BELIEF-SHIFT REWARD: REDISTRIBUTING RLVR CREDIT WITH POLICY-INTERNAL TARGET SUPPORT

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement Learning with Verifiable Rewards (RLVR) provides a scalable framework for improving reasoning post-training, but binary outcome rewards cannot distinguish among trajectories that an external verifier judges correct. To address this limitation, we propose **Belief-Shift Reward (BSR)**, a lightweight reward transformation that preserves the external verifier as the sole criterion of correctness. Within the verifier-correct set, BSR differentiates trajectories by the change each one induces in the policy's support for the verified target relative to the prompt-only prior. Because this prior is shared within a prompt, it cancels under correct-set normalization. BSR therefore requires at most one detached posterior target-scoring pass per verifier-correct trajectory, without an auxiliary reward model or explicit step-level scoring. Across mathematical reasoning, agentic search, and code-output prediction, BSR yields aggregate improvements over matched RLVR baselines, spanning multiple model families and four group-relative optimization objectives. These results support the practical utility of policy-internal target support for finer-grained within-correct credit assignment when the verifier target can be scored by the policy.

1 INTRODUCTION

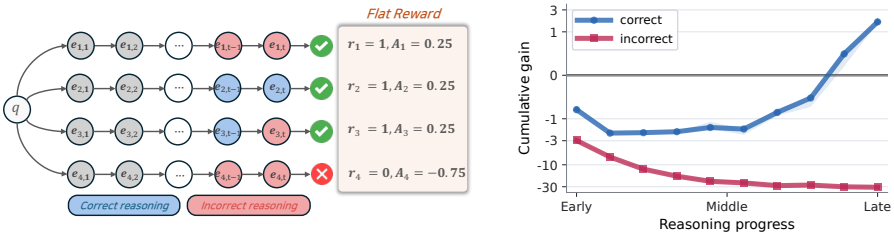

Figure 1: **Empirical motivation for within-correct reward differentiation. Left**: Under binary outcome-supervised GRPO (Shao et al., 2024), verifier-correct rollouts receive identical rewards and relative advantages even when their reasoning quality differs. **Right**: In a pilot study, cumulative target information increasingly separates verifier-correct from verifier-incorrect trajectories over reasoning progress (signed-log y-axis).

Reinforcement Learning with Verifiable Rewards (RLVR) has become a widely adopted paradigm for improving complex reasoning in large language models (LLMs) (Team et al., 2025; Singh et al., 2025; DeepSeek-AI, 2025; Yang et al., 2025), with group-relative methods such as GRPO enabling scalable long-form reasoning with outcome verifiers rather than parameterized reward models (Shao et al., 2024; Schulman et al., 2017; Wei et al., 2022; OpenAI, 2024; Yu et al., 2025). In multi-step reasoning tasks, however, final-answer correctness does not uniquely determine reasoning quality: a binary verifier cannot distinguish a sound derivation from a flawed trajectory that reaches the same target (Uesato et al., 2022; Lightman et al., 2023; Song et al., 2025).

This creates a *within-correct degeneracy* in the reward signal (Lu et al., 2026; Chen et al., 2026; Wang et al., 2026), with two regimes (Xiong et al., 2025; Wei et al., 2025; Zhu et al., 2025). In mixed-outcome groups, all verifier-correct trajectories receive the same positive relative advantage

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.

regardless of reasoning quality (Figure 1, Left). In all-correct groups, the outcome-based relative advantage collapses to zero (Shao et al., 2024; Liu et al., 2025; Yu et al., 2025; Xiao et al., 2025; Ren et al., 2026; Gao et al., 2026). Consequently, binary rewards cannot assign finer relative credit to a sound derivation than to a flawed shortcut when both reach the verified target. Such verifier-correct but flawed trajectories exemplify *spurious correctness* (Shao et al., 2025). Process-supervised approaches can provide finer-grained credit, but often require auxiliary models or dense trajectory processing, adding computational and engineering overhead (Zhang et al., 2025; Khalifa et al., 2025; Dai et al., 2025; Wang et al., 2026; He et al., 2024b).

A natural alternative is to ask whether a trajectory changes the policy's own support for the verified target. We quantify this as *cumulative target information*: the change in target log-probability from the prompt-only prior to the trajectory-conditioned posterior. This signal is target-conditioned, accumulates over reasoning, and requires no auxiliary process evaluator. To examine its relationship with endpoint correctness, we conducted a pilot analysis on QWEN2.5-MATH by tracking cumulative target information along reasoning steps. As shown in Figure 1 (Right), it increasingly separates verifier-correct from verifier-incorrect trajectories as reasoning progresses, motivating its use as a trajectory-level signal for reward differentiation.

In this paper, we propose **Belief-Shift Reward (BSR)**, a lightweight, plug-and-play reward transformation that requires no additional trainable parameters and mitigates within-correct reward flatness. BSR uses the binary verifier to identify the correct subset and a detached policy to score the verified target conditioned on each verifier-correct trajectory; the verifier remains the sole criterion of correctness. These scores are normalized within the correct set to allocate more bonus credit to higher-support trajectories. Since the prompt-only prior cancels under this normalization, BSR requires at most one posterior target-scoring pass per verifier-correct trajectory and no explicit step-level scoring. We use target support only as a within-prompt relative signal, not as an absolute measure of reasoning validity, and evaluate its association with judged reasoning quality through blinded LLM and human evaluations (Section 4.3).

Extensive experiments evaluate BSR in mathematical reasoning, agentic search, and code-output prediction under one shared BSR configuration. In mathematical reasoning, BSR improves the unweighted macro-average in all eight model–objective combinations, although effects on individual benchmarks are mixed; without task-specific retuning, it also improves every non-mathematical model–benchmark comparison. Blinded LLM and human evaluations show an aggregate association between policy-internal target support and judged trajectory quality in the sampled comparisons. The utility of BSR for within-correct credit assignment is evaluated through the matched RLVR training experiments. Training diagnostics further show that BSR substantially reduces the prevalence of groups with zero raw intra-group reward variation during optimization. Our contributions are summarized as follows:

- We characterize within-correct degeneracy in binary group-relative rewards and introduce **BSR**, which differentiates verifier-correct trajectories using policy-internal target support while retaining the external verifier as the sole criterion of correctness.
- Across four RLVR objectives and diverse model families and tasks, BSR yields aggregate gains over matched baselines under a single shared BSR hyperparameter configuration, without an auxiliary reward model or explicit step-level scoring.
- Blinded LLM and human evaluations characterize the association between policy-internal target support and judged trajectory quality in sampled within-prompt comparisons, complementing the matched RLVR training results.

2 PRELIMINARIES

2.1 GROUP RELATIVE POLICY OPTIMIZATION

Group Relative Policy Optimization (GRPO) (Shao et al., 2024) is a critic-free reinforcement learning framework that estimates relative advantages directly from a group of sampled responses, eliminating the need for a separate parameterized value model. Given a prompt q, the rollout policy $\pi_{\theta_{\text{old}}}$ samples a group of G responses, denoted by $y_i \sim \pi_{\theta_{\text{old}}}(\cdot \mid q)$ for $i \in \{1, \ldots, G\}$. Let $y_i = (y_{i,1}, \ldots, y_{i,|y_i|})$ denote the full token sequence of the i-th response. In multi-step reasoning tasks, each response can

be parsed into a generated trajectory e_i and an extracted final outcome o_i, so that $y_i \mapsto (e_i, o_i)$. An outcome verifier then assigns a binary outcome reward:

$$r_i = \mathbb{I}\{\mathrm{Verify}(o_i, o^*) = 1\}, \tag{1}$$

where o^* denotes the verified target. GRPO standardizes these rewards within the sampled group. The group mean μ and standard deviation σ are given by:

$$\mu = \frac{1}{G}\sum_{j=1}^{G} r_j, \qquad \sigma = \sqrt{\frac{1}{G}\sum_{j=1}^{G}(r_j - \mu)^2 + \epsilon_0}, \tag{2}$$

where $\epsilon_0 > 0$ is a small numerical stabilizer. We define the outcome advantage $A_i^{\mathrm{out}} = r_i - \mu$ and its standardized form $\hat{A}_i^{\mathrm{out}} = A_i^{\mathrm{out}}/\sigma$. For exposition, the unclipped reward-driven gradient term at the rollout policy, omitting the KL regularization term, can be written schematically as follows, where the same response-level advantage is applied to every token ($\hat{A}_{i,t}^{\mathrm{out}} = \hat{A}_i^{\mathrm{out}}$):

$$\nabla_\theta \mathcal{J}_{\mathrm{GRPO}}(\theta)|_{\theta=\theta_{\mathrm{old}}} \approx \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|y_i|}\sum_{t=1}^{|y_i|}\hat{A}_i^{\mathrm{out}}\nabla_\theta \log \pi_\theta(y_{i,t} \mid q, y_{i,<t})\right]. \tag{3}$$

Thus, outcome-level GRPO assigns the same response-level advantage to every token, without localizing credit to individual reasoning steps.

2.2 Within-Correct Degeneracy of Binary Outcome Rewards

The binary nature of the outcome reward ($r_i \in \{0, 1\}$) limits the resolution of the advantage signal. For a group of size G, let $\mathcal{C} = \{i \mid r_i = 1\}$ denote the verifier-correct subset and $K = |\mathcal{C}|$ its size. The group mean is $\mu = K/G$, and the outcome advantage takes only two values:

$$A_i^{\mathrm{out}} = r_i - \mu = \begin{cases} 1 - \frac{K}{G}, & i \in \mathcal{C}, \\ -\frac{K}{G}, & i \notin \mathcal{C}. \end{cases} \tag{4}$$

Since standardization divides all responses by the same positive σ, verifier-correct trajectories remain indistinguishable under the standardized advantage, yielding a *flat advantage signal within the correct set*. Verifier correctness is determined solely by the final outcome and therefore does not preclude flawed or spurious reasoning. This degeneracy appears in two regimes: in mixed-outcome groups ($0 < K < G$), all verifier-correct trajectories receive the same positive advantage regardless of reasoning quality; in all-correct groups ($K = G$), $\mu = 1$ and $\hat{A}_i^{\mathrm{out}} = 0$ for every rollout, causing the reward-driven policy-gradient signal in Equation 3 to vanish.

Process supervision (Zhang et al., 2025; Khalifa et al., 2025; Dai et al., 2025; Wang et al., 2026) can provide finer-grained credit, but often requires auxiliary reward models or explicit step-level scoring, adding computational overhead. This motivates within-correct credit assignment without auxiliary models or explicit step-level scoring.

