{cases} \dfrac{1}{N}\displaystyle\sum_{i=1}^{N} \frac{R_{qi} - \overline{R}_q}{\overline{R}_q}\, s_{qi}, & \overline{R}_q > 0, \\ 0, & \overline{R}_q = 0. \end{cases} \tag{2}$$

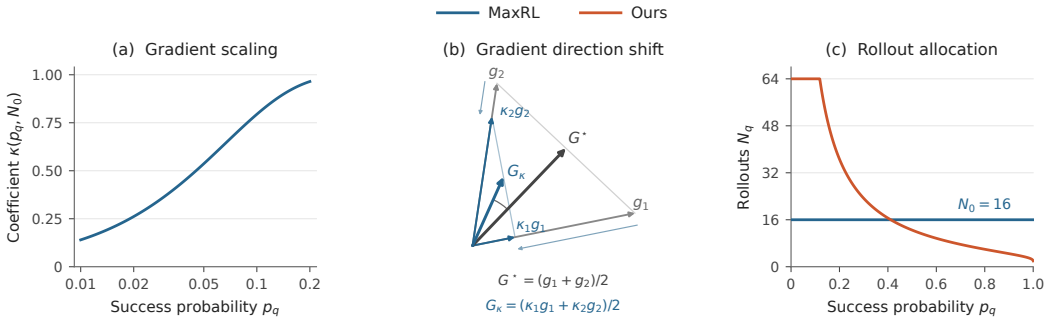

Figure 1: **Gradient scaling and rollout allocation.** (a) Fixed-rollout MaxRL induces success-dependent scaling. (b) Unequal prompt-wise scaling can alter the expected batch-gradient direction. (c) Ours assigns more rollouts to lower-success prompts under the same total budget ($N_{\max} = 64$).

For $p_q \in (0, 1)$, its expectation is

$$\mathbb{E}[\hat{g}_q^{(N)}] = \underbrace{\left[1 - (1 - p_q)^{N-1}\right]}_{\kappa(p_q, N)} \nabla_\theta \log p_q. \tag{3}$$

We refer to $\kappa(p, N)$ as the *finite-rollout fidelity*. At a fixed rollout count, this coefficient increases with the success probability, so lower-success prompts experience stronger attenuation. Uniform rollout allocation can therefore produce unequal scaling across prompts. Since prompt gradients need not be collinear, this can change the direction of their expected aggregate. Further theoretical details are provided in Appendix B.

3 Method

SERA redistributes a fixed rollout budget to approximately equalize the prompt-wise finite-rollout scaling coefficients of the centered MaxRL estimator. The resulting allocation is paired with multiplicity correction to remove the additional response-count weighting introduced by unequal rollout counts under response-mean aggregation.

3.1 Rollout Allocation

Equation 3 shows that the expected MaxRL estimator is scaled by $\kappa(p_q, N) = 1 - (1 - p_q)^{N-1}$. As illustrated in Figure 1 (a), under a fixed rollout count N_0, this coefficient varies with the success probability p_q. These unequal coefficients change the relative contributions of prompt gradients and can therefore alter the direction of their expected aggregate. Figure 1 (b) illustrates this effect: although scaling preserves each prompt gradient's direction, their expected mean deviates from the uniformly averaged log-success gradient.

If the success probabilities were known, directly dividing each estimator by $\kappa(p_q, N_0)$ would remove this scaling in expectation. However, this correction leaves the probability of zero-signal groups unchanged and multiplies the estimator's covariance by $1/\kappa(p_q, N_0)^2$, amplifying noise when κ is small. We instead adjust rollout counts to equalize $\kappa(p_q, N_q)$ across prompts. A shared positive coefficient preserves the relative log-success gradient contributions up to a common factor, while changing the rollout counts also changes the probability of obtaining mixed-reward groups.

Let $\mathcal{B}$ contain B prompts, with a fixed total rollout budget $M := BN_0$ and $N_0 \geq 2$. Under this fixed budget, we maximize the minimum fidelity coefficient across prompts, thereby improving the most strongly attenuated prompt. Initially treating counts as continuous and setting aside practical count bounds, we consider

$$\max_{\{N_q\}_{q \in \mathcal{B}}} \quad \min_{q \in \mathcal{B}} \kappa(p_q, N_q) \qquad \text{s.t.} \quad \sum_{q \in \mathcal{B}} N_q = M, \qquad N_q \geq 1. \tag{4}$$

Because $\kappa(p_q, N_q)$ increases with N_q, any continuous allocation with unequal fidelity levels can increase the minimum by shifting rollout budget from a higher-fidelity prompt to a lower-fidelity one. The continuous optimum therefore equalizes the coefficients across prompts. Writing their common value as $\kappa^\star_{\mathcal{B}}$ and defining $c_{\mathcal{B}} := -\log(1-\kappa^\star_{\mathcal{B}})$, we obtain $(N^\star_q - 1)[-\log(1-p_q)] = c_{\mathcal{B}}$. The resulting allocation is

$$N^\star_q = 1 + \frac{c_{\mathcal{B}}}{-\log(1-p_q)}. \tag{5}$$

The budget constraint then gives

$$c_{\mathcal{B}} = \frac{M-B}{\sum_{q\in\mathcal{B}} [-\log(1-p_q)]^{-1}}, \qquad \kappa^\star_{\mathcal{B}} = 1 - e^{-c_{\mathcal{B}}}. \tag{6}$$

Thus, lower-success prompts receive more rollouts to compensate for stronger attenuation. The shared water level $c_{\mathcal{B}}$ depends on the entire batch, and equal success probabilities recover uniform allocation. In practice, rollout counts must be integers satisfying $N_{\min} \le N_q \le N_{\max}$. We first incorporate these bounds into the continuous allocation:

$$N^{\text{cont}}_q = \text{clip}\left(1 + \frac{c_{\mathcal{B}}}{-\log(1-p_q)}, N_{\min}, N_{\max}\right), \qquad \sum_{q\in\mathcal{B}} N^{\text{cont}}_q = M. \tag{7}$$

Here, $c_{\mathcal{B}}$ is a shared constant recomputed so that the bounded counts sum to M. This waterline solves the bounded continuous max–min problem (Appendix B.3). Figure 1 (c) illustrates this allocation: under the same total budget as fixed-rollout MaxRL, lower-success prompts receive more rollouts, up to $N_{\max}$. We then round the counts to integers while preserving the budget and bounds, yielding approximate scale equalization. The integerization procedure is detailed in Appendix C.2.

3.2 Observability Safeguard

Equalizing gradient scales does not ensure that a sampled group provides a nonzero learning signal. For the centered estimator, both all-incorrect and all-correct groups have zero advantages. Nonzero advantages therefore require a group to contain both a success and a failure, which occurs with probability

$$U_q(N) := 1 - (1-p_q)^N - p_q^N. \tag{8}$$

Using estimates $\widehat{p}_q$, we rank prompts by $\widehat{U}_q(N_{\max})$ and search over capacity-feasible prefixes of this ranking. For each candidate set, we recompute the waterline under the original budget $M = BN_0$, obtain integer counts using Section 3.1, and check whether every retained prompt satisfies $\widehat{U}_q(N_q) \ge u_0$, where $u_0 \in [0, 1)$. If no tested prefix qualifies, a capacity-preserving fallback allocates the full budget to the top-ranked prompts (Appendix C.3). The selected set $\mathcal{A} \subseteq \mathcal{B}$ receives the full original budget, while inactive prompts receive zero rollouts.

3.3 Implementation

Success-probability estimation. Inspired by other allocation methods (Li et al., 2025; Xiong et al., 2025), we estimate p_q using reward observations from previous epochs. At the start of epoch e, we compute

$$\widehat{p}^{(e)}_q = \frac{\alpha_0 + S^{(e)}_q}{\alpha_0 + \beta_0 + S^{(e)}_q + F^{(e)}_q}, \tag{9}$$

where $S^{(e)}_q$ and $F^{(e)}_q$ are discounted success and failure counts from earlier epochs, and $\alpha_0, \beta_0 > 0$ define a fixed Beta prior. Estimates remain fixed within each epoch, and new outcomes are incorporated at the next epoch boundary.

Multiplicity correction. Under response-mean aggregation, directly averaging the losses of all generated responses weights each prompt in proportion to its rollout count N_q. Even when the coefficients $\kappa(p_q, N_q)$ are equalized, this additional weighting can change the relative contributions

Algorithm 1 Training procedure for SERA

Require: Policy π_θ; baseline count N_0; bounds $N_{\min}, N_{\max}$; threshold u_0
Require: Prior $\alpha_0, \beta_0 > 0$; discount $\lambda \in [0, 1]$
Initialize historical statistics and pending outcomes
for each epoch **do**
  Update discounted statistics and freeze available success estimates
  Clear pending outcomes
  for candidate batch $\mathcal{B}$ **do**
    $M \leftarrow |\mathcal{B}|N_0$
    if all prompts have available estimates **then**
      Jointly select $\mathcal{A}$ and integer counts using observability-guided selection and bounded waterline Equation 7
    else
      $\mathcal{A} \leftarrow \mathcal{B}$; $N_q \leftarrow N_0$ for all $q \in \mathcal{B}$
    end if
    Generate N_q responses for each $q \in \mathcal{A}$ and evaluate rewards
    Compute centered MaxRL advantages as in Equation 2
    Apply response weights from Equation 10 to the advantages and update θ
    Accumulate observed rewards in pending outcomes
  end for
end for

of prompt gradients. To remove this response-count weighting, we multiply each response's centered advantage by

$$w_q := \frac{N_0}{N_q}, \qquad q \in \mathcal{A}. \tag{10}$$

Since $N_q w_q = N_0$, every retained prompt has the same total response weight, regardless of its rollout count. Under response-mean aggregation, this preserves the uniform weighting of retained prompts used in the allocation analysis.

Training procedure. Algorithm 1 summarizes the training procedure. For each candidate batch, we combine the rollout allocation in Section 3.1 with the observability safeguard in Section 3.2 to select the retained set and determine its integer rollout counts under the original budget. If any candidate prompt lacks an available estimate, the batch uses uniform counts N_0. We then generate the assigned responses, compute centered advantages, and apply the response weights before the policy update. Further implementation details are provided in Appendix C.

4 Experiments

We evaluate SERA through controlled gradient comparisons, sequence generation, and mathematical reasoning. We begin with ImageNet (Deng et al., 2009) classification (Section 4.1), where exact maximum-likelihood gradients provide a controlled reference for assessing finite-rollout updates. Using exact success probabilities for allocation, we isolate the effect of rollout allocation from success-probability estimation error and examine both gradient alignment and held-out performance. We then study maze navigation with a fixed training set (Section 4.2), extending the comparison to sequence generation with binary verifier-based rewards. Finally, we evaluate mathematical reasoning with SmolLM2-360M-Instruct (Allal et al., 2025) (Section 4.3). We further evaluate Qwen2.5-Math-1.5B (Yang et al., 2024b) and Qwen3-4B-Base (Yang et al., 2025a) (Section 4.4) to examine performance at larger model scales and across a broader range of mathematical reasoning benchmarks.

We use MaxRL (Tajwar et al., 2026) as our primary baseline to assess the performance gains of SERA. We match the total training rollout budget between the two methods and use the same loss normalization within each comparison. On SmolLM2-360M-Instruct, we additionally compare against GRPO (Shao et al., 2024), PKPO (Walder & Karkhanis, 2025), and the rollout allocation method DARS-HW (Yang et al., 2025b). We report Pass@1 and multi-sample Pass@K to assess single-sample accuracy and solution coverage, respectively.

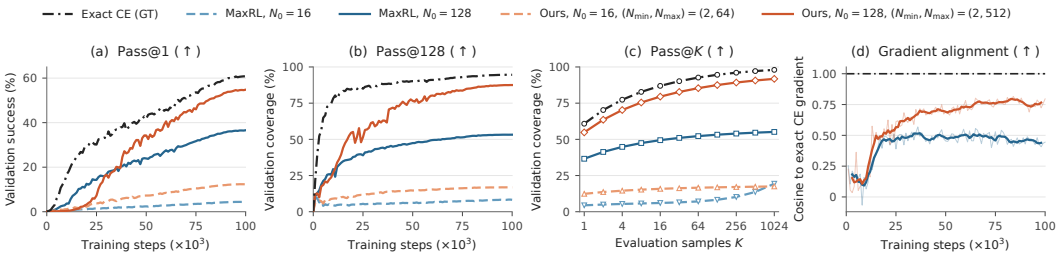

Figure 2: **Oracle rollout allocation on ImageNet.** MaxRL and SERA (Ours) are compared under matched rollout budgets. (a,b) Validation Pass@1 and Pass@128 during training. (c) Pass@K at step 100,000. Black curves in (a–c) show the performance of a model trained directly with cross-entropy (CE). (d) Sampled-to-exact CE gradient cosine at $N_0 = 128$, measured along each method's own training trajectory. Faint and thick lines in (d) show raw measurements and centered five-point moving averages, respectively.

4.1 Exact-Likelihood Reference on ImageNet

We first evaluate how rollout allocation affects the fidelity of finite-rollout updates to the exact maximum-likelihood gradient. Following MaxRL, we consider ImageNet classification with a ResNet-50 (He et al., 2016) classifier. Each rollout samples a class label and receives a binary correctness reward. In this setting, minimizing cross-entropy (CE) exactly optimizes the log-success objective, providing a reference without finite-rollout estimation error. We compare uniform-rollout MaxRL and SERA under matched rollout budgets, alongside direct CE training. Our allocation uses the exact success probabilities available from the classifier, while policy updates still rely on sampled labels and rewards. This controlled setting isolates the effect of rollout allocation from success-probability estimation error.