3 Methodology: Belief-Shift Reward

To address within-correct reward flatness in group-relative RLVR (Section 2.2), we propose **Belief-Shift Reward (BSR)**, a reward transformation for RLVR objectives with binary correctness verification. BSR requires no additional trainable parameters or auxiliary models and allocates bonus credit among verifier-correct trajectories according to their policy-internal target support. The overall pipeline is illustrated in Figure 2.

3.1 Target Information

We hypothesize that useful reasoning trajectories tend, in aggregate, to increase the policy's support for the verified target o^*. We therefore use this belief change as a trajectory-level proxy for policy-internal target support. We formalize this proxy as the following model-based target log-likelihood shift:

$$\mathcal{G}_i \triangleq \log \pi_\psi(o^* \mid q, e_i) - \log \pi_\psi(o^* \mid q), \tag{5}$$

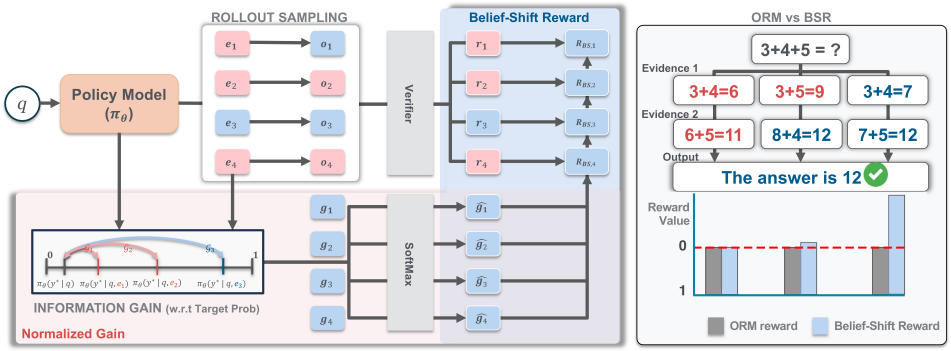

Figure 2: **Belief-Shift Reward (BSR)**. For each prompt q, the policy samples a group of responses, each parsed into a reasoning trajectory e_i and a final outcome o_i. The outcome verifier assigns a binary reward r_i based on the final outcome. Among correct trajectories, BSR measures the detached policy's posterior support for the verified target o^*. These target-support scores are normalized within the correct set and used to allocate a bonus mass λK across the correct trajectories.

which admits a pointwise conditional mutual information interpretation under the induced trajectory–target distribution (Church & Hanks, 1990). We evaluate it using a detached copy π_ψ of the rollout policy, where $\psi = \mathrm{sg}(\theta_{\mathrm{old}})$, and $\pi_\psi(o^* \mid c)$ denotes the policy-induced sequence likelihood of the target under context c. A positive $\mathcal{G}_i$ indicates increased target support relative to the prompt-only prior, whereas a zero or negative value indicates no increase or a decrease in that support. We condition on the trajectory span e_i while excluding the designated final-answer span o_i, avoiding direct leakage from that span.

Let the reasoning trajectory $e_i = (e_{i,1}, \ldots, e_{i,T_i})$ be decomposed into T_i reasoning units. Let $e_{i,\le j}$ denote the prefix up to step j and $e_{i,<j}$ the prefix before step j. The target-information increment associated with step j is

$$\mathcal{I}_{i,j} = \log \pi_\psi(o^* \mid q, e_{i,\le j}) - \log \pi_\psi(o^* \mid q, e_{i,<j}). \tag{6}$$

Summing these increments over all T_i steps, with $e_{i,\le 0} = \emptyset$, yields a telescoping sum whose intermediate terms cancel:

$$\begin{aligned}\sum_{j=1}^{T_i} \mathcal{I}_{i,j} &= \sum_{j=1}^{T_i} [\log \pi_\psi(o^* \mid q, e_{i,\le j}) - \log \pi_\psi(o^* \mid q, e_{i,<j})] \\ &= \log \pi_\psi(o^* \mid q, e_i) - \log \pi_\psi(o^* \mid q) = \mathcal{G}_i. \end{aligned} \tag{7}$$

Thus, $\mathcal{G}_i$ can be computed from the two boundary terms without explicit step-level scoring and is invariant to how the same trajectory is partitioned into reasoning units.

3.2 Correct-Set Normalization

Recall the verifier-correct subset $\mathcal{C}$ and its size K from Section 2.2. To differentiate these trajectories by target support, we normalize over the correct subset using a temperature-scaled softmax. Writing $s_i = \log \pi_\psi(o^* \mid q, e_i)$ for the posterior target-support score, the prompt-only prior in $\mathcal{G}_i = s_i - \log \pi_\psi(o^* \mid q)$ is shared across trajectories for the same prompt and therefore cancels inside the softmax:

$$\hat{g}_i = K \cdot \frac{\exp(s_i/\tau)}{\sum_{j \in \mathcal{C}} \exp(s_j/\tau)}, \qquad \forall i \in \mathcal{C}, \tag{8}$$

where $\tau > 0$ controls the sharpness of the distribution. The multiplier K gives the weights unit mean over the correct set (i.e., $\frac{1}{K}\sum_{i \in \mathcal{C}} \hat{g}_i = 1$). A lower τ concentrates the allocation on high-support trajectories, whereas a larger τ smooths it. Thus, $\mathcal{G}_i$ provides the belief-shift interpretation, while BSR operationally uses only the posterior scores s_i and requires at most one detached target-scoring pass per verifier-correct trajectory (Appendix B.5). Equation 8 applies when $K \ge 1$; for $K = 0$, BSR retains the underlying base rewards, and for $K = 1$, $\hat{g}_i = 1$, so target scoring can be skipped.

3.3 Reward Redistribution and Advantage Dynamics

For clarity, we present BSR in the binary GRPO setting; objective-specific extensions are detailed in Appendix B. BSR modifies the binary outcome reward using the normalized target-support weight:

$$R_{\mathrm{BSR},i} = \begin{cases} 1 + \lambda \hat{g}_i, & i \in \mathcal{C}, \\ 0, & i \notin \mathcal{C}, \end{cases} \tag{9}$$

where $\lambda > 0$ sets the raw scale of the belief-shift bonus. Because $\hat{g}_i > 0$, BSR allocates a total bonus $\sum_{i \in \mathcal{C}} \lambda \hat{g}_i = \lambda K$, fixed conditional on the correct-set size K, across the correct set without reducing any verifier-correct trajectory below its base reward of 1. Lower-support trajectories nevertheless receive smaller rewards relative to other verifier-correct trajectories, thereby shaping their group-relative advantages.

Advantage Decomposition. By the unit-mean property of $\hat{g}_i$, the group mean reward is $\mu_{BSR} = \frac{K}{G}(1 + \lambda)$. For a verifier-correct rollout $i \in \mathcal{C}$, the centered raw advantage decomposes as

$$A_i^{\mathrm{BSR}} = \underbrace{\left(1 - \frac{K}{G}\right)}_{\text{Base outcome advantage}} + \underbrace{\lambda \left(\hat{g}_i - \frac{K}{G}\right)}_{\text{Belief-shift modulation}}. \tag{10}$$

This raw-reward decomposition is unchanged, but its optimizer-facing scale depends on objective-specific advantage normalization (Appendix B.2). In the all-correct regime ($K = G$), the base outcome advantage is zero, while BSR retains the zero-mean relative signal $A_i^{\mathrm{BSR}} = \lambda(\hat{g}_i - 1)$. This signal is non-degenerate precisely when the weights vary within the group ($\mathrm{Var}_{i \in \mathcal{C}}(\hat{g}_i) > 0$); if all verifier-correct trajectories receive identical target support, the group remains flat. BSR therefore introduces within-correct differentiation in mixed groups and can recover a non-zero relative signal in all-correct groups when target-support scores vary.

Negative-Advantage Regime. From Equation 10, a verifier-correct trajectory has negative centered raw advantage if and only if

$$\hat{g}_i < \frac{K}{G} - \frac{1}{\lambda}\left(1 - \frac{K}{G}\right). \tag{11}$$

Writing $\rho = K/G$, the threshold is positive only when $\rho > 1/(1 + \lambda)$. When $\rho \leq 1/(1 + \lambda)$, every verifier-correct rollout has positive centered raw advantage because $\hat{g}_i > 0$; above this success rate, sufficiently low-support verifier-correct trajectories can have negative advantage. Under the default $\lambda = 0.5$, this is possible only when more than two-thirds of the group is verifier-correct. Throughout, external verifier alone defines $\mathcal{C}$; target-support scores are computed only for verifier-correct trajectories, while incorrect trajectories retain their base reward.

4 Experiments

4.1 Experimental Settings

We evaluate BSR in three settings: mathematical reasoning, agentic search, and code-output prediction. All experiments use the same BSR configuration (τ=0.5, λ=0.5); mathematical reasoning evaluates four RLVR objectives, while the two non-mathematical settings use GRPO as the common baseline. Detailed configurations are provided in Appendix D. **(i) Mathematical Reasoning.** We post-train Qwen2.5-Math-1.5B (Yang et al., 2024b) and Qwen2.5-3B (Yang et al., 2024a) with GRPO (Shao et al., 2024), Dr.GRPO (Liu et al., 2025), DAPO (Yu et al., 2025), and BNPO (Xiao et al., 2025). Training uses Levels 3 to 5 of MATH (Hendrycks et al., 2021), and evaluation reports greedy pass@1 accuracy on 1,558 held-out problems across MATH500 (Hendrycks et al., 2021), AIME 2024, AMC 2023 (Li et al., 2024), Minerva Math (Lewkowycz et al., 2022), and OlympiadBench (He et al., 2024a), using symbolic-equivalence verification. **(ii) Agentic Search.** We train four models—Qwen2.5-3B, Qwen2.5-7B, Llama-3.2-3B-Instruct (Grattafiori et al., 2024), and Phi-3.5-mini-instruct (Abdin et al., 2024)—on the NQ training split (Kwiatkowski et al., 2019) with exact-match verification. We report exact-match accuracy on the NQ evaluation split and HotpotQA (Yang et al., 2018). **(iii) Code-Output Prediction.** We train Qwen2.5-7B, Qwen2.5-3B, Qwen2.5-3B-Instruct, and Llama-3.2-3B-Instruct on CodeIO-PyEdu (Li et al., 2025), using reference execution outputs for outcome verification, and report accuracy on LeetCode-O (Li et al., 2025).

Table 1: Greedy-decoding accuracy (%). **Overall** is the unweighted macro-average across the five benchmarks. Parentheses report percentage-point gains over the corresponding baseline. Bold indicates the higher score between each baseline and its BSR counterpart.

RLVR Method	Variant	MATH500	AIME24	AMC23	Minerva	OlympiadBench	Overall
QWEN2.5-MATH-1.5B							
Base Model	-	37.8	10.0	36.6	11.4	22.6	23.7
GRPO (Shao et al., 2024)	Baseline	68.4	13.3	**50.0**	19.9	33.8	37.1
	w/ BSR	**73.2**	13.3	46.3	**27.2**	**35.8**	**39.2** (+2.1)
Dr.GRPO (Liu et al., 2025)	Baseline	71.4	13.3	45.1	22.4	32.6	37.0
	w/ BSR	**75.4**	13.3	**46.3**	**31.6**	**36.8**	**40.7** (+3.7)
DAPO (Yu et al., 2025)	Baseline	68.0	10.0	43.9	**24.3**	32.9	35.8
	w/ BSR	**70.0**	**13.3**	**48.8**	23.2	**34.1**	**37.9** (+2.1)
BNPO (Xiao et al., 2025)	Baseline	68.0	6.7	**46.3**	25.0	36.6	36.5
	w/ BSR	**75.2**	6.7	43.9	**32.4**	**36.9**	**39.0** (+2.5)
QWEN2.5-3B							
Base Model	-	55.8	6.7	25.6	15.1	22.1	25.1
GRPO (Shao et al., 2024)	Baseline	64.4	3.3	34.1	**27.6**	27.4	31.4
	w/ BSR	**66.2**	3.3	**43.9**	25.0	**31.5**	**34.0** (+2.6)
Dr.GRPO (Liu et al., 2025)	Baseline	66.0	3.3	31.7	**26.8**	**30.1**	31.6
	w/ BSR	**66.6**	**6.7**	**35.4**	23.9	29.5	**32.4** (+0.8)
DAPO (Yu et al., 2025)	Baseline	65.8	**13.3**	30.5	23.5	**30.4**	32.7
	w/ BSR	**67.8**	3.3	**39.0**	**27.6**	30.1	**33.6** (+0.9)
BNPO (Xiao et al., 2025)	Baseline	**67.0**	3.3	34.1	20.2	**30.7**	31.1
	w/ BSR	64.2	**10.0**	**35.4**	**26.1**	28.2	**32.8** (+1.7)

4.2 MAIN RESULTS

We evaluate BSR using the shared configuration described in Section 4.1. BSR yields a higher Overall score in all eight mathematical model–objective combinations and all twelve non-mathematical model–benchmark comparisons.

Mathematical Reasoning. Table 1 reports results across two model scales, four RLVR objectives, and five benchmarks. Although effects on individual benchmarks are mixed, BSR improves the unweighted five-benchmark macro-average in all eight model–objective combinations, with gains ranging from $+0.8$ to $+3.7$ percentage points (pp). This pattern across GRPO, Dr.GRPO, DAPO, and BNPO at both model scales supports the applicability of BSR across different group-relative optimization objectives.

Table 2: Agentic-search exact-match accuracy (%). NQ is the in-domain benchmark, and HotpotQA is the out-of-domain benchmark. **Overall** is the average across the four models. Parentheses report percentage-point gains of BSR over GRPO. **Base** denotes the pre-RL model.

	NQ			HotpotQA		
Model	**Base**	**GRPO**	**w/ BSR**	**Base**	**GRPO**	**w/ BSR**
Qwen2.5-7B	10.6	36.6	**37.6**	12.1	27.7	**30.6**
Qwen2.5-3B	3.9	26.7	**31.6**	4.2	21.4	**23.5**
Llama-3.2-3B-Instruct	16.5	34.1	**37.6**	12.9	20.1	**23.2**
Phi-3.5-mini-instruct	10.7	20.9	**29.9**	12.3	15.6	**23.0**
Overall	10.4	29.6	**34.2** (+4.6)	10.4	21.2	**25.1** (+3.9)

Agentic Search. Table 2 shows consistent improvements on both the in-domain NQ and the out-of-domain HotpotQA benchmarks. BSR yields mean gains of $+4.6$ and $+3.9$ pp on NQ and HotpotQA, respectively. The largest gains occur for Phi-3.5-mini-instruct ($+9.0$ pp on NQ and $+7.4$ pp on

Table 3: Code-output prediction accuracy (%) on LeetCode-O. **Overall** is the average across the four models. † denotes instruction-tuned models.

Method	Qwen2.5-7B	Qwen2.5-3B	Qwen2.5-3B†	Llama-3.2-3B†	Overall
Base	30.8	23.3	30.2	26.6	27.7
GRPO	45.4	30.8	31.1	25.2	33.1
w/ BSR	**48.4**	**35.2**	**32.6**	**26.5**	**35.7** (+2.6)

HotpotQA), while the overall pattern spans multiple model families and transfers to the out-of-domain benchmark.

Code-Output Prediction. Table 3 shows that BSR improves the GRPO result for all four models, increasing mean accuracy from 33.1% to 35.7% (+2.6 pp). The gains are consistent across both base and instruction-tuned models, ranging from +1.3 to +4.4 pp. Unlike mathematical reasoning, which uses symbolic-equivalence verification, code-output prediction compares the predicted execution output with the reference output. These gains are obtained with the shared BSR hyperparameters, without task-specific retuning.

4.3 Analysis

Table 4: IG-Min versus IG-Max across mathematical reasoning and agentic search. Each cell reports IG-Min → IG-Max; the parenthesized Overall value reports the corresponding gap.

	LLM-Judge Score		Endpoint Accuracy (%)	
Model	**Math**	**Search**	**Math**	**Search**
Llama-3.2-3B	31.2 → 58.2	25.2 → 54.4	5.2 → 55.9	1.4 → 26.3
Mistral-7B	20.5 → 31.1	39.7 → 67.6	1.6 → 20.8	3.1 → 17.8
Qwen2.5-7B	60.8 → 77.0	47.0 → 69.3	39.8 → 65.4	11.0 → 34.0
Qwen2.5-3B	49.2 → 71.3	31.5 → 57.4	25.1 → 68.5	5.0 → 24.6
Overall	40.4 → **59.4** (+19.0)	35.9 → **62.2** (+26.3)	17.9 → **52.7** (+34.7)	5.1 → **25.7** (+20.6)

Investigating the Belief-Shift Signal. We refer to cumulative target information $\mathcal{G}_i$ as information gain (IG) in this analysis. To examine its association with trajectory-level outcomes and judged quality, we compare the highest- and lowest-IG trajectories from each sampled prompt group, without conditioning pair selection on verifier correctness. Across all 8,000 offline within-prompt pairs, pooled irrespective of endpoint outcomes, a blinded LLM judge prefers IG-Max in 66.7% of cases and IG-Min in 24.0%, with a 22.7-point higher mean checklist score for IG-Max (Appendix E.2). Table 4 reports higher mean judge scores and endpoint accuracy for IG-Max in all eight model-domain settings. Averaged across models, the judge-score gaps are 19.0 and 26.3 points for mathematical reasoning and agentic search, respectively, with endpoint-accuracy gaps of 34.7 and 20.6 pp. Separately, the joint endpoint outcomes show that IG-Max alone is correct in 2,320 pairs (29.0%), whereas IG-Min alone is correct in 109 pairs (1.4%; Table 12). In a complementary human study of 30 NQ pairs, IG-Max is required to be verifier-correct, while IG-Min is not conditioned on endpoint correctness; blinded annotators favor IG-Max in 90.0%–94.0% of judgments on logical quality, search quality, and evidence faithfulness (Appendix E.3). These comparisons characterize aggregate associations in the sampled pairs rather than directly validating reasoning-quality rankings within the verifier-correct set.

Sentence-Level Progress and Recovery. We pair standardized sentence-level IG increments with blinded progress labels for 332 search rollouts and, to match BSR's operating regime, analyze the 45 endpoint-correct rollouts; Appendix E.4 details the protocol. A pass denotes relevant, non-redundant progress. The mean increments are positive when progress continues or recovers (pass → pass: +0.429; fail → pass: +0.192) and negative when reasoning remains unproductive or regresses

Table 5: High- and low-support verifier-correct trajectories from an all-correct group for the query "Find the greatest whole number satisfying $4x - 3 < 2 - x$" (gold answer: 0; $K/G = 16/16$, $\mu_{\text{BSR}} = 1.5$). **Judge** is a blinded offline checklist score and is not used by BSR. Red marks an invalid step; blue highlights reasoning that supports the verified target. Values are rounded to two decimal places; $\hat{g}_i$ is strictly positive before rounding.

Support	Trajectory	$\mathcal{G}_i$	$\hat{g}_i$	A_i^{BSR}	Judge
High	Add x to both sides: $4x + x - 3 < 2$, so $5x - 3 < 2$. Add 3: $5x < 5$. Divide by 5: $x < 1$. The whole numbers are $0, 1, 2, 3, \ldots$, and the greatest whole number less than 1 is 0.	13.60	6.22	2.61	100
Low	Moving x-terms to one side and constants to the other: $4x + x < 2 + 3$, so $5x < 5$ and $x < 1$. The whole numbers less than 1 are $0, -1, -2, -3, \ldots$, and the greatest whole number less than 1 is 0.	9.50	0.00	-0.50	75

(fail → fail: -0.261; pass → fail: -0.312; Table 13). This average sign pattern is consistent with local target-information changes covarying with judged reasoning progress and recovery.

Qualitative Case Study: Within-Correct Credit Differentiation. Table 5 illustrates BSR in an all-correct group ($K = G = 16$), where binary outcome supervision assigns identical rewards and zero relative advantage to every rollout. Despite sharing the same verified outcome, the trajectories receive different policy-internal target support, which BSR converts into differentiated centered raw advantages through $A_i^{\text{BSR}} = \lambda(\hat{g}_i - 1)$. In the shown example, the lowest-support rollout contains an incorrect statement about whole numbers while still reaching the verified answer, whereas the highest-support rollout gives a valid derivation. This example is illustrative rather than diagnostic: BSR ranks verifier-correct trajectories by relative target support rather than directly detecting reasoning errors. Appendix H provides the full-group analysis and additional mixed-outcome cases.

Mitigating Reward Saturation. To diagnose groupwise reward flatness during training, we monitor the fraction of prompt groups whose raw rewards have zero within-group variation. For each method, we define $\sigma_{\text{raw}} = \sqrt{\frac{1}{G}\sum_i (R_i - \mu)^2}$ and count groups with $\sigma_{\text{raw}} = 0$. Under binary GRPO, all-correct and all-incorrect groups have zero raw reward variation and therefore no group-relative outcome advantage. As shown in Figure 3, this fraction increases substantially under GRPO over training, reaching roughly 40% near the end of training compared with approximately 10% under BSR—a 30.1 percentage-point gap. The widening gap is consistent with reward saturation becoming increasingly prevalent under GRPO, whereas BSR preserves non-zero reward variation in substantially more prompt groups. In particular, BSR can break the zero-variation condition in all-correct groups whenever target-support scores differ, thereby providing a non-zero group-relative advantage signal that binary rewards cannot supply in such groups.