Figures 2 (a,b) show validation Pass@1 and Pass@128 during training. With the larger rollout budget, SERA progressively separates from MaxRL and narrows the gap to direct CE training in both metrics. The advantage persists during later training, demonstrating sustained improvements in both single-sample performance and solution coverage. With the smaller budget, SERA also improves Pass@1 and Pass@128. Figure 2 (c) shows the final Pass@K profiles. At the larger training budget, SERA outperforms MaxRL across all displayed evaluation budgets, extending the gains beyond Pass@1 and Pass@128. At the smaller training budget, SERA leads over most of the displayed range. The larger training budget thus produces the broadest coverage gains.

Figure 2 (d) compares the realized sampled training-loss gradient with the exact CE gradient, computed at the same policy parameters and from the same forward pass within each run. The CE reference covers the full candidate batch. At $N_0 = 128$, SERA achieves a mean cosine similarity of 0.6266 between the sampled training-loss gradient and the exact CE gradient, compared with 0.4317 for MaxRL, with the advantage persisting into later training. These measurements provide direct gradient-level evidence of stronger alignment with the likelihood gradient along the observed training trajectories. Together with the validation results, they show that SERA improves both sampled-gradient alignment and held-out performance under matched rollout budgets. Further experimental details and additional results are provided in Appendices D.1.1 and D.2.1.

4.2 Maze Navigation with a Lightweight Transformer

We next evaluate SERA on maze navigation with verifier-based rewards. Following MaxRL, we use a lightweight decoder-only Transformer based on the Qwen2 architecture (Yang et al., 2024a) to solve 17×17 mazes. Given a maze, the model generates navigation actions and receives a binary reward for reaching the goal without violating the maze constraints. We initialize RL training from the same pretrained checkpoint used in MaxRL and train on a fixed subset of mazes over multiple epochs, evaluating on a separate held-out set. We compare MaxRL and SERA under matched rollout budgets using both token-mean normalization and fixed-length sequence normalization (Seqnorm). For rollout allocation, SERA uses historical success-probability estimates from previous epochs, as described in Section 3.3.

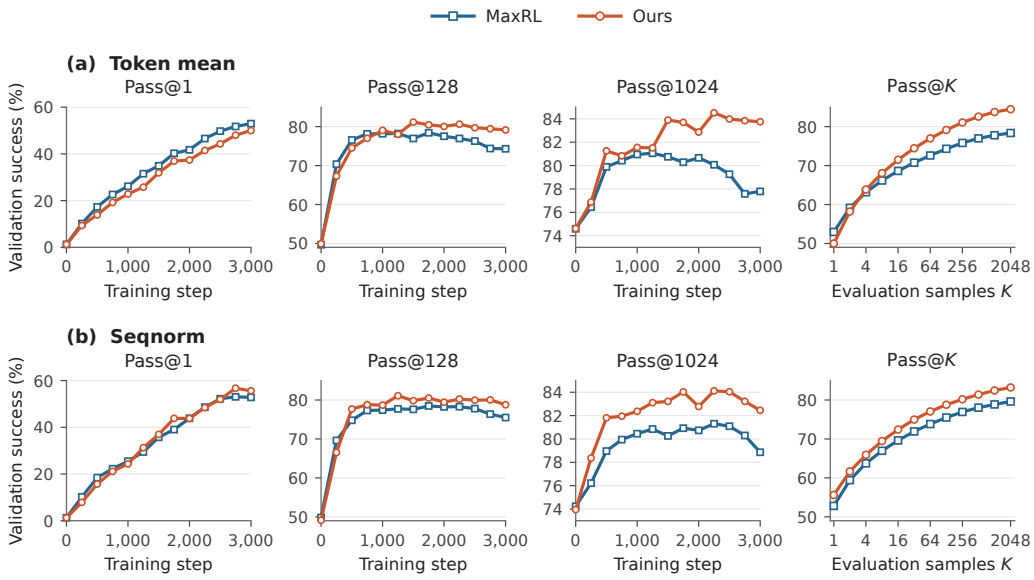

Figure 3: **Maze navigation.** MaxRL and SERA (Ours) under (a) token-mean and (b) fixed-length sequence normalization (Seqnorm). Columns show Pass@1, Pass@128, and Pass@1024 during training, followed by Pass@K at step 3,000.

Figure 3 shows Pass@1, Pass@128, and Pass@1024 during training, alongside the final Pass@K profiles. Under both normalizations, solution coverage initially improves for both methods. During later training, MaxRL's high-budget coverage declines, while SERA mitigates this decline and retains higher Pass@128 and Pass@1024. The final Pass@K profiles provide a broader assessment of maze-solving performance across the evaluated range of sampling budgets. Under token-mean normalization (Figure 3 (a)), SERA achieves higher Pass@K than MaxRL at larger evaluation budgets, with the advantage growing as K increases. Under Seqnorm (Figure 3 (b)), SERA outperforms MaxRL across the entire displayed Pass@K profile. Together, these results demonstrate that our method improves high-budget solution coverage under the same rollout budget, extending the benefits observed on ImageNet to sequence generation with verifier-based rewards. Further experimental details and additional results are provided in Appendices D.1.2 and D.2.2.

4.3 Mathematical Reasoning with SmolLM2

We next evaluate SERA on mathematical reasoning with SmolLM2-360M-Instruct (Allal et al., 2025). Following MaxRL, we train the model on GSM8K (Cobbe et al., 2021) with binary rewards and evaluate on GSM8K-Platinum (Vendrow et al., 2025). We use a reference rollout count of $N_0 = 16$, with SERA allocating at most $N_{\max} = 64$ rollouts per prompt. We compare MaxRL and SERA under matched training rollout budgets using both token-mean normalization and fixed-length sequence normalization (Seqnorm).

Table 1 reports Pass@1, Pass@32, Pass@128, and Pass@256 at training step 2,000, with the initial model included as a pre-RL reference. Under both normalizations, SERA improves Pass@32, Pass@128, and Pass@256 over MaxRL, with little change in Pass@1. The consistent gains across these evaluation budgets demonstrate improved solution coverage from redistributing the same number of training responses. We further compare SERA with additional RL and rollout allocation baselines. SERA outperforms GRPO (Shao et al., 2024) at every reported K. PKPO (Walder & Karkhanis, 2025) achieves higher Pass@K at large K, but substantially lower Pass@1 than SERA. We also include DARS-HW (Yang et al., 2025b) as a closely related allocation baseline. Our allocation is derived specifically from MaxRL's finite-sample gradient scaling and enforces a fixed total rollout budget, whereas DARS-HW imposes per-prompt limits without fixing the batch-level total. In this comparison, SERA achieves higher Pass@32, Pass@128, and Pass@256 than DARS-HW using fewer training responses. DARS-HW attains higher Pass@1 and consumes approximately

Table 1: **SmolLM2-360M-Instruct on GSM8K-Platinum.** All RL methods are evaluated at step 2,000, with the base model as the pre-RL reference. Values are percentages; Δ reports percentage-point changes relative to MaxRL under the same normalization. †: DARS-HW uses 1.49× as many training rollouts as the other methods.

Method	Pass@1	Pass@32	Pass@128	Pass@256
Base model	6.80	53.85	76.32	84.37
GRPO (Shao et al., 2024)	30.46	45.80	49.28	50.62
PKPO (T=16) (Walder & Karkhanis, 2025)	19.93	70.32	84.92	90.16
MaxRL (token-mean) (Tajwar et al., 2026)	32.60	59.41	68.44	72.21
MaxRL (Seqnorm) (Tajwar et al., 2026)	32.08	56.66	64.58	68.24
DARS-HW† (Yang et al., 2025b)	34.25	52.38	57.31	59.06
SERA (token-mean)	33.01	63.38	73.17	77.34
Δ vs. MaxRL (token-mean)	+0.41	+3.97	+4.73	+5.13
SERA (Seqnorm)	32.00	61.35	70.12	73.37
Δ vs. MaxRL (Seqnorm)	−0.08	+4.69	+5.54	+5.13

1.49× as many training rollouts as other methods over the same number of training steps. These results demonstrate the effectiveness of SERA in improving mathematical reasoning coverage under a fixed training rollout budget. Further experimental details and additional results are provided in Appendices D.1.3 and D.2.3.

4.4 Mathematical Reasoning with Qwen

We further evaluate SERA on mathematical reasoning with Qwen2.5-Math-1.5B (Yang et al., 2024b) and Qwen3-4B-Base (Yang et al., 2025a). For Qwen2.5-Math-1.5B, we train on MATH level 3–5 problems (Hendrycks et al., 2021). Evaluation covers BeyondAIME (ByteDance-Seed, 2025), AIME 2025 (Zhang & Team, 2025), MATH-500 (Lightman et al., 2024), and OlympiadBench (He et al., 2024). For Qwen3-4B-Base, we train on POLARIS-53K (An et al., 2025), following the MaxRL training setup. We evaluate our checkpoint at step 1,000 against the publicly released GRPO and MaxRL checkpoints at the same training step (Tajwar et al., 2026). All three checkpoints are evaluated using the same sampling and scoring protocol on BeyondAIME, AIME 2025, MATH-500, and Minerva Math (Lewkowycz et al., 2022).

Figure 4 compares the Pass@K profiles of Qwen2.5-Math-1.5B and Qwen3-4B. Across both model scales, SERA achieves higher Pass@K than GRPO at the largest displayed K on all eight model-benchmark pairs. Relative to MaxRL, SERA consistently improves the high-budget coverage of Qwen2.5-Math-1.5B, attaining higher Pass@K at every reported $K \geq 32$ on all four benchmarks while keeping Pass@1 within 0.22 percentage points. The gains increase with the evaluation budget on BeyondAIME and AIME 2025, reaching 7.00 and 6.67 percentage points at $K = 512$, respectively. On Qwen3-4B, SERA improves Pass@1 on all four benchmarks and delivers substantial Pass@512 gains of 8.00 points on BeyondAIME and 4.78 points on Minerva Math. It matches MaxRL at $K = 512$ on AIME 2025. On MATH-500, MaxRL and SERA already achieve approximately 97–98% coverage at $K = 256$ across both model scales, leaving limited headroom for absolute improvements. Overall, these results show that SERA transfers across model scales and training datasets, improving the single-sample to multi-sample performance profile under matched training rollout budgets, with the largest gains on benchmarks that retain substantial high-budget headroom. Further experimental details and additional results are provided in Appendices D.1.4 and D.2.4.

5 Related Work

RL objectives and reasoning coverage. GRPO (Shao et al., 2024) and DeepSeek-R1 (Guo et al., 2025) established group-based reinforcement learning with verifiable rewards as an effective paradigm for reasoning models. Subsequent work studies how the objective and its estimator shape the update, including normalization effects in Dr. GRPO (Liu et al., 2025), sequence-level ra-

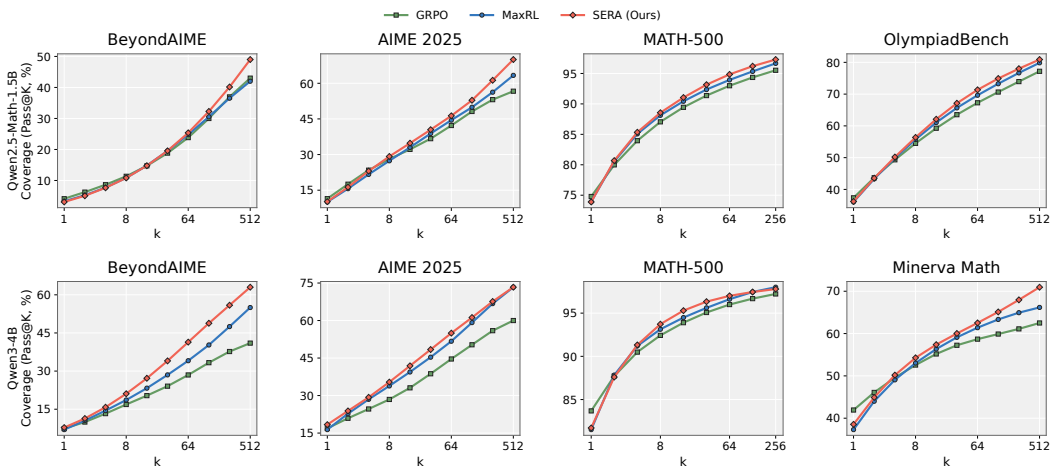

Figure 4: **Pass@K profiles for Qwen models.** Top: Qwen2.5-Math-1.5B at step 600. Bottom: Qwen3-4B-Base at step 1,000. Both rows compare GRPO, MaxRL, and SERA.

tios in GSPO (Zheng et al., 2025a), and transformed success-probability objectives (Davis & Recht, 2025). These choices matter for repeated sampling: Pass@K measures multi-sample solution coverage, and improvements in Pass@1 need not improve coverage at larger K (Chen et al., 2021; Yue et al., 2025; Barakat et al., 2026). Inference-aware objectives such as PKPO (Walder & Karkhanis, 2025) and diversity-oriented objectives such as SetPO (Li et al., 2026) and UCPO (Lochab et al., 2026) explicitly target multi-sample or diversity-sensitive behavior. MaxRL (Tajwar et al., 2026), in contrast, targets prompt-wise log-success using a centered finite-rollout estimator. SERA keeps the same log-success target and instead reallocates a fixed rollout budget to control the prompt-dependent finite-rollout scaling of the centered estimator.

Adaptive rollout allocation and selection. Recent methods allocate or filter rollouts using different notions of value: Knapsack RL (Li et al., 2025) uses estimated learning gain, VIP (Nguyen et al., 2026) and DynaMO (Fang et al., 2026) use variance-related signals, and HORA (Wang et al., 2026) optimizes the probability of finding successful responses. DARS-HW (Yang et al., 2025b) gives a likelihood-oriented inverse-success schedule, while Reinforce-Ada (Xiong et al., 2025) derives adaptive sampling and weighting from nonlinear objectives. Other work makes decisions through selection, replay, or sequential continuation, including DAPO (Yu et al., 2025), Pilot-Commit (Kim et al., 2026), and SARA (Nomand et al., 2026b). SERA differs in the quantity it controls: for a fixed retained set, it solves a bounded max–min problem over the finite-rollout scaling coefficients of the centered MaxRL estimator under a fixed total rollout budget. Historical estimates determine the inputs to this allocation, and multiplicity correction removes the additional response-count weighting induced by unequal group sizes. Appendix A gives the detailed comparison.

6 Conclusion

We considered the problem of realizing MaxRL's likelihood objective under a fixed rollout budget. Our theoretical analysis shows that uniform rollout counts induce unequal attenuation of prompt-wise likelihood gradients, more strongly attenuating gradients from low-success prompts and potentially shifting the expected update direction. Building on this analysis, we proposed SERA, which redistributes the available rollouts to approximately equalize finite-rollout gradient scaling across retained prompts while correcting the additional prompt weighting introduced by unequal rollout counts. Experiments show stronger alignment with exact likelihood gradients on ImageNet and improved multi-sample solution coverage over MaxRL on maze navigation and mathematical reasoning under matched training rollout budgets. On the mathematical reasoning benchmarks, Pass@1 remains comparable or improves while multi-sample coverage increases. Overall, SERA provides a principled way to redistribute a fixed rollout budget so as to better preserve MaxRL's intended gradient structure while improving downstream multi-sample coverage.

AI USE STATEMENT

Generative AI tools were used to assist with language editing and to improve the clarity and readability of the paper. All AI-assisted edits were reviewed and verified by the authors. The authors take full responsibility for the final content of this paper.

REPRODUCIBILITY STATEMENT

We provide the information required to reproduce the theoretical and empirical results of this work. The complete derivations and proofs of the theoretical results are provided in the appendix. Detailed experimental settings, including datasets, model architectures, training configurations, optimization hyperparameters, rollout budgets, evaluation protocols, and computational resources, are reported in the Experimental Details appendix. Additional results and training diagnostics are also provided in the appendix. Our implementation follows the algorithm described in the main paper, and the code necessary to reproduce the experiments is provided as supplementary material.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce-style optimization for learning from human feedback in llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, 2024.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, et al. Smollm2: When smol goes big–data-centric training of a small language model. *arXiv preprint arXiv:2502.02737*, 2025.

Chenxin An, Zhihui Xie, Xiaonan Li, Lei Li, Jun Zhang, Shansan Gong, Ming Zhong, Jingjing Xu, Xipeng Qiu, Mingxuan Wang, and Lingpeng Kong. Polaris: A post-training recipe for scaling reinforcement learning on advanced reasoning models, 2025. URL https://hkunlp.github.io/blog/2025/Polaris.

Farid Bagirov, Mikhail Arkhipov, Ksenia Sycheva, Evgeniy Glukhov, and Egor Bogomolov. The best of n worlds: Aligning reinforcement learning with best-of-n sampling via max@ k optimisation. *arXiv preprint arXiv:2510.23393*, 2025.

Ryan Bahlous-Boldi, Isha Puri, Idan Shenfeld, Akarsh Kumar, Mehul Damani, Sebastian Risi, Omar Khattab, Zhang-Wei Hong, and Pulkit Agrawal. Vector policy optimization: Training for diversity improves test-time search. *arXiv preprint arXiv:2605.22817*, 2026.

Anas Barakat, Souradip Chakraborty, Khushbu Pahwa, and Amrit Singh Bedi. Why pass@ k optimization can degrade pass@ 1: Prompt interference in llm post-training. *arXiv preprint arXiv:2602.21189*, 2026.

ByteDance-Seed. BeyondAIME: Advancing math reasoning evaluation beyond high school olympiads. https://huggingface.co/datasets/ByteDance-Seed/BeyondAIME, 2025. Hugging Face dataset.

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

Zhipeng Chen, Xiaobo Qin, Youbin Wu, Yue Ling, Qinghao Ye, Wayne Xin Zhao, and Guang Shi. Pass@ k training for adaptively balancing exploration and exploitation of large reasoning models. *arXiv preprint arXiv:2508.10751*, 2025.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Damek Davis and Benjamin Recht. What is the objective of reasoning with reinforcement learning? *arXiv preprint arXiv:2510.13651*, 2025.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Yangyi Fang, Jiaye Lin, Xiaoliang Fu, Cong Qin, Haolin Shi, Chaowen Hu, Lu Pan, Ke Zeng, and Xunliang Cai. How to allocate, how to learn? dynamic rollout allocation and advantage modulation for policy optimization. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 14727–14744, 2026.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, 2024.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

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

Heyang Jiang, Henry Liu, and Baharan Mirzasoleiman. Learning as reasoning unfolds: Progressive rollout allocation for efficient reinforcement learning. *arXiv preprint arXiv:2607.22002*, 2026.

Woojeong Kim, Ziyi Yang, Jing Nathan Yan, and Jialu Liu. Spend your rollouts where it counts: Rollout allocation for group-based rl post-training. *arXiv preprint arXiv:2605.26606*, 2026.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

Chenyi Li, Yuan Zhang, Bo Wang, Guoqing Ma, Wei Tang, Haoyang Huang, and Nan Duan. Setpo: Set-level policy optimization for diversity-preserving llm reasoning. *arXiv preprint arXiv:2602.01062*, 2026.

Ziniu Li, Congliang Chen, Tianyun Yang, Tian Ding, Ruoyu Sun, Ge Zhang, Wenhao Huang, and Zhi-Quan Luo. Knapsack rl: Unlocking exploration of llms via optimizing budget allocation. *arXiv preprint arXiv:2509.25849*, 2025.

Mengqi Liao, Xiangyu Xi, Chen Ruinian, Jia Leng, Yangen Hu, Ke Zeng, Shuai Liu, and Huaiyu Wan. Enhancing efficiency and exploration in reinforcement learning for llms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 1451–1463, 2025.

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*, volume 2024, pp. 39578–39601, 2024.

Zhizhao Liu, Zhiliang Tian, Xi Wang, Zhihua Wen, Yihang Xiong, Zhiquan Lai, and Dongsheng Li. Efficient rlvr scheduling via graph-structured online difficulty estimation. *arXiv preprint arXiv:2608.17941*, 2026.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

Anamika Lochab, Bolian Li, and Ruqi Zhang. Uniform-correct policy optimization: Breaking rlvr's indifference to diversity. *arXiv preprint arXiv:2605.00365*, 2026.

Hieu Trung Nguyen, Bao Nguyen, Wenao Ma, Yuzhi Zhao, Ruifeng She, and Viet Anh Nguyen. Adaptive rollout allocation for online reinforcement learning with verifiable rewards. In *International Conference on Learning Representations*, volume 2026, pp. 136100–136132, 2026.

Pixel Nomand, Elena Voss, Marcus Hale, and Sofia Reyes. Pair: Pairwise-aware inclusion reweighting for adaptive rollout allocation in rlvr, 2026a. URL https://arxiv.org/abs/2608.11368.

Pixel Nomand, Elena Voss, Marcus Hale, and Sofia Reyes. Early verdicts, better budgets: Sequential adaptive rollout allocation for compute-efficient rlvr, 2026b. URL https://arxiv.org/abs/2607.26253.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

Yun Qu, Qi Wang, Yixiu Mao, Vincent Tao Hu, Björn Ommer, and Xiangyang Ji. Can prompt difficulty be online predicted for accelerating rl finetuning of reasoning models? In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pp. 1240–1250, 2026.

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

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

Yifan Sun, Jingyan Shen, Yibin Wang, Tianyu Chen, Zhendong Wang, Mingyuan Zhou, and Huan Zhang. Improving data efficiency for llm reinforcement fine-tuning through difficulty-targeted online data selection and rollout replay. *Advances in Neural Information Processing Systems*, 38: 173449–173476, 2025.

Fahim Tajwar, Guanning Zeng, Yueer Zhou, Yuda Song, Daman Arora, Yiding Jiang, Jeff Schneider, Ruslan Salakhutdinov, Haiwen Feng, and Andrea Zanette. Maximum likelihood reinforcement learning. *arXiv preprint arXiv:2602.02710*, 2026.

Yunhao Tang, Kunhao Zheng, Gabriel Synnaeve, and Rémi Munos. Optimizing language models for inference time objectives using reinforcement learning. *arXiv preprint arXiv:2503.19595*, 2025.

Joshua Vendrow, Edward Vendrow, Sara Beery, and Aleksander Madry. Do large language model benchmarks test reliability? *arXiv preprint arXiv:2502.03461*, 2025.

Christian Walder and Deep Tejas Karkhanis. Pass@ k policy optimization: Solving harder reinforcement learning problems. *Advances in Neural Information Processing Systems*, 38:152416–152445, 2025.

Tao Wang, Shuo Li, Yan Sun, Dongsheng Ding, and Edgar Dobriban. Where to spend rollouts: Hit-utility optimal rollout allocation for group-based rlvr. *arXiv preprint arXiv:2605.07114*, 2026.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Wei Xiong, Chenlu Ye, Baohao Liao, Hanze Dong, Xinxing Xu, Christof Monz, Jiang Bian, Nan Jiang, and Tong Zhang. Reinforce-ada: An adaptive sampling framework under non-linear rl objectives. *arXiv preprint arXiv:2510.04996*, 2025.

Yixuan Even Xu, Yash Savani, Fei Fang, and J Zico Kolter. Not all rollouts are useful: Down-sampling rollouts in llm reinforcement learning. *arXiv preprint arXiv:2504.13818*, 2025.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report, 2024a. URL https://arxiv.org/abs/2407.10671.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024b.

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

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. *arXiv preprint arXiv:2508.13755*, 2025b.

Jiarui Yao, Yifan Hao, Hanning Zhang, Hanze Dong, Wei Xiong, Nan Jiang, and Tong Zhang. Optimizing chain-of-thought reasoners via gradient variance minimization in rejection sampling and rl. *Advances in Neural Information Processing Systems*, 38:163245–163284, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *Advances in Neural Information Processing Systems*, 38:113222–113244, 2025.

Yang Yu. Pass@ k metric for rlvr: A diagnostic tool of exploration, but not an objective. *arXiv preprint arXiv:2511.16231*, 2025.

Youngjun Yu, Sanghwan Jang, and Hwanjo Yu. Difficulty-adaptive tree-structured policy optimization for expanding reasoning coverage in rlvr. *arXiv preprint arXiv:2609.08650*, 2026.

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?, 2025. URL https://arxiv.org/abs/2504.13837.

Yifan Zhang and Math-AI Team. American invitational mathematics examination (aime) 2025, 2025. URL https://huggingface.co/datasets/math-ai/aime25.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization, 2025a. URL https://arxiv.org/abs/2507.18071.

Haizhong Zheng, Yang Zhou, Brian Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. Act only when it pays: Efficient reinforcement learning for llm reasoning via selective rollouts. *Advances in Neural Information Processing Systems*, 38:124321–124346, 2025b.

Yiming Zong, Yige Wang, and Jiashuo Jiang. Cross-epoch adaptive rollout optimization for rl post-training. *arXiv preprint arXiv:2606.05606*, 2026.

CONTENTS OF THE APPENDIX

A **Extended Related Work** 14

B **Additional Theory** 16

B.1 Finite-Rollout Fidelity of the Centered Estimator 16

B.2 Conditional Objectives and Batch-Gradient Direction 17

B.3 Derivation of the Fixed-Budget Allocation 18

B.4 Mixed-Group Observability and Budget Concentration 19

B.5 Comparison with Inverse-Fidelity Reweighting 19

C **Implementation Details** 19

C.1 Historical Success-Probability Estimates 19

C.2 Bounded Allocation and Exact-Budget Integerization 20

C.3 Active-Prefix Search and Capacity Fallback 21

C.4 Loss Aggregation and Multiplicity Correction 21

D **Extended Experimental Details** 22

D.1 Experimental Setup 22

D.1.1 ImageNet 23

D.1.2 Maze 23

D.1.3 SmolLM2 on GSM8K 24

D.1.4 Qwen 25

D.2 More Experimental Results 27

D.2.1 ImageNet 27

D.2.2 Maze 28

D.2.3 SmolLM2 on GSM8K 28

D.2.4 Mathematical reasoning with Qwen 29

D.2.5 Training Dynamics and Allocation Behavior 31

A EXTENDED RELATED WORK

Reinforcement learning for reasoning. Language-model post-training includes preference-based optimization with PPO (Schulman et al., 2017; Ouyang et al., 2022) and REINFORCE-style methods without a learned value function (Ahmadian et al., 2024). For reasoning with verifiable rewards, GRPO (Shao et al., 2024) constructs advantages from sampled response groups, and DeepSeek-R1 (Guo et al., 2025) demonstrates the effectiveness of this paradigm at scale. The update depends on both the objective and its gradient estimator: Dr. GRPO (Liu et al., 2025) analyzes biases introduced by loss and advantage normalization, while GSPO (Zheng et al., 2025a) uses sequence-level importance ratios and clipping. Davis & Recht (2025) further interprets binary-reward updates through transformed success-probability objectives.