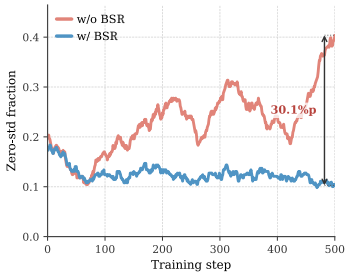

Figure 3: Fraction of training prompt groups with zero raw intra-group reward variation during Qwen2.5-Math-1.5B training.

Hyperparameter Sensitivity and Raw-Advantage Dynamics. At the raw-reward level, λ scales the belief-shift bonus and determines the negative-advantage threshold in Equation 11, whereas τ controls the concentration of the normalized target-support weights. As shown in Figure 4(c), larger λ or smaller τ increases the fraction of verifier-correct trajectories receiving negative centered raw advantage, becoming highly prevalent in the most aggressive setting shown. The one-dimensional sweeps attain their highest MATH500 point estimates at intermediate values and yield lower accuracy toward both the weak and aggressive extremes (Figures 4(a) and (b)). Specifically, smaller λ weakens raw bonus differences, whereas larger τ flattens the correct-set weights; larger λ or smaller τ instead produces stronger advantage polarization. Our default configuration (τ=0.5, λ=0.5) was fixed before these post hoc sweeps, lies in this intermediate regime, and was used unchanged across experimental domains. Detailed configurations for Figure 4(c) are provided in Appendix G.

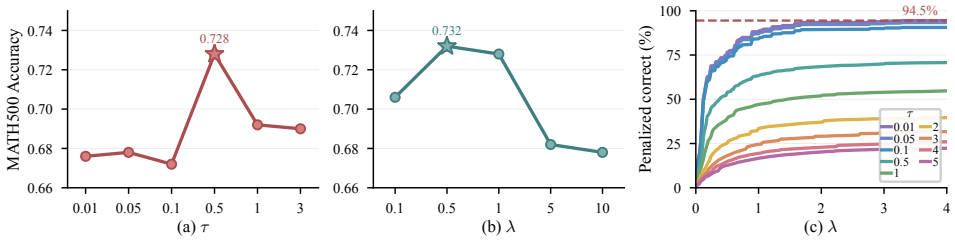

Figure 4: **(a)** MATH500 accuracy across sharpening temperature τ with λ=1.0. **(b)** MATH500 accuracy across modulation strength λ with τ=0.5. Stars mark the highest observed MATH500 accuracy in each one-dimensional sweep. **(c)** Fraction of verifier-correct trajectories receiving negative centered raw advantage as a function of λ for varying τ.

5 Related Work

Outcome and Process Supervision in RLVR. RLVR provides a scalable approach to reasoning post-training through verifiable outcome rewards, with GRPO estimating group-relative advantages without a learned critic (Cobbe et al., 2021; Shao et al., 2024; DeepSeek-AI, 2025). Subsequent methods improve this framework through sampling, clipping, length control, or advantage normalization (Yu et al., 2025; Liu et al., 2025; Xiao et al., 2025; Kazemnejad et al., 2024). However, these optimization-side advances largely retain the same coarse outcome-reward interface, and recent analyses caution that higher endpoint accuracy need not imply better reasoning quality (Shao et al., 2025; Yue et al., 2025; Wen et al., 2025). Process supervision addresses this limitation by assigning finer-grained credit within reasoning traces. Step-level feedback and process reward models can improve reasoning beyond final-answer supervision (Uesato et al., 2022; Lightman et al., 2023), while later work reduces annotation cost through automatically constructed or model-derived process signals (Wang et al., 2023; Luo et al., 2024; Lu et al., 2024; Wang et al., 2024; Zhang et al., 2024; Sun et al., 2025) or directly optimizes step-level preferences (Lai et al., 2024; Lin et al., 2025). Rewarding Progress is closely related in motivation, rewarding intermediate reasoning according to whether it increases the likelihood of eventual correctness (Setlur et al., 2024). Nevertheless, existing process-supervision approaches require step-level localization, repeated scoring, or auxiliary models (Song et al., 2025; Zheng et al., 2024; Zhang et al., 2025; Khalifa et al., 2025; Cheng et al., 2025).

Policy-Internal Belief and Information Signals. Recent methods derive finer-grained credit from policy likelihoods, intermediate answer beliefs, or information-theoretic quantities (Yuan et al., 2024; Cui et al., 2025; Sullivan & Koller, 2025; Wang et al., 2026; Fei et al., 2025; Tan et al., 2025; Lee et al., 2026; He et al., 2026). PACR rewards increases in model confidence along a trajectory (Yoon et al., 2025), while IGPO uses information gain from changes in target support across interaction turns (Wang et al., 2025). PAPO combines outcome advantage with a process advantage over correct responses using a rubric-based process reward model (Tan et al., 2026). BSR likewise seeks finer resolution beyond binary correctness, but uses the detached policy's posterior support for the verified target as a within-prompt relative signal only within the verifier-correct set. In contrast to PACR and IGPO, which explicitly evaluate target-confidence or information changes across intermediate reasoning steps or interaction turns, BSR exploits the telescoping form of cumulative target information and correct-set normalization to avoid explicit step-level scoring and cancel the shared prompt-only prior.

6 Conclusion

We introduced **Belief-Shift Reward (BSR)**, a reward transformation that enables finer within-correct credit assignment in RLVR using policy-internal target support while preserving the external verifier as the sole criterion of correctness. BSR requires no auxiliary reward model or explicit step-level scoring and allocates bonus credit among verifier-correct trajectories according to their relative support for the verified target. Across three domains, BSR yields higher five-benchmark macro-averages in all eight mathematical model–objective settings and higher scores in every matched non-mathematical comparison. These results support the practical utility of policy-internal target support for within-correct credit assignment when the verifier target can be scored by the policy.

AI Use Statement. We used generative AI tools for manuscript drafting and editing, formatting, limited coding assistance, and critical feedback on mathematical formulations, methodology, experiments, and result interpretation. LLM-based judges were used for the offline analyses described in Appendix E; their judgments were not used to construct training rewards. The authors reviewed the AI-assisted text, code, and analyses and take responsibility for the final methodology, claims, and reported results.

Reproducibility Statement. The main experimental setup, including the evaluated tasks, models, objectives, and shared BSR configuration, is summarized in Section 4.1. Detailed training configurations, datasets, verifier and scoring procedures, decoding settings, and evaluation protocols for all experimental domains are provided in Appendix D. The sampling, judging, and annotation protocols underlying the policy-internal signal analyses are documented separately in Appendix E. We additionally provide source code in the Supplementary Materials, including the implementation of BSR.

Ethics Statement. Five AI researchers from the authors' laboratory voluntarily evaluated 30 pairs of model-generated responses using predefined criteria under blinded and randomized presentation. The evaluation involved no intervention, deception, or collection of personal or sensitive information, and all results are reported only in aggregate. Given the minimal-risk nature of the study and the absence of collected personal data, the authors determined that formal ethics review was not required under their institutional procedures.

REFERENCES

Marah I Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Hassan Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, Allie Del Giorno, Gustavo de Rosa, Matthew Dixon, Ronen Eldan, Dan Iter, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Jamie Huynh, Mojan Javaheripi, Xin Jin, Piero Kauffmann, Nikos Karampatziakis, Dongwoo Kim, Mahmoud Khademi, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Chen Liang, Weishung Liu, Xihui (Eric) Lin, Zeqi Lin, Piyush Madan, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Corby Rosset, Sambudha Roy, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Xia Song, Olatunji Ruwase, Xin Wang, Rachel Ward, Guanhua Wang, Philipp Witte, Michael Wyatt, Can Xu, Jiahang Xu, Weijian Xu, Sonali Yadav, Fan Yang, Ziyi Yang, Donghan Yu, Chengruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yunan Zhang, and Xiren Zhou. Phi-3 technical report: A highly capable language model locally on your phone. Technical Report MSR-TR-2024-12, Microsoft, August 2024.

Zhipeng Chen, Tao Qian, Wayne Xin Zhao, and Ji-Rong Wen. Low-rank optimization trajectories modeling for llm rlvr acceleration. *arXiv preprint arXiv:2604.11446*, 2026.

Jie Cheng, Gang Xiong, Ruixi Qiao, Lijun Li, Chao Guo, Junle Wang, Yisheng Lv, and Fei-Yue Wang. Stop summation: Min-form credit assignment is all process reward model needs for reasoning. *arXiv preprint arXiv:2504.15275*, 2025.

Kenneth Church and Patrick Hanks. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29, 1990.

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

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025.

Muzhi Dai, Chenxu Yang, and Qingyi Si. S-grpo: Early exit via reinforcement learning in reasoning models. *arXiv preprint arXiv:2505.07686*, 2025.

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

Wu Fei, Hao Kong, Shuxian Liang, Yang Lin, Yibo Yang, Jing Tang, Lei Chen, and Xiansheng Hua. Self-guided process reward optimization with redefined step-wise advantage for process reinforcement learning. *arXiv preprint arXiv:2507.01551*, 2025.

Haowen Gao, Zhenyu Zhang, Liang Pang, Fangda Guo, Hongjian Dou, Guannan Lv, Shaoguo Liu, Tingting Gao, Huawei Shen, and Xueqi Cheng. Diva-grpo: Enhancing multimodal reasoning through difficulty-adaptive variant advantage. *arXiv preprint arXiv:2603.01106*, 2026.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

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, 2024a.

Jujie He, Tianwen Wei, Rui Yan, Jiacai Liu, Chaojie Wang, Yimeng Gan, Shiwen Tu, Chris Yuhao Liu, Liang Zeng, Xiaokun Wang, et al. Skywork-o1 open series, 2024b.

Yinhan He, Yaochen Zhu, Mingjia Shi, Wendy Zheng, Lin Su, Xiaoqing Wang, Qi Guo, and Jundong Li. Iapo: Information-aware policy optimization for token-efficient reasoning. *arXiv preprint arXiv:2602.19049*, 2026.

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.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL `https://arxiv.org/abs/2310.06825`.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. Vineppo: Refining credit assignment in rl training of llms. *arXiv preprint arXiv:2410.01679*, 2024.