Multi-sample evaluation introduces an additional distinction between single-sample accuracy and solution coverage. Pass@K estimates the probability that at least one of K sampled responses is correct (Chen et al., 2021). RL can improve Pass@1 while reducing coverage at larger sampling budgets (Yue et al., 2025); analyses of prompt interference and training objectives help explain why gains across evaluation budgets need not coincide (Barakat et al., 2026; Yu, 2025). Tang et al.

(2025) formulate objectives for inference-time aggregation, including Pass@K and majority voting. PKPO (Walder & Karkhanis, 2025) and Pass@K training (Chen et al., 2025) target multi-sample success, while max@K optimization (Bagirov et al., 2025) targets expected maximum reward.

Another line of work explicitly promotes diversity among solutions. SetPO (Li et al., 2026) incorporates each trajectory's contribution to set-level diversity, UCPO (Lochab et al., 2026) regularizes the conditional distribution over correct responses, and VPO (Bahlous-Boldi et al., 2026) encourages specialization across reward scalarizations. These training interventions are distinct from inference-time answer selection through self-consistency or majority voting (Wang et al., 2022; Lewkowycz et al., 2022), or through learned outcome and process verifiers (Cobbe et al., 2021; Lightman et al., 2024).

MaxRL (Tajwar et al., 2026) develops compute-indexed approximations to prompt-wise log-success maximization and characterizes the finite-rollout behavior of its practical centered estimator. SERA adopts this likelihood target and studies how rollout allocation controls the relative scaling of prompt gradients under a fixed rollout budget. Pass@K profiles assess the resulting solution coverage across inference budgets, while the allocation objective concerns the finite-rollout approximation to the log-success gradient.

Adaptive rollout allocation. Adaptive allocation methods differ in how they value additional responses, which prompts and responses they retain, and when they make sampling decisions. Liao et al. (2025) combine difficulty-based allocation with temperature scheduling. Knapsack RL (Li et al., 2025) combines mixed-reward probability with estimated learning gain. GVM-RAFT (Yao et al., 2025) studies variance-based allocation for rejection sampling and extends its framework to GRPO, while VIP (Nguyen et al., 2026) predicts success probabilities to minimize expected gradient variance. DynaMO (Fang et al., 2026) uses historical Bernoulli reward variance as an allocation proxy alongside token-level advantage modulation. HORA (Wang et al., 2026) maximizes the sum of posterior hit probabilities for additional rollouts, and CERO (Zong et al., 2026) uses concave utilities of cumulative budgets across epochs. These criteria emphasize learning gain, variance reduction, successful-response discovery, or utility over a training horizon.

Selection methods act at different stages of generation and optimization. DAPO (Yu et al., 2025) filters identical-reward groups and continues sampling to collect informative prompts. GRESO (Zheng et al., 2025b) uses historical outcomes to skip likely uninformative prompts before generation, while DOTS (Sun et al., 2025) combines difficulty-targeted selection with rollout replay. POLARIS (An et al., 2025) reuses successful responses to rescue all-failure groups. PODS (Xu et al., 2025) instead selects a subset of already generated responses to reduce policy-update cost. These approaches distinguish avoiding generation, reusing responses, and reducing the update batch.

Success-probability estimation for rollout scheduling commonly uses pilot samples, historical statistics, or predictive models. Pilot-based methods estimate prompt difficulty from a small set of fresh rollouts before allocating additional rollouts, as in DARS (Yang et al., 2025b). History-based methods reuse outcomes from earlier visits or epochs: Knapsack RL (Li et al., 2025) uses success rates observed in the previous epoch, while the Ada-EMA variant of Reinforce-Ada (Xiong et al., 2025) maintains discounted success and trial counts with Bayesian smoothing. MoPPS (Qu et al., 2026) also uses streaming Bayesian estimates to track prompt success rates. Predictive models additionally exploit relationships among prompts: VIP (Nguyen et al., 2026) uses a Gaussian process over prompt embeddings with recent rollout feedback, while graph-structured estimation (Liu et al., 2026) shares observations across related prompts. These approaches supply the estimates $\widehat{p}_q$; the allocation criterion determines how those estimates translate into rollout counts.

Sequential allocation additionally uses outcomes observed during rollout collection. Pilot-Commit (Kim et al., 2026) screens prompts with pilot rollouts before committing further budget. VIGOR (Jiang et al., 2026) progressively expands sampling for prompts with high observed reward variance, while SARA (Nomand et al., 2026b) uses sequential evidence to commit or abandon groups. DATPO (Yu et al., 2026) adapts generation through difficulty-dependent tree budgets and entropy-based forking. Adaptive continuation also changes which gradient terms become observable. PAIR (Nomand et al., 2026a) uses inverse joint inclusion probabilities to recover the complete-candidate pair gradient in expectation for an unclipped, unstandardized leave-one-out estimator under its stated sampling assumptions.

The DARS-HW variant of DARS (Yang et al., 2025b) and Reinforce-Ada (Xiong et al., 2025) are particularly close to SERA through their likelihood-oriented formulations. DARS uses preliminary outcomes to allocate further rollouts to difficult prompts; its HW schedule is the variant whose idealized prompt-level target is maximum likelihood. Under the population-baseline approximation used in its analysis, the idealized schedule $N_q \propto 1/p_q$ makes the summed reward-gradient contribution $N_q \nabla_\theta p_q$ proportional to $\nabla_\theta \log p_q$. Reinforce-Ada derives the weight $f'(p_q)$ from a nonlinear objective $f(p_q)$ and realizes it through sampling, explicit gradient weighting, or a combination of the two, including $f(p) = \log p$. These likelihood-oriented formulations share MaxRL's log-success target. The comparison with SERA therefore concerns the estimator being controlled and the role assigned to rollout allocation.

SERA starts from the practical centered MaxRL estimator in Equation 2. Let $g_q = \nabla_\theta \log p_q$ and $\kappa_q = 1 - (1 - p_q)^{N_q - 1}$, so that $\mathbb{E}[\widehat{g}_q^{(N_q)}] = \kappa_q g_q$ for a fixed count. Consider a candidate batch of size B, a retained set $\mathcal{A}$, and a total rollout budget $M = BN_0$. Let $\mathcal{F}$ denote the current policy, candidate batch, historical information, and allocation decisions fixed before fresh on-policy sampling. Under response-mean sequence-score aggregation, the conditional means without and with the per-response weight N_0/N_q are

$$\mathbb{E}[\widehat{G}_{\text{plain}} \mid \mathcal{F}] = \frac{1}{B} \sum_{q \in \mathcal{A}} \frac{N_q}{N_0} \kappa_q g_q, \qquad \mathbb{E}[\widehat{G}_{\text{corr}} \mid \mathcal{F}] = \frac{1}{B} \sum_{q \in \mathcal{A}} \kappa_q g_q. \tag{11}$$

The correction removes the additional response-count weighting, while allocation controls the remaining finite-rollout coefficients (Appendix C.4). For a fixed retained set and frozen success estimates, the bounded continuous allocator maximizes $\min_{q \in \mathcal{A}} \kappa(\widehat{p}_q, x_q)$ subject to $\sum_{q \in \mathcal{A}} x_q = M$ and $N_{\min} \le x_q \le N_{\max}$. Its interior waterline rule satisfies $x_q - 1 = c_{\mathcal{B}}/[-\log(1 - \widehat{p}_q)]$ (Appendix B.3), with inverse-success behavior arising as a small-$\widehat{p}_q$ approximation. The practical procedure converts these counts to integers while preserving the budget and bounds. Thus, SERA allocates samples to control the finite-rollout scaling of an already likelihood-oriented estimator and explicitly removes the additional weighting caused by unequal group sizes. Together, these components target comparable scaling of prompts' log-success gradients under a fixed rollout budget (Appendix B.2).

B Additional Theory

B.1 Finite-Rollout Fidelity of the Centered Estimator

Following the finite-rollout analysis of MaxRL (Tajwar et al., 2026), we derive the expectation of the centered sequence-score estimator used in our analysis. Fix a prompt q and draw $N \ge 1$ i.i.d. responses from $\pi_\theta(\cdot \mid q)$, with N chosen before sampling. The verifier is binary and does not explicitly depend on θ. Suppressing the prompt subscript, write $p = \mathbb{E}[R] \in (0, 1)$, $s = \nabla_\theta \log \pi_\theta(y \mid q)$, and $g = \nabla_\theta \log p$. Under the usual score-function regularity conditions,

$$\mathbb{E}[s] = 0, \qquad \mathbb{E}[Rs] = \nabla_\theta p = pg. \tag{12}$$

It follows that

$$\mu_1 := \mathbb{E}[s \mid R = 1] = g, \qquad \mu_0 := \mathbb{E}[s \mid R = 0] = -\frac{p}{1 - p} g. \tag{13}$$

Finite-rollout expectation. The centered estimator in Equation 2 satisfies

$$\mathbb{E}[\widehat{g}^{(N)}] = \kappa(p, N) g, \qquad \kappa(p, N) := 1 - (1 - p)^{N-1}. \tag{14}$$

Proof. Let $C := \sum_{i=1}^{N} R_i \sim \text{Binomial}(N, p)$. For $C > 0$, the estimator can be written as

$$\widehat{g}^{(N)} = \frac{1}{C} \sum_{i=1}^{N} R_i s_i - \frac{1}{N} \sum_{i=1}^{N} s_i. \tag{15}$$

Conditional on $C = k > 0$, the successful and unsuccessful responses have expected scores μ_1 and μ_0, respectively. Hence,

$$\begin{aligned}\mathbb{E}[\widehat{g}^{(N)} \mid C = k] &= \mu_1 - \frac{k\mu_1 + (N-k)\mu_0}{N} \\ &= \frac{N-k}{N(1-p)} g. \end{aligned} \tag{16}$$

Since the estimator is zero when $C = 0$,

$$\begin{aligned}\mathbb{E}[\widehat{g}^{(N)}] &= \frac{\mathbb{E}[(1 - C/N)\mathbf{1}_{\{C>0\}}]}{1-p} g \\ &= \frac{1 - p - (1-p)^N}{1-p} g \\ &= [1 - (1-p)^{N-1}] g. \end{aligned} \tag{17}$$

B.2 Conditional Objectives and Batch-Gradient Direction

Let $\mathcal{B}$ be a candidate batch with $B := |\mathcal{B}|$, and let $\mathcal{A} \subseteq \mathcal{B}$ be a nonempty retained set. We condition on the history, current policy, candidate batch, and the set and integer counts chosen before the current rollouts; denote this information by $\mathcal{F}$. The set $\mathcal{A}$ and counts N_q are held fixed when differentiating with respect to θ. Write $g_q := \nabla_\theta \log p_q$ and $\kappa_q := \kappa(p_q, N_q)$.

A conditional objective for heterogeneous counts. For an integer $m \geq 0$, define

$$\Phi_m(p) := -\sum_{j=1}^{m} \frac{(1-p)^j}{j}, \qquad \Phi_0(p) := 0. \tag{18}$$

Differentiation and the log-series identity give

$$\Phi_m'(p) = \frac{1 - (1-p)^m}{p}, \qquad \lim_{m\to\infty} \Phi_m(p) = \log p. \tag{19}$$

For the current retained set and counts, define

$$J_{\mathcal{A},\mathbf{N}}(\theta) := \frac{1}{B} \sum_{q \in \mathcal{A}} \Phi_{N_q - 1}(p_q(\theta)). \tag{20}$$

The finite-rollout identity then implies

$$\begin{aligned}\nabla_\theta J_{\mathcal{A},\mathbf{N}}(\theta) &= \frac{1}{B} \sum_{q \in \mathcal{A}} \kappa_q g_q \\ &= \mathbb{E}[\widehat{G} \mid \mathcal{F}], \qquad \widehat{G} := \frac{1}{B} \sum_{q \in \mathcal{A}} \widehat{g}_q^{(N_q)}. \end{aligned} \tag{21}$$

Heterogeneous counts therefore assign different truncation orders of the same log-success expansion to different prompts. This characterizes the expected on-policy estimator at the current allocation.

Relative scaling and direction. Using the original batch normalizer, define the retained-set maximum-likelihood reference and the conditional mean estimator:

$$G_{\mathcal{A}}^{\log} := \frac{1}{B} \sum_{q \in \mathcal{A}} g_q, \qquad \overline{G} := \mathbb{E}[\widehat{G} \mid \mathcal{F}]. \tag{22}$$

For any $\gamma > 0$,

$$\overline{G} = \gamma G_{\mathcal{A}}^{\log} + \frac{1}{B} \sum_{q \in \mathcal{A}} (\kappa_q - \gamma) g_q. \tag{23}$$

If every retained prompt has $\kappa_q = \gamma$, then $\overline{G} = \gamma G_{\mathcal{A}}^{\log}$. Whenever $G_{\mathcal{A}}^{\log} \neq 0$, the two vectors have the same direction. Unequal coefficients can alter the aggregate direction: for two linearly independent prompt gradients, different coefficients change their relative contributions to the sum.

Approximate equalization. Let $\delta := \max_{q \in \mathcal{A}} |\kappa_q - \gamma|$. Applying the triangle inequality to Equation 23 gives

$$\left\| \overline{G} - \gamma G_{\mathcal{A}}^{\log} \right\| \leq \frac{\delta}{B} \sum_{q \in \mathcal{A}} \|g_q\|. \tag{24}$$

For fixed prompt gradients and reference scale, reducing the maximum coefficient deviation tightens this bound on the conditional mean-gradient discrepancy. Exact equalization makes the discrepancy zero. These results motivate controlling the relative finite-rollout coefficients across retained prompts. SERA uses estimated success probabilities to target this equalization.

B.3 Derivation of the Fixed-Budget Allocation

We derive the allocation in Equations 5 and 6 from the fixed-budget objective in Equation 4. Consider a batch $\mathcal{B}$ of B prompts with total rollout budget $M = BN_0 \geq B$. We first treat the counts as continuous variables $N_q \geq 1$, before imposing practical count bounds and integer constraints. For $p_q \in (0, 1)$, define $h_q := -\log(1 - p_q) > 0$, so that

$$\kappa(p_q, N_q) = 1 - \exp[-h_q(N_q - 1)]. \tag{25}$$

Rollout counts required by a common fidelity target. Suppose every prompt must attain a fidelity of at least $\gamma \in [0, 1)$. Writing $c := -\log(1 - \gamma) \geq 0$, this requirement is equivalent to

$$\kappa(p_q, N_q) \geq \gamma \quad \Longleftrightarrow \quad h_q(N_q - 1) \geq c \quad \Longleftrightarrow \quad N_q \geq 1 + \frac{c}{h_q}. \tag{26}$$

Thus, a common fidelity target determines a minimum rollout count for each prompt.

Determining the target from the budget. Summing the required counts over the batch gives

$$M = \sum_{q \in \mathcal{B}} N_q \geq B + c \sum_{q \in \mathcal{B}} h_q^{-1}. \tag{27}$$

Consequently, every feasible target must satisfy

$$c \leq c_{\mathcal{B}} := \frac{M - B}{\sum_{q \in \mathcal{B}} h_q^{-1}}. \tag{28}$$

This upper bound is attained by setting $N_q = 1 + c_{\mathcal{B}}/h_q$ for every prompt. These counts sum to M and give every prompt the same fidelity $1 - \exp(-c_{\mathcal{B}})$. Any larger common fidelity would require more than M rollouts, so this allocation maximizes the minimum fidelity under the continuous budget constraint.

Substituting the budget-determined value of $c_{\mathcal{B}}$ yields the explicit allocation

$$N_q^\star = 1 + \frac{c_{\mathcal{B}}}{h_q} = 1 + (M - B) \frac{[-\log(1 - p_q)]^{-1}}{\sum_{r \in \mathcal{B}} [-\log(1 - p_r)]^{-1}}. \tag{29}$$

The factor $M - B$ follows from $\sum_{q \in \mathcal{B}} (N_q - 1) = M - B$: the centered estimator's fidelity depends on $N_q - 1$. Lower success probabilities produce larger inverse exposures h_q^{-1} and therefore receive more rollouts.

Allocation on a retained set. The same derivation applies to a fixed nonempty retained set $\mathcal{A} \subseteq \mathcal{B}$ of size $L = |\mathcal{A}|$. Keeping the original total budget $M = BN_0$, we obtain

$$N_q^\star = 1 + (M - L) \frac{[-\log(1 - p_q)]^{-1}}{\sum_{r \in \mathcal{A}} [-\log(1 - p_r)]^{-1}}, \qquad q \in \mathcal{A}. \tag{30}$$

Here, $\sum_{q \in \mathcal{A}} (N_q^\star - 1) = M - L$. The practical procedure uses frozen success estimates $\widehat{p}_q$, incorporates count bounds, and converts the continuous counts to integers while preserving the total budget, as detailed in Appendix C.2.

Optimality with count bounds. Assume $LN_{\min} \le M \le LN_{\max}$. A common target $\kappa(p_q, N_q) \ge 1 - e^{-t}$, with $t \ge 0$, is feasible if and only if

$$t \le \min_{q \in \mathcal{A}} h_q(N_{\max} - 1), \qquad \sum_{q \in \mathcal{A}} \max\left\{N_{\min}, 1 + \frac{t}{h_q}\right\} \le M.$$

The clipped waterline attains the largest feasible target and therefore solves the bounded continuous max–min problem.

B.4 Mixed-Group Observability and Budget Concentration

Observability as a safeguard. The monotonicity of U implies that $U(p, N_{\max}) < u_0$ rules out meeting the observability threshold with any permitted rollout count. Conversely, passing this maximum-count screen does not ensure that a smaller allocated count meets the threshold. SERA therefore evaluates observability again at the final integer counts, using the frozen success estimates $\widehat{p}_q$.

The quantities κ and U describe different aspects of the estimator. At fixed $N \ge 2$, $\kappa(p, N)$ increases with p, whereas $U(p, N)$ is symmetric around $p = 1/2$. In particular, as $p \to 1$, $\kappa(p, N) \to 1$ while $U(p, N) \to 0$. The former describes the multiplicative coefficient in the expected gradient; the latter describes the probability of observing nonzero centered advantages. Accordingly, SERA uses κ as the allocation criterion and U as an observability safeguard.

Low-success budget concentration. For a fixed positive target exposure c, the unclipped allocation in Equation 5 satisfies

$$N_q - 1 = \frac{c}{-\log(1 - p_q)} \sim \frac{c}{p_q} \qquad (p_q \to 0). \tag{31}$$

Maintaining a given fidelity therefore requires increasingly many rollouts as the success probability approaches zero. Under a fixed total budget, the common water level adjusts to the full set of probabilities, so very low-success prompts can absorb a large budget share and lower the common fidelity attainable across the batch. This motivates combining per-prompt count bounds with observability-guided selection when distributing the budget.

B.5 Comparison with Inverse-Fidelity Reweighting

For oracle p and a fixed integer $N \ge 2$, $\kappa(p, N) > 0$, so the corrected estimator $\widetilde{g} := \widehat{g}^{(N)}/\kappa(p, N)$ satisfies $\mathbb{E}[\widetilde{g}] = g$. Whenever the covariance exists,

$$\mathrm{Cov}(\widetilde{g}) = \frac{1}{\kappa(p, N)^2} \mathrm{Cov}(\widehat{g}^{(N)}). \tag{32}$$

This correction rescales the sampled estimator. It leaves the probability of observing a mixed group unchanged, and all-correct or all-incorrect groups still produce zero centered advantages. For an individual prompt, increasing its rollout count instead increases both $\kappa(p, N)$ and $U(p, N)$. Rollout allocation therefore acts on the sampling process, affecting both finite-rollout scaling and mixed-group observability.

C Implementation Details

C.1 Historical Success-Probability Estimates

We maintain historical success statistics using persistent prompt identifiers. For each prompt q, let $S_q^{(e)}$ and $F_q^{(e)}$ denote the discounted success and failure counts available at the start of epoch e. Both are initialized to zero. Let $n_q^{(e)}$ and $k_q^{(e)}$ be the numbers of responses and successes collected during

epoch e. At the next epoch boundary, we update

$$\begin{aligned} S_q^{(e+1)} &= \lambda S_q^{(e)} + k_q^{(e)}, \\ F_q^{(e+1)} &= \lambda F_q^{(e)} + n_q^{(e)} - k_q^{(e)}, \\ \widehat{p}_q^{(e+1)} &= \frac{\alpha_0 + S_q^{(e+1)}}{\alpha_0 + \beta_0 + S_q^{(e+1)} + F_q^{(e+1)}}. \end{aligned} \tag{33}$$

Here $\alpha_0, \beta_0 > 0$ are fixed prior parameters and $\lambda \in [0, 1]$ controls the discount applied to historical evidence. The prior parameters themselves are not discounted.

The estimates $\widehat{p}_q^{(e)}$ remain fixed throughout epoch e. New outcomes are accumulated in pending counts and incorporated at the next epoch boundary. Thus, allocation decisions for the current responses depend only on previously incorporated observations. For a prompt receiving no responses during an epoch, $n_q^{(e)} = k_q^{(e)} = 0$, and only the historical evidence is discounted.

Availability and uniform initialization. A prompt has an available estimate once observations for its identifier have been incorporated at an epoch boundary. The prior alone does not mark an unseen prompt as available. If any prompt in a candidate batch lacks an available estimate, we retain the full batch and assign $N_q = N_0$ to every prompt. These uniform batches also contribute observations to the historical estimator. Previously observed prompts remain available even when they receive no responses in a later epoch.

This procedure supplies smoothed historical estimates for allocation without requiring additional preliminary rollouts from the current batch. Task-specific prior and discount parameters are reported in Appendix D.

C.2 Bounded Allocation and Exact-Budget Integerization

For a candidate batch $\mathcal{B}$ of size B, the total rollout budget is $M = BN_0$. Given a retained set $\mathcal{A}$, we allocate this full budget using the frozen success estimates $\widehat{p}_q$. The integer count bounds satisfy $2 \le N_{\min} \le N_0 \le N_{\max}$. A retained set is feasible when

$$|\mathcal{A}|N_{\min} \le M \le |\mathcal{A}|N_{\max}. \tag{34}$$

Bounded continuous counts. Applying the allocation derived in Appendix B.3, define $h_q := -\log(1 - \widehat{p}_q)$ and solve

$$x_q := \mathrm{clip}\left(1 + \frac{c_{\mathcal{A}}}{h_q}, N_{\min}, N_{\max}\right), \qquad \sum_{q \in \mathcal{A}} x_q = M. \tag{35}$$

The sum of the clipped counts is continuous and nondecreasing in $c_{\mathcal{A}}$. For an interior budget, the implementation uses 100 bisection iterations in log water level. It returns the lower-budget side of the final bracket before integerization. At either endpoint of Equation 34, the budget forces every retained count to the corresponding bound.

The water level is recomputed whenever the retained set changes. It is also solved with the bounds already included, so the bounded allocation preserves the original budget.

Completing the integer budget. Let $\gamma_\star := 1 - \exp(-c_{\mathcal{A}})$ denote the interior fidelity target. We first floor the continuous counts:

$$n_q := \lfloor x_q \rfloor, \qquad r := M - \sum_{q \in \mathcal{A}} n_q. \tag{36}$$

For every prompt with $n_q < N_{\max}$, compute the change in squared fidelity error caused by one additional rollout:

$$\Delta_q := [\kappa(\widehat{p}_q, n_q + 1) - \gamma_\star]^2 - [\kappa(\widehat{p}_q, n_q) - \gamma_\star]^2. \tag{37}$$

Let $\mathcal{R}$ contain the r below-cap prompts with the smallest Δ_q, breaking ties by their original candidate-batch order. The final counts are

$$N_q = n_q + \mathbf{1}_{\{q \in \mathcal{R}\}}, \qquad q \in \mathcal{A}. \tag{38}$$

The marginal errors are computed once, and each selected prompt receives one increment. Every below-cap prompt is eligible, including a prompt whose continuous count was already an integer.

This completion satisfies

$$\sum_{q \in \mathcal{A}} N_q = M, \qquad N_{\min} \le N_q \le N_{\max}. \tag{39}$$

For any feasible one-increment completion $z_q \in \{0, 1\}$ with $\sum_q z_q = r$, the change in total squared fidelity error is $\sum_q z_q \Delta_q$. Selecting the smallest marginals therefore minimizes this error within the class of one-increment completions. We denote the complete bounded allocation and integerization procedure by ALLOCATE.

C.3 ACTIVE-PREFIX SEARCH AND CAPACITY FALLBACK

For each candidate prompt, compute its estimated mixed-group probability at the maximum permitted count:

$$s_q := \widehat{U}_q(N_{\max}), \qquad \widehat{U}_q(N) := 1 - (1 - \widehat{p}_q)^N - \widehat{p}_q^N. \tag{40}$$

We stably rank the prompts by decreasing s_q and let $\mathcal{A}_K$ denote the first K prompts. The minimum retained-set size required to absorb the budget and the number passing the maximum-count screen are

$$K_{\min} := \left\lceil \frac{M}{N_{\max}} \right\rceil, \qquad K_{\text{elig}} := \left|\{q \in \mathcal{B} : s_q \ge u_0\}\right|. \tag{41}$$

Since $M = BN_0$ and $N_{\min} \le N_0 \le N_{\max}$, every prefix size between $K_{\min}$ and B satisfies the count-capacity conditions.

For each tested prefix, we recompute its allocation using the original budget M and evaluate

$$\min_{q \in \mathcal{A}_K} \widehat{U}_q(N_q^{(K)}) \ge u_0 - 10^{-12}. \tag{42}$$

This check uses the final integer counts returned by ALLOCATE. The tolerance accommodates floating-point comparisons near the threshold.

The implementation uses binary search over prefix sizes in $[K_{\min}, K_{\text{elig}}]$ and returns the largest passing prefix it encounters. Because integerization can break exact monotonicity, this need not be the largest passing prefix over all possible sizes. If the maximum-count screen leaves fewer than $K_{\min}$ prompts, or no tested prefix passes the final-count check, we allocate the budget to the top $K_{\min}$ prompts. This capacity fallback preserves the budget and count bounds; the returned prompts may fall below the observability threshold. Inactive prompts receive zero rollouts.

C.4 LOSS AGGREGATION AND MULTIPLICITY CORRECTION

Correction for heterogeneous rollout counts. Each response from a retained prompt receives the weight $w_q = N_0/N_q$. Using the unstabilized centered advantage a_{qi} from Equation 2, response-mean score aggregation gives

$$\frac{1}{M} \sum_{q \in \mathcal{A}} \sum_{i=1}^{N_q} w_q a_{qi} s_{qi} = \frac{1}{B} \sum_{q \in \mathcal{A}} \widehat{g}_q^{(N_q)}. \tag{43}$$

The equality follows from $M = BN_0$ and $N_q w_q = N_0$. Thus, heterogeneous rollout counts preserve the same total response weight for each retained prompt. Weights are evaluated only for prompts with $N_q > 0$.