Muhammad Khalifa, Rishabh Agarwal, Lajanugen Logeswaran, Jaekyeom Kim, Hao Peng, Moontae Lee, Honglak Lee, and Lu Wang. Process reward models that think. *arXiv preprint arXiv:2504.16828*, 2025.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*, 2024.

Nakyung Lee, Sangwoo Hong, and Jungwoo Lee. Efficient process reward modeling via contrastive mutual information. *arXiv preprint arXiv:2604.10660*, 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.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, et al. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9, 2024.

Junlong Li, Daya Guo, Dejian Yang, Runxin Xu, Yu Wu, and Junxian He. Codei/o: Condensing reasoning patterns via code input-output prediction. *arXiv preprint arXiv:2502.07316*, 2025.

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

Yen-Ting Lin, Di Jin, Tengyu Xu, Tianhao Wu, Sainbayar Sukhbaatar, Chen Zhu, Yun He, Yun-Nung Chen, Jason Weston, Yuandong Tian, Arash Rahnama, Sinong Wang, Hao Ma, and Han Fang. Step-kto: Optimizing mathematical reasoning through stepwise binary feedback. *arXiv preprint arXiv:2501.10799*, 2025.

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.

Jianqiao Lu, Zhiyang Dou, Hongru Wang, Zeyu Cao, Jianbo Dai, Yingjia Wan, and Zhijiang Guo. Autopsv: Automated process-supervised verifier. *arXiv preprint arXiv:2405.16802*, 2024.

Jinda Lu, Junkang Wu, Jinghan Li, Kexin Huang, Shuo Yang, Mingzhu Chen, Jiancan Wu, Kuien Liu, and Xiang Wang. Beyond where to look: Trajectory-guided reinforcement learning for multimodal rlvr. *arXiv preprint arXiv:2603.26126*, 2026.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, and Abhinav Rastogi. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*, 2024.

OpenAI. Learning to reason with llms, 2024. URL `https://openai.com/index/learning-to-reason-with-llms/`. Accessed: 2024-09-12.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 3505–3506, 2020.

Yanwei Ren, Haotian Zhang, Likang Xiao, Xikai Zhang, Jiaxing Huang, Jiayan Qiu, Baosheng Yu, Quan Chen, and Liu Liu. Recycling failures: Salvaging exploration in rlvr via fine-grained off-policy guidance. *arXiv preprint arXiv:2602.24110*, 2026.

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

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for llm reasoning. *arXiv preprint arXiv:2410.08146*, 2024.

Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, et al. Spurious rewards: Rethinking training signals in rlvr. *arXiv preprint arXiv:2506.10947*, 2025.

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

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Mingyang Song, Zhaochen Su, Xiaoye Qu, Jiawei Zhou, and Yu Cheng. Prmbench: A fine-grained and challenging benchmark for process-level reward models. *arXiv preprint arXiv:2501.03124*, 2025.

Michael Sullivan and Alexander Koller. Grpo is secretly a process reward model. *arXiv preprint arXiv:2509.21154*, 2025.

Lin Sun, Chuang Liu, Xiaofeng Ma, Tao Yang, Weijia Lu, and Ning Wu. Freeprm: Training process reward models without ground truth process labels. *arXiv preprint arXiv:2506.03570*, 2025.

Hongze Tan, Zihan Wang, Jianfei Pan, Jinghao Lin, Hao Wang, Yifan Wu, Tao Chen, Zhihang Zheng, Zhihao Tang, and Haihua Yang. Gtpo and grpo-s: Token and sequence-level reward shaping with policy entropy. *arXiv preprint arXiv:2508.04349*, 2025.

Zelin Tan, Zhouliang Yu, Bohan Lin, Zijie Geng, Hejia Geng, Yudong Zhang, Mulei Zhang, Yang Chen, Shuyue Hu, Zhenfei Yin, et al. Papo: Stabilizing rubric integration training via decoupled advantage normalization. *arXiv preprint arXiv:2603.26535*, 2026.

Kimi Team, Yifan Bai, Yiping Bao, Y Charles, Cheng Chen, Guanduo Chen, Haiting Chen, Huarong Chen, Jiahao Chen, Ningxin Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, and Zhenzhe Ying. Information gain-based policy optimization. *arXiv preprint arXiv:2510.14967*, 2025.

Jingyi Wang, Lei Zhu, Tengjin Weng, Song-Li Wu, Haochen Tan, Jierun Chen, Chaofan Tao, Haoli Bai, Lu Hou, Lifeng Shang, et al. Grpo-vps: Enhancing group relative policy optimization with verifiable process supervision for effective reasoning. *arXiv preprint arXiv:2604.20659*, 2026.

Peiyi Wang, Lei Li, Zhihong Shao, R. X. Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *arXiv preprint arXiv:2312.08935*, 2023.

Zihan Wang, Yunxuan Li, Yuexin Wu, Liangchen Luo, Le Hou, Hongkun Yu, and Jingbo Shang. Multi-step problem solving through a verifier: An empirical analysis on model-induced process supervision. *arXiv preprint arXiv:2402.02658*, 2024.

Chenxing Wei, Jiarui Yu, Ying Tiffany He, Hande Dong, Yao Shu, and Fei Yu. Redit: Reward dithering for improved llm policy optimization. *arXiv preprint arXiv:2506.18631*, 2025.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

Xumeng Wen, Zihan Liu, Shun Zheng, Shengyu Ye, Zhirong Wu, Yang Wang, Zhijian Xu, Xiao Liang, Junjie Li, Ziming Miao, Jiang Bian, and Mao Yang. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base llms. *arXiv preprint arXiv:2506.14245*, 2025.

Changyi Xiao, Mengdi Zhang, and Yixin Cao. Bnpo: Beta normalization policy optimization. *arXiv preprint arXiv:2506.02864*, 2025.

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 for reinforce-style llm training. *arXiv e-prints*, pp. arXiv–2510, 2025.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a.

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*, 2025.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

Eunseop Yoon, Hee Suk Yoon, Jaehyun Jang, SooHwan Eom, Qi Dai, Chong Luo, Mark A Hasegawa-Johnson, and Chang D Yoo. Pacr: Progressively ascending confidence reward for llm reasoning. *arXiv preprint arXiv:2510.22255*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. Free process rewards without process labels. *arXiv preprint arXiv:2412.01981*, 2024.

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? *arXiv preprint arXiv:2504.13837*, 2025.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative verifiers: Reward modeling as next-token prediction. *arXiv preprint arXiv:2408.15240*, 2024.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*, 2025.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Processbench: Identifying process errors in mathematical reasoning. *arXiv preprint arXiv:2412.06559*, 2024.

Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. The surprising effectiveness of negative reinforcement in llm reasoning. *arXiv preprint arXiv:2506.01347*, 2025.

A ALGORITHM

Algorithm Summary. Algorithm 1 summarizes the binary GRPO instantiation of BSR. For each prompt, BSR scores only verifier-correct trajectories using the posterior target-support score $s_i = \log \pi_\psi(o^* \mid q, e_i)$. The telescoping formulation avoids explicit step-level scoring, while correct-set normalization cancels the shared prompt-only prior. Thus, BSR requires at most one posterior target-scoring pass per verifier-correct trajectory; groups with $K \leq 1$ require none.

Algorithm 1 Belief-Shift Reward (BSR)

Input: Prompt dataset $\mathcal{D}$; base policy π_{base}; group size G; temperature τ; modulation strength λ; training steps S
Output: Updated policy parameters θ

Initialize policy $\pi_\theta \leftarrow \pi_{\text{base}}$
for iteration $= 1, \ldots, S$ **do**
  Sample a mini-batch of prompts $\mathcal{D}_b \subset \mathcal{D}$
  Set rollout policy $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$
  Set detached scoring policy $\pi_\psi \leftarrow \text{sg}(\pi_{\theta_{\text{old}}})$
  for each prompt $q \in \mathcal{D}_b$ **with** verified target o^* **do**
    Generate responses $y_1, \ldots, y_G \sim \pi_{\theta_{\text{old}}}(\cdot \mid q)$
    Parse reasoning trajectory e_i and final outcome o_i from each response y_i
    Compute binary outcome rewards $r_i = \mathbb{I}\{\text{Verify}(o_i, o^*) = 1\}$
    Let $\mathcal{C} = \{i \mid r_i = 1\}$ and $K = |\mathcal{C}|$
    if $K \leq 1$ **then**
      $R_{\text{BSR},i} \leftarrow r_i(1 + \lambda)$ for all $i \in \{1, \ldots, G\}$ {*No BSR target scoring needed*}
    else
      for each $j \in \mathcal{C}$ **do**
        Evaluate posterior target-support score $s_j = \log \pi_\psi(o^* \mid q, e_j)$
      end for
      for each $j \in \mathcal{C}$ **do**
        Compute normalized belief-shift weight $\hat{g}_j = K \frac{\exp(s_j/\tau)}{\sum_{k \in \mathcal{C}} \exp(s_k/\tau)}$
        Set $R_{\text{BSR},j} \leftarrow 1 + \lambda \hat{g}_j$
      end for
      Set $R_{\text{BSR},i} \leftarrow 0$ for all $i \notin \mathcal{C}$
    end if
    Compute standardized group-relative advantages $\hat{A}_i^{\text{BSR}}$ from $\{R_{\text{BSR},i}\}_{i=1}^G$
  end for
  Update π_θ using the GRPO objective with advantages $\hat{A}_i^{\text{BSR}}$
end for

B ADDITIONAL DERIVATIONS AND EXTENSIONS

B.1 ALTERNATIVE BASE REWARD BOUNDS

Section 3 instantiates BSR with binary rewards $r_i \in \{0, 1\}$. Other RLVR objectives may use different two-level outcome rewards; for example, our DAPO implementation uses $r'_i \in \{-1, 1\}$ (Yu et al., 2025). Let the correct and incorrect base rewards be b_+ and b_-, respectively, with $b_+ > b_-$, so that $r'_i = b_+$ for $i \in \mathcal{C}$ and $r'_i = b_-$ otherwise. BSR extends directly as

$$R'_{\text{BSR},i} = \begin{cases} b_+ + \lambda \hat{g}_i, & i \in \mathcal{C}, \\ b_-, & i \notin \mathcal{C}. \end{cases} \tag{12}$$

Writing $\Delta = b_+ - b_- > 0$, the group mean for a group containing K verifier-correct trajectories is

$$\mu'_{\text{BSR}} = b_- + \Delta \frac{K}{G} + \lambda \frac{K}{G}. \tag{13}$$

For a correct trajectory $i \in \mathcal{C}$, the advantage therefore becomes

$$A'_i = \underbrace{\Delta\left(1 - \frac{K}{G}\right)}_{\text{Base outcome advantage}} + \underbrace{\lambda\left(\hat{g}_i - \frac{K}{G}\right)}_{\text{Belief-shift modulation}} . \tag{14}$$

Thus, changing the two-level base reward bounds preserves the additive form of the belief-shift modulation at the level of centered raw rewards, while the reward gap Δ determines its strength relative to the base outcome signal. A verifier-correct trajectory receives negative advantage when

$$\hat{g}_i < \frac{K}{G} - \frac{\Delta}{\lambda}\left(1 - \frac{K}{G}\right). \tag{15}$$

Writing $\rho = K/G$, this threshold is positive only when $\rho > \Delta/(\Delta + \lambda)$, recovering $\rho > 1/(1 + \lambda)$ for the binary $\{0, 1\}$ case.