In the sequence-task implementation, the response weight multiplies the centered advantage before the policy loss is computed. Reward means are computed within the original generation groups before the responses are partitioned into PPO minibatches and microbatches. The resulting advantages and response weights remain attached to their response records throughout subsequent batching.

Algorithm 2 Observability-guided prefix search

Require: Candidate batch $\mathcal{B}$ with available estimates $\widehat{\mathbf{p}}$
Require: Budget $M = BN_0$, bounds $N_{\min}, N_{\max}$, threshold u_0
Stably rank $\mathcal{B}$ by decreasing $\widehat{U}_q(N_{\max})$
Compute $K_{\min}$ and K_{elig} using Equation 41
$l \leftarrow K_{\min}$; $h \leftarrow K_{\text{elig}}$; $K_{\text{best}} \leftarrow 0$
while $l \leq h$ **do**
    $K \leftarrow \lfloor (l+h)/2 \rfloor$
    $\mathcal{A}_K \leftarrow$ the top K prompts
    $\mathbf{N}^{(K)} \leftarrow \textsc{Allocate}(\mathcal{A}_K, \widehat{\mathbf{p}}, M, N_{\min}, N_{\max})$
    if $\min_{q \in \mathcal{A}_K} \widehat{U}_q(N_q^{(K)}) \geq u_0 - 10^{-12}$ **then**
        $K_{\text{best}} \leftarrow K$; $\mathbf{N}^{\text{best}} \leftarrow \mathbf{N}^{(K)}$
        $l \leftarrow K + 1$
    else
        $h \leftarrow K - 1$
    end if
end while
if $K_{\text{best}} > 0$ **then**
    $\mathcal{A} \leftarrow \mathcal{A}_{K_{\text{best}}}$; $\mathbf{N} \leftarrow \mathbf{N}^{\text{best}}$
else
    $\mathcal{A} \leftarrow \mathcal{A}_{K_{\min}}$
    $\mathbf{N} \leftarrow \textsc{Allocate}(\mathcal{A}, \widehat{\mathbf{p}}, M, N_{\min}, N_{\max})$
end if
Set $N_q = 0$ for $q \in \mathcal{B} \setminus \mathcal{A}$
return $\mathcal{A}, \mathbf{N}$

Sequence and token normalization. For an aggregation unit containing responses $i \in \mathcal{I}$, let q_i be the associated prompt, m_{it} the valid-response-token mask, and ℓ_{it} the token policy-loss term before response weighting. We use the following reductions:

$$L_{\text{seq}} = \frac{\sum_{i \in \mathcal{I}} \sum_t w_{q_i} m_{it} \ell_{it}}{|\mathcal{I}| L_{\text{cap}}}, \tag{44}$$

$$L_{\text{token}} = \frac{\sum_{i \in \mathcal{I}} \sum_t w_{q_i} m_{it} \ell_{it}}{\sum_{i \in \mathcal{I}} \sum_t m_{it}}. \tag{45}$$

Here L_{cap} is a fixed length normalizer shared by all responses. Both reductions retain the response weights in the numerator. The token-mean denominator is the number of valid response tokens in the aggregation unit.

D Extended Experimental Details

D.1 Experimental Setup

For all tasks, B denotes the candidate-prompt batch size and N_0 denotes the average number of generations per prompt, giving a fixed rollout budget $M = BN_0$ per full batch. Fixed MaxRL assigns $N_q = N_0$ to every prompt. For SERA, we use an observability threshold $u_0 = 0.05$ and active-prompt bounds $N_{\min} = 2$ and $N_{\max} = 4N_0$. Inactive prompts receive no rollouts. Each response from an active prompt is weighted by N_0/N_q.

Maze, SmolLM2, and Qwen use historical success-probability estimates with a Beta$(0.5, 0.5)$ prior and an evidence discount of 0.75 per epoch. Estimates are updated at epoch boundaries and remain frozen within each epoch. Batches containing prompts without available historical estimates use uniform allocation, $N_q = N_0$. ImageNet instead uses detached exact correct-class probabilities at each training step.

We follow the task-specific setup of MaxRL (Tajwar et al., 2026), with the main training and validation settings summarized in Figures 5–13. Generations per prompt denotes the average budget

N_0; the realized count N_q can vary across prompts under SERA. Sequence lengths are measured in tokens. Grad update per step denotes the number of optimizer updates per RL training step. Training Steps denotes the checkpoint used for comparison. For sequence tasks, MaxRL and SERA use the same token-mean or Seqnorm loss aggregation within each controlled comparison.

D.1.1 IMAGENET

ImageNet-1K is an image-classification task in which a model assigns an input image to one of 1,000 categories. We formulate classification as a one-step decision problem: each image is a prompt, the policy samples a class label, and the reward is one if the sampled label matches the ground-truth class and zero otherwise. This setting provides an exact success probability, $p_q = \pi_\theta(y_q^\star \mid q)$, allowing us to study rollout allocation without success-probability estimation error.

We train a randomly initialized ResNet-50 using 1,281,167 training images and 50,000 validation images. Training inputs use random resized cropping to 224×224 and horizontal flipping; validation uses a 224×224 center crop. We compare average rollout budgets $N_0 \in \{16, 128\}$ and report the step-100,000 checkpoint. Pass@K is computed exactly as the validation-set mean of $1 - (1 - p_q)^K$. The main configuration is shown in Figure 5.

Training configuration for ImageNet

Parameter	Value	Parameter	Value
Base model	ResNet-50	Prompts per batch	256
Generations per prompt	16 / 128	Grad update per step	1
Learning rate	0.1	Training Steps	100,000
KL coeff	0	Entropy coeff	0
Rollout temp	1.0	Device	4 × NVIDIA A100

Figure 5: Training and validation settings for ImageNet.

D.1.2 MAZE

Maze is a sequential navigation task in which the policy must generate a valid path from a designated starting position to a goal while respecting the maze constraints. A completed navigation sequence receives a binary reward indicating whether it successfully solves the maze. This task extends the one-step ImageNet setting to multi-step generation with a verifiable outcome.

We use a fixed set of 7,424 training mazes and evaluate on 400 held-out mazes, all with grid size 17×17. The initial policy is the step-1500 supervised fine-tuning checkpoint of a Qwen2 decoder with approximately 3.94M parameters.

Both loss-normalization comparisons use the step-3000 checkpoint. Evaluation samples 2,048 responses per maze. The reported curves use empirical Pass@1 and with-replacement bootstrap for $K \geq 2$. The main configuration is shown in Figure 6.

Training configuration for fixed-data maze navigation

Parameter	Value	Parameter	Value
Base model	Qwen2 (3.94M)	Prompts per batch	256
Generations per prompt	128	Grad update per step	1
Max prompt length	320	Max response len	180
Learning rate	1×10^{-4}	Training Steps	3000
KL coeff	0	Entropy coeff	0
Rollout temp	1.0	Validation top-p	1.0
Validation temp	1.0	Device	4 × NVIDIA A100

Figure 6: Training and validation settings for Maze.

We represent each maze as a row-major sequence of symbolic cell tokens and generate navigation actions after `PATH_START`. Figure 7 shows an example input and its corresponding valid output, with part of the input omitted for readability.

Maze example input and output

Input:

```
<bos> GRID_START WALL WALL WALL ... WALL NEWLINE
WALL START PATH PATH PATH ... WALL NEWLINE
...
WALL PATH WALL PATH ... PATH GOAL WALL NEWLINE
WALL WALL WALL ... WALL NEWLINE GRID_END PATH_START
```

Output:

```
RIGHT RIGHT DOWN DOWN RIGHT RIGHT DOWN DOWN DOWN DOWN DOWN DOWN DOWN DOWN
DOWN DOWN RIGHT RIGHT RIGHT RIGHT RIGHT RIGHT DOWN DOWN RIGHT RIGHT RIGHT
RIGHT DONE <eos>
```

Figure 7: Example input and output for a 17×17 Maze.

D.1.3 SmolLM2 on GSM8K

GSM8K is a mathematical reasoning task consisting of grade-school word problems that require multi-step arithmetic. Given a problem, the model generates a solution and a final numerical answer. The final answer is checked against the reference answer to obtain a binary correctness reward. We use this setting to evaluate rollout allocation for natural-language reasoning with a compact language model.

We initialize from SmolLM2-360M-Instruct and train on the GSM8K training split. Final evaluation is configured on the 1,209-question GSM8K-Platinum set with 256 responses per question, using the step-2000 checkpoint.

Pass@K is computed using the combinatorial estimator. All Pass@K values for a checkpoint use the same response pool, which is separate from the online validation pool. The main configuration is shown in Figure 8.

Training configuration for SmolLM2-360M-Instruct

Parameter	Value	Parameter	Value
Base model	SmolLM2-360M-Instruct	Prompts per batch	256
Generations per prompt	16	Grad update per step	1
Max prompt length	512	Max response len	2,048
Learning rate	1×10^{-5}	Training Steps	2,000
KL coeff	0	Entropy coeff	0
Rollout temp	1.0	Validation top-p	0.95
Validation temp	0.6	Device	8 × NVIDIA A100

Figure 8: Training and validation settings for SmolLM2-360M-Instruct.

For both GSM8K training and GSM8K-Platinum evaluation, we append the same reasoning and answer-format instruction to each question and apply the SmolLM2 tokenizer's chat template. Figure 9 shows the prompt.

User prompt template for SmolLM2-360M-Instruct

```
{question} Let's think step by step and output the final answer within
\boxed{}.
```

Figure 9: Prompt template for SmolLM2-360M-Instruct.

D.1.4 QWEN

The Qwen experiments evaluate mathematical reasoning with Qwen2.5-Math-1.5B (Yang et al., 2024b) and Qwen3-4B-Base (Yang et al., 2025a). Qwen2.5-Math-1.5B is trained on MATH levels 3–5 (Hendrycks et al., 2021), while Qwen3-4B-Base is trained on POLARIS-53K (An et al., 2025). Both comparisons use token-mean loss aggregation. For Qwen3-4B-Base, we follow the publicly released MaxRL training configuration (Tajwar et al., 2026), retaining its data preprocessing, prompt template, optimizer settings, generation limits, and total training rollout budget. Training and online validation settings are summarized in Figures 10 and 11.

Final evaluation uses checkpoints at step 600 for Qwen2.5-Math-1.5B and step $1{,}000$ for Qwen3-4B-Base. For Qwen3-4B-Base, the GRPO and MaxRL baselines use the publicly released checkpoints. Qwen2.5-Math-1.5B is evaluated on BeyondAIME, AIME 2025, MATH-500, and the English, text-only mathematics subset of OlympiadBench. Qwen3-4B-Base is evaluated on BeyondAIME, AIME 2025, MATH-500, and Minerva Math. For both models, we sample $n = 512$ responses per question with temperature 0.6 and top-p 0.95. The maximum response length is $3{,}000$ tokens for Qwen2.5-Math-1.5B and $4{,}096$ tokens for Qwen3-4B-Base.

We compute Pass@K using the unbiased combination estimator,

$$\widehat{\text{Pass@}K} = \frac{1}{Q} \sum_{q=1}^{Q} \left[1 - \frac{\binom{n-c_q}{K}}{\binom{n}{K}} \right],$$

where Q is the number of evaluation questions and c_q counts responses that pass the answer verifier and have a generated length strictly below the model-specific response limit.

Training configuration for Qwen2.5-Math-1.5B

Parameter	Value	Parameter	Value
Base model	Qwen2.5-Math-1.5B	Prompts per batch	256
Generations per prompt	16	Grad update per step	1
Max prompt length	1,024	Max response len	3,000
Learning rate	1×10^{-6}	Training Steps	600
KL coeff	0	Entropy coeff	0
Rollout temp	1.0	Validation top-p	0.95
Validation temp	0.6	Device	8 × NVIDIA A100

Figure 10: Training and validation settings for Qwen2.5-Math-1.5B.

Training configuration for Qwen3-4B-Base

Parameter	Value	Parameter	Value
Base model	Qwen3-4B-Base	Prompts per batch	256
Generations per prompt	16	Grad update per step	1
Max prompt length	1,024	Max response len	4,096
Learning rate	1×10^{-6}	Training Steps	1,000
KL coeff	0	Entropy coeff	0
Rollout temp	1.0	Validation top-p	0.95
Validation temp	0.6	Device	4 nodes × 8 H20

Figure 11: Training and validation settings for Qwen3-4B-Base.

Figures 12 and 13 show the resulting prompts for Qwen2.5-Math-1.5B and Qwen3-4B-Base, respectively.

User prompt template for Qwen2.5-Math-1.5B

```
{question}
Please reason step by step, and put your final answer within \boxed{{}}.
```

Figure 12: Prompt template for Qwen2.5-Math-1.5B.

User prompt template for Qwen3-4B-Base

```
{question}
Please reason step by step, and put your final answer within \boxed{{}}.
```

Figure 13: Prompt template for Qwen3-4B-Base.