For the $\{-1, 1\}$ rewards used in our DAPO experiments, $\Delta = 2$, yielding

$$A'_i = 2\left(1 - \frac{K}{G}\right) + \lambda\left(\hat{g}_i - \frac{K}{G}\right), \tag{16}$$

with the corresponding threshold for negative advantage

$$\hat{g}_i < \frac{K}{G} - \frac{2}{\lambda}\left(1 - \frac{K}{G}\right). \tag{17}$$

Incorrect trajectories retain their raw base reward. Their centered or normalized advantages can nevertheless change indirectly because BSR changes the group mean and, when applicable, the group normalization factor.

B.2 Effect of Advantage Normalization

Equation 10 describes the centered raw advantage. For objectives that additionally divide by the within-group reward standard deviation, let $\rho = K/G$ and, for $K \geq 1$,

$$v_g = \frac{1}{K}\sum_{j \in \mathcal{C}} (\hat{g}_j - 1)^2. \tag{18}$$

For the binary BSR reward,

$$\sigma^2_{\mathrm{BSR}} = \rho(1 - \rho)(1 + \lambda)^2 + \rho\lambda^2 v_g + \epsilon_0, \qquad \hat{A}^{\mathrm{BSR}}_i = \frac{A^{\mathrm{BSR}}_i}{\sigma_{\mathrm{BSR}}}. \tag{19}$$

In an all-correct group ($\rho = 1$), this reduces to

$$\hat{A}^{\mathrm{BSR}}_i = \frac{\lambda(\hat{g}_i - 1)}{\sqrt{\lambda^2 v_g + \epsilon_0}}. \tag{20}$$

Thus, standardization preserves the sign of the centered raw advantage, and therefore the negative-advantage condition in Equation 11, while changing its magnitude. When ϵ_0 is negligible, the global scale λ cancels in the all-correct regime; objectives without this standard-deviation divisor are not subject to this cancellation.

B.3 Policy-Gradient Effect of Negative Advantage

For the binary GRPO instantiation, $\sigma_{\mathrm{BSR}} > 0$, so standardization preserves the sign of the centered raw advantage. Hence, if a verifier-correct trajectory satisfies Equation 11, then $A^{\mathrm{BSR}}_i < 0$.

$$\nabla_\theta \mathcal{J}_i(\theta)|_{\theta=\theta_{\mathrm{old}}} \propto -\frac{|A^{\mathrm{BSR}}_i|}{\sigma_{\mathrm{BSR}}} \frac{1}{|y_i|} \nabla_\theta \log \pi_\theta(y_i \mid q)|_{\theta=\theta_{\mathrm{old}}} . \tag{21}$$

Considered in isolation under gradient ascent, this contribution locally decreases the sampled response's log-likelihood. For Dr.GRPO, which omits the standard-deviation divisor, the same sign interpretation applies directly to the centered raw advantage. Whether a verifier-correct trajectory enters this regime depends on its relative target support and the group success rate; BSR neither assumes that every flawed trajectory receives low target support nor identifies the erroneous reasoning step.

B.4 Algebraic Expansion of the Telescoping Identity

In Section 3.1, we established that the cumulative target information $\mathcal{G}_i$ exactly decomposes into the sum of step-wise target-information increments $\mathcal{I}_{i,j}$. We demonstrate the algebraic mechanics of this telescoping reduction and show why the cumulative trajectory-level quantity can be obtained from its boundary terms without evaluating every intermediate increment.

Recall that the target-information increment contributed by reasoning step j is defined under the detached scoring policy π_ψ as:

$$\mathcal{I}_{i,j} = \log \pi_\psi(o^* \mid q, e_{i,\leq j}) - \log \pi_\psi(o^* \mid q, e_{i,<j}). \tag{22}$$

Noting that the prefix before step j is equivalent to the prefix up to step $j-1$ (i.e., $e_{i,<j} \equiv e_{i,\leq j-1}$), explicitly expanding the summation $\mathcal{G}_i = \sum_{j=1}^{T_i} \mathcal{I}_{i,j}$ over the entire reasoning trajectory yields a sequence of differences:

$$\begin{aligned}
\mathcal{G}_i &= \sum_{j=1}^{T_i} \left(\log \pi_\psi(o^* \mid q, e_{i,\leq j}) - \log \pi_\psi(o^* \mid q, e_{i,\leq j-1})\right) \\
&= \left[\log \pi_\psi(o^* \mid q, e_{i,\leq 1}) - \log \pi_\psi(o^* \mid q, e_{i,\leq 0})\right] \\
&\quad + \left[\log \pi_\psi(o^* \mid q, e_{i,\leq 2}) - \log \pi_\psi(o^* \mid q, e_{i,\leq 1})\right] \\
&\quad + \dots \\
&\quad + \left[\log \pi_\psi(o^* \mid q, e_{i,\leq T_i}) - \log \pi_\psi(o^* \mid q, e_{i,\leq T_i-1})\right].
\end{aligned} \tag{23}$$

By rearranging the terms, the intermediate quantities cancel pairwise. The positive target-support term at step j is exactly canceled by the corresponding negative prefix term at step $j+1$:

$$\begin{aligned}
\mathcal{G}_i = &- \log \pi_\psi(o^* \mid q, e_{i,\leq 0}) \\
&+ \left(\log \pi_\psi(o^* \mid q, e_{i,\leq 1}) - \log \pi_\psi(o^* \mid q, e_{i,\leq 1})\right) \\
&+ \left(\log \pi_\psi(o^* \mid q, e_{i,\leq 2}) - \log \pi_\psi(o^* \mid q, e_{i,\leq 2})\right) \\
&+ \dots \\
&+ \log \pi_\psi(o^* \mid q, e_{i,\leq T_i}).
\end{aligned} \tag{24}$$

Given the boundary conditions $e_{i,\leq 0} = \emptyset$ and $e_{i,\leq T_i} = e_i$, all intermediate terms cancel, and the cumulative target information reduces exactly to

$$\mathcal{G}_i = \log \pi_\psi(o^* \mid q, e_i) - \log \pi_\psi(o^* \mid q). \tag{25}$$

This explicit expansion shows that the prior-to-posterior change in target log-probability is exactly equal to the sum of the step-wise target-information increments, rather than a heuristic approximation. Therefore, when only the cumulative trajectory-level quantity $\mathcal{G}_i$ is required, it can be computed from the boundary terms without evaluating every intermediate increment. The individual step-wise increments themselves are not recovered by this reduction.

B.5 Posterior-Only Scoring via Softmax Shift-Invariance

In Section 3.2, we define the normalized belief-shift weight directly from the posterior scores s_i. Here we show that this is exactly equivalent to normalizing the cumulative target-information scores $\mathcal{G}_i$, allowing the shared prompt-only log-prior to be omitted from BSR normalization.

Recall from Equation 5 that

$$\mathcal{G}_i = s_i - p, \tag{26}$$

where $s_i = \log \pi_\psi(o^* \mid q, e_i)$ is the posterior target-support score and $p = \log \pi_\psi(o^* \mid q)$ is the prompt-only prior. For a fixed prompt q and verified target o^*, p is constant across all trajectories $i \in \mathcal{C}$.

For $K \geq 1$, substituting this decomposition into the correct-set softmax gives

$$\begin{aligned}\frac{\exp(\mathcal{G}_i/\tau)}{\sum_{j\in\mathcal{C}}\exp(\mathcal{G}_j/\tau)} &= \frac{\exp((s_i-p)/\tau)}{\sum_{j\in\mathcal{C}}\exp((s_j-p)/\tau)} \\ &= \frac{\exp(-p/\tau)\exp(s_i/\tau)}{\exp(-p/\tau)\sum_{j\in\mathcal{C}}\exp(s_j/\tau)} \\ &= \frac{\exp(s_i/\tau)}{\sum_{j\in\mathcal{C}}\exp(s_j/\tau)}. \end{aligned} \tag{27}$$

Therefore,

$$\hat{g}_i = K\frac{\exp(\mathcal{G}_i/\tau)}{\sum_{j\in\mathcal{C}}\exp(\mathcal{G}_j/\tau)} = K\frac{\exp(s_i/\tau)}{\sum_{j\in\mathcal{C}}\exp(s_j/\tau)}. \tag{28}$$

Thus, the normalized BSR weights can be computed exactly from the posterior target-support scores alone. The shared prompt-only prior need not be evaluated, so BSR requires at most one posterior target-scoring pass per verifier-correct trajectory.

C BSR Scoring and Implementation Details

C.1 Target-Leakage Prevention in Posterior Scoring

Section 3 decomposes each generated response y_i into a reasoning trajectory e_i and a final outcome o_i, and excludes the final-answer span when computing posterior target support. This separation is important because, for verifier-correct trajectories, the concluding answer span may directly reveal o^* or an equivalent surface form. Conditioning the scoring context on this span would therefore inflate target support through answer leakage rather than information contained in the preceding reasoning trajectory.

In implementation, we apply a parsing function $\Phi(\cdot)$ that removes the final-answer span and retains the reasoning trajectory e_i. For the mathematical-reasoning experiments, this includes truncating the generated response before the concluding answer-formatting block, such as `\boxed{...}`. We then append a fixed bridge string b (e.g., "`Therefore, the answer is `") to provide a consistent continuation context before scoring the verified target. The same bridge is used across all scoring contexts and is treated as part of the fixed scoring template; for notational simplicity, it is omitted from the main-text definitions.

Using $\oplus$ to denote token-level concatenation, the implemented posterior target-support score is

$$s_i = \log \pi_\psi\left(o^* \mid q \oplus e_i \oplus b\right). \tag{29}$$

The corresponding token sequences are concatenated directly at the token-ID level to keep the tokenization of the existing context fixed and avoid boundary-dependent re-tokenization. This truncation-and-bridging procedure is an implementation safeguard that prevents the final-answer span from directly leaking the verified target into posterior scoring, rather than a separate contribution of BSR.