D.2 More Experimental Results

D.2.1 ImageNet

Final performance. Table 2 reports the final Pass@K profiles at step 100,000, corresponding to Figure 2 (c). At $N_0 = 128$, SERA outperforms MaxRL at every reported K, increasing Pass@1 from 36.59% to 54.80% and Pass@128 from 53.22% to 87.51%. These improvements substantially narrow the gap to direct cross-entropy (CE) training across the evaluated sampling budgets. At $N_0 = 16$, SERA also improves Pass@1 and Pass@128, with gains extending through $K = 512$.

Table 2: **Final ImageNet Pass@K (%).** Results at step 100,000, corresponding to Figure 2 (c). Bold indicates the higher value within each matched-budget MaxRL–SERA pair. Direct CE provides the exact-likelihood training reference.

N_0	Method	1	2	4	8	16	32	64	128	256	512	1024
–	Direct CE	60.81	70.26	77.38	82.78	86.95	90.18	92.68	94.59	96.05	97.16	98.00
16	MaxRL	4.43	4.97	5.37	5.71	6.05	6.47	7.13	8.24	10.21	13.67	**19.35**
16	SERA	**12.37**	**13.61**	**14.54**	**15.26**	**15.82**	**16.25**	**16.59**	**16.86**	**17.09**	**17.29**	17.49
128	MaxRL	36.59	41.30	44.79	47.43	49.45	51.01	52.24	53.22	54.00	54.62	55.14
128	SERA	**54.80**	**63.53**	**70.26**	**75.46**	**79.52**	**82.75**	**85.37**	**87.51**	**89.25**	**90.65**	**91.79**

Exact-gradient reference and measurement. ImageNet provides an exact reference for measuring how closely sampled updates align with likelihood optimization. Let $\mathcal{B}$ denote the full candidate batch, with $B = |\mathcal{B}|$, and let $p_q(\theta) = \pi_\theta(y_q^\star \mid q)$ be the probability assigned to the ground-truth class $y_q^\star$. We define

$$L_{\mathrm{CE}}(\theta; \mathcal{B}) = -\frac{1}{B} \sum_{q \in \mathcal{B}} \log p_q(\theta), \qquad H_{\mathrm{CE}} = \nabla_\theta L_{\mathrm{CE}}(\theta; \mathcal{B}). \tag{46}$$

This reference covers the entire candidate batch, including prompts assigned zero rollouts. For the realized sampled training loss L_{sample}, incorporating the run's allocation and loss normalization, let $H_{\mathrm{sample}} = \nabla_\theta L_{\mathrm{sample}}$. We measure gradient alignment as

$$\mathrm{Cosine} = \frac{\langle H_{\mathrm{sample}}, H_{\mathrm{CE}} \rangle}{\|H_{\mathrm{sample}}\|_2 \|H_{\mathrm{CE}}\|_2}. \tag{47}$$

Both vectors use the loss-gradient convention and cover all trainable model parameters. Within each run, they are computed from the same forward pass at the same current parameters and inputs, after backpropagation and before gradient clipping or optimizer operations, including weight decay. The diagnostic therefore measures sampled-to-exact alignment along each method's own training trajectory. The separately CE-trained model supplies the performance reference in Figure 2 (a–c), while the CE reference gradient in Figure 2 (d) is computed at the corresponding RL model.

At $N_0 = 128$, diagnostics are recorded every 1,000 steps from step 1,000 through step 100,000. Across these 100 measurements, SERA achieves a mean cosine of 0.6266, compared with 0.4317 for MaxRL. At the final checkpoint, the corresponding values are 0.7960 and 0.4447. These results directly demonstrate improved alignment with the full-batch likelihood gradient along the observed training trajectories. Together with the validation results, they show that SERA improves both gradient alignment and held-out performance under the same rollout budget.

Accuracy of success-probability estimates. We use historical estimates to preserve flexibility in allocating the current batch's rollout budget. Under a fixed budget $M = BN_0$, a pilot stage assigns m samples to each prompt before adaptive allocation, committing Bm rollouts and leaving $B(N_0 - m)$ for adaptive continuation. For $N_0 = 16$ and $m = 8$, half of the budget is therefore assigned uniformly in advance. These pilot samples remain within M and are reused for training on active prompts. Historical estimation reuses past rollout outcomes without requiring a current-batch pilot stage.

We assess estimation quality on ImageNet, where $p_q = \pi_\theta(y_q^\star \mid q)$ is the exact probability of the correct class. The historical estimator uses the discounted Beta update in Appendix C.1, with $\alpha_0 = \beta_0 = 0.5$ and $\lambda = 0.75$, and remains frozen within each epoch using only outcomes from earlier epochs. For each batch, we compute the mean absolute error (MAE) and Pearson's correlation coefficient r between the estimated and exact success probabilities across all candidate prompts. We report the arithmetic mean of each metric over all batches in the final epoch. The historical estimator's MAE of 0.0386 and mean Pearson correlation of 0.8134 indicate that it captures variation in prompt-level success probabilities without current-batch pilot sampling (Table 3).

Table 3: **Success-probability estimation on ImageNet.** Batch-level metrics averaged over the $5{,}005$ batches in epoch 20 ($N_0 = 16$, nominal $B = 256$, seed 69). m denotes current-batch pilot samples per prompt; pilot estimates use $\mathrm{clip}(K/8, 1/16, 15/16)$, where K counts pilot successes. The runs have different allocation configurations and training trajectories.

Estimator	m	MAE ↓	Pearson r ↑
Pilot estimate	8	0.0634	0.9879
Historical estimate	0	0.0386	0.8134

D.2.2 MAZE

Final performance. Table 4 reports the final Pass@K profiles at step $3{,}000$, complementing Figure 3. Evaluation uses 400 held-out mazes with $2{,}048$ responses per maze. We report both token-mean normalization and fixed-length sequence normalization (Seqnorm), together with an additional GRPO baseline under token-mean normalization. The base-model reference uses the step-0 evaluation of the shared SFT checkpoint.

Under token-mean normalization, SERA outperforms MaxRL at every reported $K \geq 4$, with the advantage increasing at larger evaluation budgets. Under Seqnorm, SERA improves the entire reported Pass@K profile. The Pass@128 gains over MaxRL are 4.84 and 3.28 percentage points under token-mean and Seqnorm, respectively. These results demonstrate stronger solution coverage under both normalizations, with Seqnorm also improving single-sample performance.

D.2.3 SMOLLM2 ON GSM8K

We extend the comparison in Table 1 with validation trajectories, complete Pass@K profiles, and training diagnostics for SmolLM2-360M-Instruct. MaxRL and SERA use matched training response budgets within each loss normalization.

Figure 14 (a) tracks validation Pass@32 for MaxRL, SERA, GRPO, and DARS-HW throughout training. MaxRL, SERA, and DARS-HW use fixed-length sequence normalization (Seqnorm), while GRPO uses token-mean normalization. Coverage improves early in training for all four methods. During later training, MaxRL's coverage declines, while SERA maintains a higher Pass@32 and finishes with stronger validation coverage.

Validation is performed every 100 steps on $1{,}209$ GSM8K-Platinum questions, with 32 responses per question. We plot the logged bootstrap Best@32 metric, reported as Pass@32 by the trainer. For each question, this statistic averages the maximum binary reward over bootstrap samples of 32 responses drawn with replacement from its validation pool, then averages across questions.

Figure 14 (b) compares the final Pass@K profiles of MaxRL, SERA, GRPO, and DARS-HW. We evaluate the step-$2{,}000$ checkpoints using a separate pool of 256 responses per question on the same

Table 4: **Final Maze Pass@K (%).** RL results are evaluated at step 3,000, using 2,048 responses per held-out maze. The gray row reports the pre-RL base-model reference at step 0. Bold indicates the higher value within each MaxRL–SERA pair using the same normalization.

Normalization	Method	1	2	4	8	16	32
–	Base model	1.27	2.49	4.75	8.83	15.50	25.17
Token-mean	GRPO	34.00	34.80	35.43	36.01	36.61	37.28
Token-mean	MaxRL	**52.95**	**59.16**	63.16	66.16	68.62	70.75
Token-mean	SERA	49.98	58.19	**63.85**	**68.06**	**71.49**	**74.45**
Seqnorm	MaxRL	52.80	59.42	63.74	66.96	69.63	71.94
Seqnorm	SERA	**55.61**	**61.66**	**65.93**	**69.45**	**72.43**	**74.97**

Normalization	Method	64	128	256	512	1024	2048
–	Base model	37.18	49.67	60.51	68.79	74.59	78.28
Token-mean	GRPO	38.01	38.74	39.47	40.24	41.03	41.77
Token-mean	MaxRL	72.60	74.31	75.82	76.98	77.79	78.38
Token-mean	SERA	**76.98**	**79.15**	**81.06**	**82.58**	**83.75**	**84.48**
Seqnorm	MaxRL	73.83	75.51	76.93	78.04	78.86	79.62
Seqnorm	SERA	**77.06**	**78.79**	**80.19**	**81.40**	**82.45**	**83.22**

evaluation set. All values at $K \in \{1, 2, 4, 8, 16, 32, 64, 128, 256\}$ are computed from this pool using

$$\widehat{\text{Pass@}K} = \frac{1}{Q} \sum_{q=1}^{Q} \left(1 - \frac{\binom{256-c_q}{K}}{\binom{256}{K}} \right), \tag{48}$$

where $Q = 1{,}209$, c_q is the number of correct responses for question q, and $\binom{a}{K} = 0$ for $K > a$.

SERA achieves the highest Pass@K among the four methods at every displayed $K \geq 4$. Compared with MaxRL, it improves coverage at every displayed $K \geq 2$ while retaining nearly identical Pass@1 under the same training response budget. It also outperforms GRPO across the entire displayed range of evaluation budgets.

Table 5 quantifies uncertainty in the final Seqnorm comparison. We use 10,000 paired bootstrap resamples of evaluation questions, keeping the checkpoints and response pools fixed, and report pointwise 95% percentile confidence intervals. The intervals for SERA's gains over MaxRL are above zero at all three reported budgets.

Table 5: **SmolLM2 uncertainty analysis.** Δ denotes SERA minus MaxRL at step 2,000, in percentage points. CI denotes the question-paired 95% confidence interval.

	Pass@32	Pass@128	Pass@256
Δ	+4.69	+5.54	+5.13
95% CI	[2.32, 7.04]	[3.09, 8.01]	[2.48, 7.86]

D.2.4 Mathematical reasoning with Qwen

Table 6 reports the numerical Pass@K profiles of MaxRL and SERA from Figure 4, together with the pretrained base model as a reference. All results use the evaluation protocol described in Appendix D.1.4.

Using the same question-paired bootstrap procedure, Table 7 reports uncertainty in the final Qwen2.5-Math-1.5B comparisons. The Pass@32 intervals are above zero on AIME 2025, MATH-500, and OlympiadBench, while intervals become wider at larger evaluation budgets.

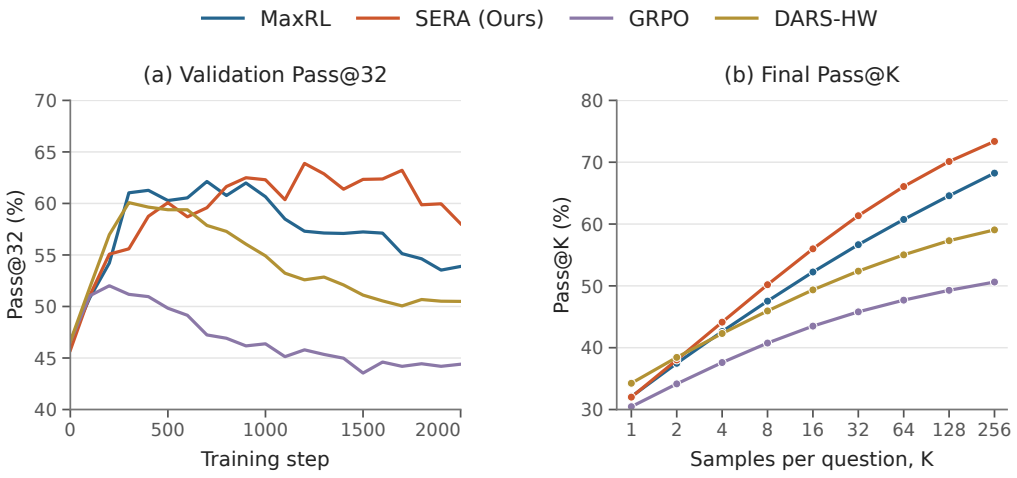

Figure 14: **SmolLM2 results on GSM8K-Platinum.** (a) Validation Pass@32 (bootstrap Best@32) during training. (b) Final Pass@K at step 2,000 from 256 responses per question. MaxRL, SERA (Ours), and DARS-HW use Seqnorm; GRPO uses token-mean normalization. DARS-HW uses approximately $1.49\times$ as many training responses as the other methods.

Table 6: **Qwen2.5-Math-1.5B Pass@K (%).** MaxRL and SERA are evaluated at step 600 using 512 sampled responses per question. Gray rows show the pretrained base model under the same protocol. Bold indicates the higher unrounded value between MaxRL and SERA for each benchmark and K.