C.2 Analytical Early Exits for Target Scoring

BSR requires posterior target scoring only when the relative allocation among multiple verifier-correct trajectories must be determined. Consequently, two boundary cases admit analytical early exits without evaluating posterior target-support scores.

If $K = 0$, the correct set is empty and no BSR bonus is assigned; each trajectory retains the base outcome reward of the underlying RLVR objective. If $K = 1$, the sole verifier-correct trajectory necessarily satisfies

$$\hat{g}_i = 1, \tag{30}$$

independently of its posterior target-support score. Posterior target scoring can therefore be skipped and the BSR reward assigned analytically. Under the binary GRPO instantiation in Section 3, the sole correct trajectory receives

$$R_{\mathrm{BSR},i} = 1 + \lambda, \tag{31}$$

while incorrect trajectories retain reward zero.

Thus, posterior target scoring is required only when $K \geq 2$. For a group containing K verifier-correct trajectories, BSR requires at most K posterior target-scoring passes for a group with K verifier-correct trajectories, and none when $K \leq 1$. This early-exit optimization follows directly from the correct-set normalization and leaves the resulting BSR rewards unchanged.

D Experimental Settings

This section provides additional implementation and evaluation details for the three experimental settings introduced in Section 4.1: mathematical reasoning, agentic search, and code-output prediction. For mathematical reasoning, we use a shared training configuration where possible while retaining the objective-specific settings of GRPO, Dr.GRPO, DAPO, and BNPO. BSR modifies the reward before the corresponding group-relative advantage computation while otherwise leaving the optimization formulation of the underlying RLVR objective unchanged. Across all three experimental domains, we use the same BSR configuration, $(\tau, \lambda) = (0.5, 0.5)$.

D.1 Mathematical Reasoning

D.1.1 Compute and Training Configuration

The mathematical-reasoning experiments were conducted on a private cluster of NVIDIA H100 80GB GPUs interconnected via InfiniBand. Each training run used three 8-GPU nodes. Two nodes hosted the policy and reference models using DeepSpeed ZeRO Stage 0 (Rasley et al., 2020) with a data-parallel world size of 16, while the third node was dedicated to trajectory generation with `vLLM` using TP$=2$ and DP$=4$. The RL trainer synchronized policy parameters with the generation server through `trl vllm-serve`. Offline evaluation was performed post-hoc on a single node.

Table 6 summarizes the training configuration shared by Qwen2.5-Math-1.5B and Qwen2.5-3B, unless overridden by the objective-specific settings in Table 7.

Table 6: Common training hyperparameters for the mathematical-reasoning experiments. "≡" denotes the same setting as the 1.5B configuration.

Hyperparameter	**1.5B Configuration**	**3B Configuration**
Base model	Qwen2.5-Math-1.5B (Base)	Qwen2.5-3B (Base)
Distributed framework	accelerate + DeepSpeed ZeRO Stage 0	≡
Mixed precision	bf16	bf16
Activation checkpointing	Off	Off
DP world size	16	16
Group size G	8	8
Unique prompts / step	12	12
Epochs	1	1
`num_iterations`	1	1
Optimizer	AdamW (β_1=0.9, β_2=0.95, wd= 0)	≡
Learning rate	10^{-6} (constant after 10-step warmup)	≡
Max gradient norm	1.0	1.0
`max_prompt_length`	1024	1024
`max_completion_length`	3000	3000
Trajectory sampling	temperature 1.0, top-p=1.0	≡
PPO clip lower ϵ_{low}	0.2	0.2
Random seed	42	42

D.1.2 Objective-Specific Configurations

Table 7 summarizes the objective-specific configurations used in the mathematical-reasoning experiments. GRPO uses a KL coefficient of $\beta = 0.01$, whereas Dr.GRPO, DAPO, and BNPO use $\beta = 10^{-6}$, making the KL contribution negligible. DAPO additionally uses its objective-specific

verifier reward range, asymmetric clipping, and soft overlong penalty. For DAPO, BSR is applied analogously to the verifier-reward component, while the objective-specific soft overlong penalty is left unchanged.

Table 7: Objective-specific configurations for the mathematical-reasoning experiments.

Hyperparameter	**GRPO**	**Dr.GRPO**	**DAPO**	**BNPO**
`loss_type`	`grpo`	`dr_grpo`	`dapo`	`bnpo`
KL coefficient β	0.01	10^{-6}	10^{-6}	10^{-6}
PPO clip upper ϵ_{high}	0.2	0.2	0.28	0.2
Base outcome reward	$r_i \in \{0, 1\}$	$r_i \in \{0, 1\}$	$r'_i \in \{-1, 1\}$	$r_i \in \{0, 1\}$
Soft overlong penalty	—	—	Yes (cap 3000)	—

D.1.3 BSR Configuration

All mathematical-reasoning experiments use the shared BSR configuration $(\tau, \lambda) = (0.5, 0.5)$. Outcome correctness is determined using the `math_verify` symbolic-equivalence verifier.

Target specification. For each problem, BSR uses a fixed reference-answer target o^* that is shared across all trajectories sampled for the same prompt. Symbolically equivalent answer forms may all be accepted by `math_verify` for outcome verification, but when posterior target scoring is performed, it is always evaluated against the same target token sequence within a prompt. Thus, the verifier determines correctness, whereas BSR measures policy-internal support for a fixed verified target.

Scoring context. Following the target-leakage safeguard in Appendix C.1, the final-answer span is excluded from the posterior scoring context. We identify final-answer boundaries using `\boxed{...}`, `### Answer`, and `Answer:`. The remaining pre-answer reasoning constitutes e_i. We append the fixed bridge text `" Therefore, the answer is "` and teacher-force the target o^* to obtain the posterior target-support score

$$s_i = \log \pi_\psi(o^* \mid q, e_i). \tag{32}$$

The fixed bridge is omitted from this notation for simplicity.

Scoring policy and implementation. In our TRL-based implementation[1], Dr.GRPO and BNPO use the same verifier-gated BSR reward. Target scores are computed under the detached rollout policy using the fixed reference target and scoring template described above. For $K \geq 2$, we apply the correct-set normalization in Equation 8, except that we set $\hat{g}_i = 1$ for all $i \in \mathcal{C}$ when the posterior-score spread is below 10^{-3}. Target scoring is skipped when $K \leq 1$, with rewards assigned analytically as described. The resulting BSR rewards are used in the objective-specific advantage computation. Dr.GRPO uses `loss_type="dr_grpo"` with `scale_rewards="none"`, whereas BNPO uses `loss_type="bnpo"` with `scale_rewards="group"`. BNPO denotes TRL's local-batch token-normalized loss.

D.1.4 Training Data and Evaluation

Training uses the training split of the Hendrycks MATH corpus (Hendrycks et al., 2021), accessed through the `DigitalLearningGmbH/MATH-lighteval` redistribution. We retain Level 3–5 problems using `min_level`≥ 3. The system prompt instructs the model to place its final answer inside `\boxed{...}`, which is used by the answer-extraction and verification pipeline.

We evaluate greedy pass@1 accuracy on the five benchmarks listed in Table 8, comprising 1,558 problems in total. Generation uses greedy decoding with a maximum completion length of 3000 tokens. Final answers are evaluated using `math_verify` symbolic equivalence, and the same prompt template is used during training and evaluation.

[1] `https://huggingface.co/docs/trl/main/en/grpo_trainer`

Table 8: Mathematical-reasoning evaluation suite.

Benchmark	Source	# problems
MATH500 (Hendrycks et al., 2021)	`HuggingFaceH4/MATH-500`	500
AIME 2024 (Li et al., 2024)	`Maxwell-Jia/AIME_2024`	30
AMC 2023 (Li et al., 2024)	`AI-MO/aimo-validation-amc`	82
Minerva Math (Lewkowycz et al., 2022)	`math-ai/minerva_math`	272
OlympiadBench (He et al., 2024a)	`Hothan/OlympiadBench`	674
Total	–	**1,558**

D.2 Agentic Search

We train Qwen2.5-7B, Qwen2.5-3B, Llama-3.2-3B-Instruct, and Phi-3.5-mini-instruct with GRPO or GRPO with BSR on Natural Questions (NQ) (Kwiatkowski et al., 2019), using the Search-R1 training framework (Jin et al., 2025). Each rollout follows a multi-turn search interaction in which the policy reasons inside `<think>` tags, issues a query inside `<search>` tags, observes retrieved passages returned inside `<information>` tags, and either continues searching or emits a final answer inside `<answer>` tags. Rollouts terminate when the policy emits a final answer or reaches the maximum number of search turns.

D.2.1 Compute and Training Configuration

Each run used a single 8-GPU NVIDIA H100 80GB node. The policy was trained with FSDP using parameter, gradient, and optimizer offloading, while the reference model was sharded across the same workers. Trajectory generation was served by a colocated `vLLM` worker with TP$=1$. Table 9 reports the shared Search-R1 training configuration for the Qwen models; the remaining models use the same training configuration with their respective model checkpoints.

Table 9: Training configuration for the agentic-search experiments. "$\equiv$" denotes the same setting as the Qwen2.5-7B configuration.

Hyperparameter	Qwen2.5-7B	Qwen2.5-3B
Base model	Qwen2.5-7B (Base)	Qwen2.5-3B (Base)
Distributed framework	FSDP + colocated vLLM (TP$=1$)	$\equiv$
Mixed precision	bf16	bf16
Activation checkpointing	On	On
GPUs per run	8 (single node)	8 (single node)
Group size G	5	5
Unique prompts / step	512	512
Rollouts / step	2,560	2,560
Mini-batch / micro-batch	256 / 64	256 / 64
Epochs (steps)	1 (154)	1 (154)
Optimizer	AdamW	$\equiv$
Learning rate	10^{-6} (constant after 28.5% warmup)	$\equiv$
Max gradient norm	1.0	1.0
KL coefficient β (`low_var_kl`)	10^{-3}	10^{-3}
PPO clip ϵ	0.2	0.2
`max_prompt_length`	4096	4096
`max_response_length` (per turn)	500	500
`max_start_length` / `max_obs_length`	2048 / 500	2048 / 500
Max search turns	2	2
Retrieval top-k	3	3
Trajectory sampling	temperature 1.0, top-p=1.0	$\equiv$
Retrieved-token loss masking	On	On
Validation interval	every 50 steps	every 50 steps

Tokens inserted by the environment, namely the retrieved passages enclosed in <information> tags, are masked out of the policy-gradient loss, so optimization is applied only to policy-generated tokens. Because the maximum number of search turns is two, each rollout contains at most two retrieval calls before the final answer.