Dataset	Method	1	2	4	8	16	32	64	128	256	512
BeyondAIME	Base	0.73	1.41	2.61	4.57	7.36	10.96	15.49	21.20	28.13	35.00
	MaxRL	**3.29**	**5.25**	**7.71**	10.82	**14.82**	19.53	24.78	30.53	36.53	42.00
	SERA	3.07	5.06	7.64	**10.82**	14.76	**19.55**	**25.36**	**32.26**	**40.14**	**49.00**
AIME 2025	Base	3.26	6.01	10.36	16.17	22.59	29.21	36.42	44.21	53.12	63.33
	MaxRL	10.05	15.64	21.65	27.37	33.02	38.86	44.49	49.91	56.26	63.33
	SERA	**10.18**	**16.28**	**23.04**	**29.19**	**34.77**	**40.46**	**46.30**	**52.85**	**61.26**	**70.00**
MATH-500	Base	26.70	40.03	54.22	67.28	77.93	85.68	90.80	94.10	96.25	97.60
	MaxRL	**73.91**	80.55	85.07	88.12	90.43	92.36	93.94	95.34	96.66	97.80
	SERA	73.89	**80.68**	**85.34**	**88.55**	**91.04**	**93.16**	**94.85**	**96.20**	**97.29**	**98.00**
OlympiadBench	Base	20.35	29.70	39.17	47.75	55.28	61.64	66.68	70.83	74.77	78.67
	MaxRL	36.05	43.32	49.84	55.74	61.05	65.67	69.65	73.27	76.68	79.85
	SERA	**36.20**	**43.54**	**50.19**	**56.37**	**62.09**	**67.12**	**71.35**	**74.91**	**78.01**	**80.89**

Table 7: **Qwen2.5-Math-1.5B uncertainty.** Each cell reports the difference between SERA and MaxRL at step 600 (top), with its pointwise 95% paired-bootstrap confidence interval (bottom), in percentage points. Differences are computed from unrounded scores before rounding.

Dataset	Pass@32	Pass@128	Pass@256	Pass@512
BeyondAIME	+0.01 [−1.59, 1.56]	+1.72 [−1.50, 4.98]	+3.60 [−1.21, 8.50]	+7.00 [0.00, 15.00]
AIME 2025	+1.61 [0.12, 3.11]	+2.94 [−0.82, 7.27]	+5.00 [−0.21, 11.67]	+6.67 [0.00, 16.67]
MATH-500	+0.80 [0.23, 1.43]	+0.86 [0.02, 1.77]	+0.63 [−0.26, 1.58]	+0.20 [−0.80, 1.20]
OlympiadBench	+1.45 [0.86, 2.07]	+1.64 [0.81, 2.51]	+1.33 [0.19, 2.50]	+1.04 [−0.59, 2.81]

Table 8 reports the full Pass@K results for the checkpoints at the step $1{,}000$. Each question uses 512 sampled responses. At $K = 512$, SERA improves over MaxRL by 8.00 percentage points on BeyondAIME and 4.78 points on Minerva Math.

Table 8: **Qwen3-4B-Base Pass@K (%).** Gray rows show Base; bold marks the higher unrounded score between MaxRL and SERA.

Dataset	Method	1	2	4	8	16	32	64	128	256	512
BeyondAIME	Base	4.02	6.16	8.51	11.08	14.33	18.61	23.90	30.35	37.75	45.00
	MaxRL	6.97	10.53	14.46	18.62	23.28	28.50	34.08	40.27	47.50	55.00
	SERA	**7.78**	**11.34**	**15.78**	**21.08**	**27.17**	**34.01**	**41.39**	**48.81**	**55.91**	**63.00**
AIME 2025	Base	6.82	11.78	18.33	25.23	31.48	37.23	43.08	49.90	57.41	63.33
	MaxRL	16.37	22.75	28.50	33.84	39.43	45.34	51.72	59.17	66.77	**73.33**
	SERA	**18.37**	**23.85**	**29.30**	**35.38**	**41.88**	**48.40**	**54.97**	**61.20**	**67.61**	**73.33**
MATH-500	Base	67.40	78.81	85.37	89.43	92.21	94.21	95.66	96.68	97.41	98.00
	MaxRL	81.52	**87.83**	91.20	93.13	94.48	95.61	96.61	97.44	**98.00**	**98.40**
	SERA	**81.71**	87.60	**91.32**	**93.72**	**95.29**	**96.33**	**97.00**	**97.45**	97.79	98.00
Minerva Math	Base	25.72	33.95	41.55	48.12	53.37	57.65	61.23	63.95	65.74	66.91
	MaxRL	37.30	44.01	49.06	53.03	56.34	59.11	61.39	63.34	64.93	66.18
	SERA	**38.55**	**44.91**	**50.18**	**54.27**	**57.38**	**60.02**	**62.52**	**65.11**	**67.96**	**70.96**

Table 9 quantifies uncertainty in the differences between SERA and MaxRL. We use $10{,}000$ paired-bootstrap resamples of evaluation questions and report the 2.5th and 97.5th percentiles. The same question indices are used for both methods and all values of K, and differences are computed before rounding.

Table 9: **Qwen3-4B-Base uncertainty.** SERA minus MaxRL in percentage points (top), with pointwise 95% paired-bootstrap intervals (bottom). Differences are computed from unrounded scores before rounding.

Dataset	Pass@32	Pass@128	Pass@256	Pass@512
BeyondAIME	$+5.51$ $[1.69, 9.54]$	$+8.54$ $[3.42, 13.78]$	$+8.41$ $[2.19, 14.74]$	$+8.00$ $[0.00, 17.00]$
AIME 2025	$+3.06$ $[-5.74, 11.67]$	$+2.03$ $[-9.04, 13.86]$	$+0.84$ $[-9.70, 11.88]$	$+0.00$ $[-10.00, 10.00]$
MATH-500	$+0.72$ $[-0.05, 1.55]$	$+0.00$ $[-0.67, 0.60]$	-0.21 $[-0.89, 0.39]$	-0.40 $[-1.20, 0.40]$
Minerva Math	$+0.91$ $[-1.57, 3.47]$	$+1.77$ $[-0.50, 4.14]$	$+3.03$ $[0.60, 5.56]$	$+4.78$ $[1.84, 8.09]$

D.2.5 Training Dynamics and Allocation Behavior

In the analyzed SmolLM2-360M-Instruct run, SERA achieves stronger multi-sample coverage than MaxRL with fewer cumulative mixed-reward groups and all candidate prompts remaining active. Figure 15 (a) compares both methods under matched rollout budgets and fixed-length sequence normalization (Seqnorm), with token-mean GRPO as a reference. Mixed-reward groups contain both correct and incorrect responses and yield nonzero centered advantages. Their fraction under SERA rises late in training and overtakes MaxRL's, despite a lower cumulative count over steps 30–$2{,}000$. Complementary evidence comes from ImageNet, where SERA improves alignment with the exact cross-entropy gradient (Figure 2 (d)), supporting the motivation to account for finite-rollout gradient scaling when allocating rollouts.

Figure 15 (b) reports the mean per-step active-prompt fraction $|\mathcal{A}_t|/B_t$, where $\mathcal{A}_t$ is the active prompt set and B_t is the candidate-prompt batch size. All candidate prompts remain active throughout the analyzed Maze, SmolLM2, and Qwen2.5-Math-1.5B windows. These task gains therefore occur with rollouts redistributed across the full candidate batch. The observability criterion U_q

guides prompt activation, while rollout allocation targets equalization of the gradient-scaling coefficient κ within the active set.

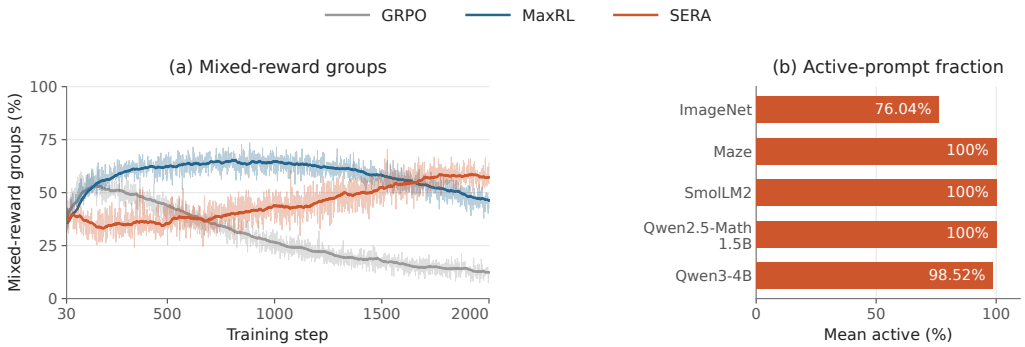

Figure 15: **Mixed-reward groups and active prompts.** (a) Mixed-reward group fraction on SmolLM2, using Seqnorm for MaxRL and SERA and token-mean normalization for GRPO. Faint and solid lines show raw values and trailing 50-step means. (b) Mean active-prompt fraction for SERA across tasks.

Figure 16 presents complementary training diagnostics on SmolLM2-360M-Instruct. MaxRL and SERA use matched rollout budgets within each normalization, with GRPO included in the token-mean comparison. SERA maintains higher late-training token entropy than MaxRL under both normalizations and than GRPO under token-mean normalization. These entropy trends accompany its stronger multi-sample coverage. Under token-mean normalization, MaxRL's mean response length increases markedly late in training, while GRPO and SERA produce shorter responses. Under Seqnorm, the response lengths of MaxRL and SERA remain closer throughout training. Entropy is measured in nats per valid response token, and response length is averaged over generated responses. The actor gradient ℓ_2 norm is measured after gradient accumulation and before clipping.

Figure 17 extends these diagnostics to Qwen2.5-Math-1.5B over the 600-step training window. All three methods use token-mean normalization and the same total rollout budget per step. SERA maintains higher late-training token entropy than GRPO and MaxRL, accompanied by longer generated responses.

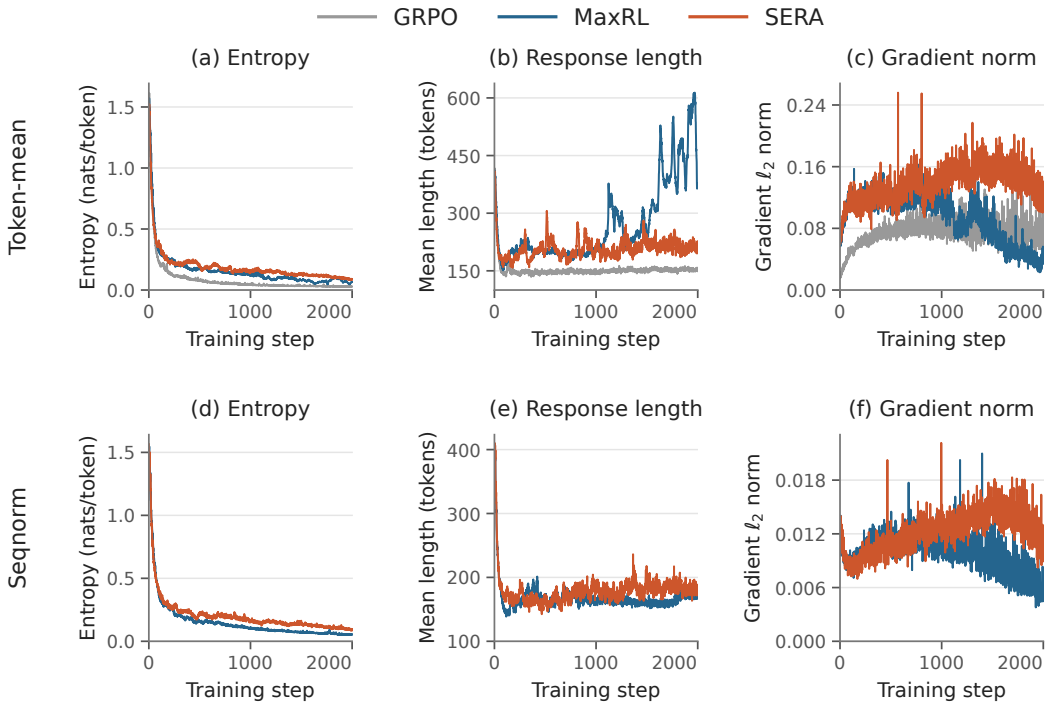

Figure 16: **SmolLM2 training diagnostics.** Rows show token-mean normalization (top) and Seqnorm (bottom); GRPO appears only in the top row. Columns show token entropy, mean response length, and the actor gradient ℓ_2 norm.

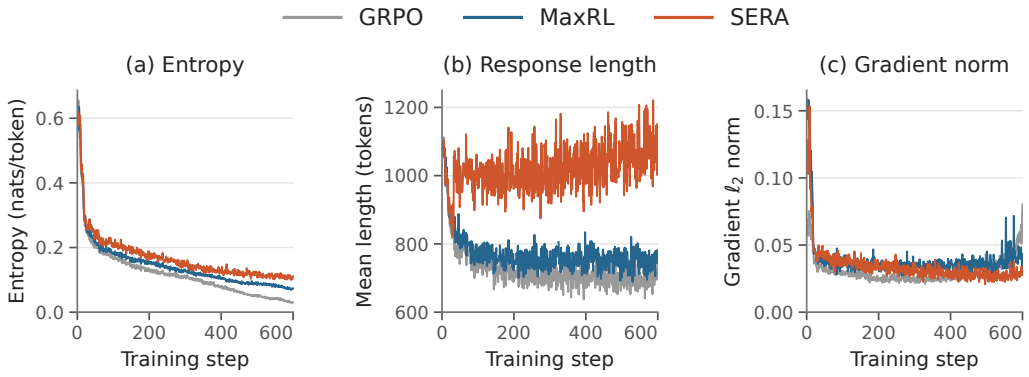

Figure 17: **Qwen2.5-Math-1.5B training diagnostics.** (a) Token entropy. (b) Mean response length. (c) Actor gradient ℓ_2 norm. GRPO, MaxRL, and SERA use token-mean normalization.