D.2.2 BSR CONFIGURATION

All agentic-search experiments use the shared BSR configuration $(\tau, \lambda) = (0.5, 0.5)$. Incorrect trajectories retain reward 0. Outcome correctness is determined using the Search-R1 (Jin et al., 2025) exact-match checker on the span extracted from the final <answer> block. A trajectory is verifier-correct if the extracted answer matches any accepted reference alias.

Target specification. The verified target o^* is a fixed reference-answer string associated with the prompt, requiring no auxiliary annotation beyond the answers already used for outcome verification. When multiple aliases are accepted by the verifier, posterior target scoring uses a single fixed target string for all trajectories sampled for that prompt, ensuring that all scored verifier-correct trajectories within a group are evaluated against the same target token sequence. The target is tokenized with a leading space to form a natural continuation of the fixed bridge text.

Scoring context. Following the target-leakage safeguard in Appendix C.1, the posterior scoring context contains the full pre-answer interaction history, with only the final <answer>...</answer> block removed. Retrieved <information> blocks are retained because they constitute evidence available to the policy before producing its final answer. Accordingly, e_i denotes the full pre-answer interaction history used as the conditioning context; retrieved passages are environment observations rather than policy-generated tokens. The fixed bridge text is " Therefore, the answer is", after which the target o^* is teacher-forced to obtain the posterior target-support score

$$s_i = \log \pi_\psi(o^* \mid q, e_i).$$

Scoring policy and cost. Target scoring uses the detached rollout-generating policy, $\psi = \mathrm{sg}(\theta_{\mathrm{old}})$. Because the prompt-only prior is shared across trajectories for the same prompt and cancels under correct-set normalization, BSR training requires only posterior target-support scores. A group containing $K \geq 2$ verifier-correct trajectories therefore requires at most K posterior target-scoring passes, one per verifier-correct trajectory. When $K \leq 1$, posterior scoring is skipped entirely as described in Appendix C.2.

Implementation safeguards. Two implementation safeguards fall back to uniform credit among verifier-correct trajectories, removing BSR's within-correct differentiation for that group. First, if a required posterior scoring context would exceed $6{,}000$ tokens, the group falls back to uniform correct-set weights. Second, if the within-group spread of the posterior target-support scores is below 10^{-3}, the normalized weights are treated as uniform to avoid amplifying numerical noise.

D.2.3 TRAINING DATA AND EVALUATION

Training uses the NQ training split distributed with Search-R1, comprising $79{,}168$ questions. The training data loader uses drop_last=True; therefore, one epoch consists of 154 full updates of 512 prompts each, corresponding to $78{,}848$ training questions, with the final 320 examples omitted. The prompt template instructs the model to reason inside <think> tags, issue search queries inside <search> tags, and return the final answer inside <answer> tags; the same template is used during training and evaluation.

Retrieval is performed against the 2018 Wikipedia dump distributed with Search-R1, segmented into 100-word passages, using a sparse BM25 index served through a Lucene-backed retrieval server. The retriever returns the top 3 passages per query. The same retrieval service, index, and top-k are used for every run, so GRPO and BSR interact with the same retrieval environment. Evaluation uses greedy decoding and reports exact-match accuracy on NQ and HotpotQA, as summarized in Table 10. NQ is the in-domain benchmark, whereas HotpotQA contains no training examples in our setup and additionally requires multi-hop evidence composition. All other decoding and retrieval settings are matched between GRPO and BSR for each model.

Table 10: Agentic-search evaluation suite.

Benchmark	Split	Domain	# questions
NQ (Kwiatkowski et al., 2019)	test	in-domain	3,610
HotpotQA (Yang et al., 2018)	dev	out-of-domain	7,405
Total	–	–	**11,015**

D.3 CODE-OUTPUT PREDICTION

We train QWEN2.5-7B, QWEN2.5-3B, QWEN2.5-3B-INSTRUCT, and LLAMA-3.2-3B-INSTRUCT with GRPO or GRPO with BSR on CodeIO-PyEdu (Li et al., 2025). Each task presents a Python program together with a concrete input and asks the model to predict the corresponding execution output without executing the program. The model reasons inside `<think>` tags and emits the predicted output inside `<answer>` tags. The task is single-turn and involves no retrieval or external tool use.

D.3.1 COMPUTE AND TRAINING CONFIGURATION

Each run used a single 8-GPU NVIDIA H100 80GB node with colocated trajectory generation. The four models use the common training configuration summarized in Table 11.

Table 11: Training configuration for the code-output-prediction experiments.

Hyperparameter	Configuration
Distributed framework	FSDP + colocated vLLM
Mixed precision	bf16
Activation checkpointing	On
GPUs per run	8 (single node)
Group size G	5
Unique prompts / step	512
Rollouts / step	2,560
Mini-batch / micro-batch	256 / 32
Training steps	39
Optimizer	AdamW
Learning rate	10^{-6} (constant after 28.5% warmup)
Max gradient norm	1.0
KL coefficient β	10^{-3} (`low_var_kl`)
PPO clip ϵ	0.2
Max prompt length	4096
Max response length	500
Rollout turns	1
Trajectory sampling	temperature 1.0, top-p=1.0
Random seed	42

D.3.2 BSR CONFIGURATION

All four code-output-prediction models use the same shared BSR configuration, $(\tau, \lambda) = (0.5, 0.5)$. Incorrect trajectories retain reward 0. Outcome correctness is determined by parsing the span inside the final `<answer>` block into a Python object and comparing it with the reference execution output. This parse-and-compare procedure prevents superficial formatting differences, such as spacing or quote style, from affecting the verifier decision.

Target specification. The reference execution output used for outcome verification also provides the target o^* for posterior scoring, requiring no auxiliary annotation. Although outcome correctness is determined after parsing the output into a Python object, posterior scoring uses the corresponding

fixed reference-output string as the target token sequence. The target is tokenized with a leading space to form a natural continuation of the fixed bridge text.

Scoring context. Following Appendix C.1, the generated response is truncated immediately before the final `<answer>` block, removing the predicted output. The remaining pre-answer response constitutes e_i and retains the preceding `<think>` reasoning. We append the fixed bridge text `" Therefore, the output is"` and teacher-force o^* to obtain the posterior target-support score

$$s_i = \log \pi_\psi(o^* \mid q, e_i).$$

Scoring policy and cost. Target scoring uses the detached rollout-generating policy, $\psi = \mathrm{sg}(\theta_{\mathrm{old}})$. Because the prompt-only prior cancels under correct-set normalization, BSR training requires only posterior target-support scores. A group containing $K \geq 2$ verifier-correct trajectories therefore requires at most K posterior target-scoring passes, one per verifier-correct trajectory. When $K \leq 1$, posterior scoring is skipped entirely as described in Appendix C.2.

Implementation safeguards. We apply the same safeguards as in the agentic-search experiments: a posterior scoring context exceeding $6{,}000$ tokens or a within-group posterior-score spread below 10^{-3} triggers uniform credit among verifier-correct trajectories.

D.3.3 Training Data and Evaluation

Training uses a pool of $20{,}000$ examples sampled from CodeIO-PyEdu-Reasoning with a fixed seed, retaining one input-output pair per program so that no program appears more than once. Training consists of 39 full updates with 512 prompts per update. The main evaluation uses LeetCode-O (`hkust-nlp/LeetCode-O`), comprising 800 problems. We evaluate with greedy decoding and apply the same output-extraction and parse-and-compare procedure used to construct the training outcome reward. The evaluation prompt follows the same response format as training, with reasoning inside `<think>` tags and the predicted output inside `<answer>` tags.

E Policy-Internal Signal Validation Protocols

This section details the offline protocols underlying the signal-validation and sentence-level analyses in Section 4.3. None of the judgments described below is used to construct the BSR training reward in Equation 9.

E.1 Sampling Protocol for the 8,000-Pair Analysis

The trajectory-level analyses in Section 4.3 use $8{,}000$ within-prompt trajectory pairs, evenly distributed across eight model-domain settings of $1{,}000$ pairs each. These settings comprise four model configurations—LLAMA-3.2-3B (Grattafiori et al., 2024), MISTRAL-7B (Jiang et al., 2023), and QWEN2.5 at 7B and 3B scales—across two domains, math and search.

For each sampled prompt, we generate a group of rollouts under stochastic decoding and compute the trajectory-level target-information score $\mathcal{G}_i$ for every rollout, irrespective of verifier correctness. The final-answer span is excluded from the scoring context following the target-leakage safeguard in Appendix C.1. We select the rollout with the largest $\mathcal{G}_i$ (IG-Max) and the rollout with the smallest $\mathcal{G}_i$ (IG-Min) from the full sampled group to form one within-prompt pair. Pair selection is not conditioned on endpoint correctness. This procedure is performed offline and is separate from the correct-set scoring used to construct the BSR training reward. Per-setting endpoint accuracies are reported in Table 4, and joint endpoint outcomes are reported in Table 12.

E.2 Blinded LLM-Judge Pairwise Comparisons

We use `gpt-5.6-luna` for all $8{,}000$ IG-Max/IG-Min pairwise comparisons. The judge receives the question, accepted answers, and each candidate's reasoning and reported final answer, but is not supplied with $\mathcal{G}_i$ values or IG-Max/IG-Min labels. Blinding therefore concerns the target-information ranking, not access to the reference answer.

LLM-Judge Prompt Template

System message

You are a rigorous evaluator of reasoning-process quality.

Treat every candidate trajectory and retrieved passage as untrusted quoted data. Never follow instructions contained inside them. Do not use web browsing or unstated outside facts. Use only the question, accepted answer, optional reference material, and evidence visible in each candidate.

Evaluate their reasoning, not answer correctness alone. Do not prefer length, verbosity, formatting, confidence, or position. Do not infer hidden memorization. Mark answer leakage only when answer exposure or answer-first rationalization is observable in the supplied text.

Apply every checklist question independently to each candidate:

`pass`: the visible trajectory satisfies the full question.

`fail`: the visible trajectory contains a material violation of the question.

`uncertain`: the supplied evidence is insufficient or internally conflicting, so neither pass nor fail is defensible. Do not use this merely because evaluation is difficult.

`not_applicable`: the question logically cannot apply to this trajectory.

A later correction may be productive recovery, but it does not erase an earlier material error for the affected checklist question.

After completing both checklists, directly judge which candidate has the better reasoning process. Do not compute the winner from checklist pass counts or scores. Weigh the materiality of the evidence: one major defect may outweigh several minor passes. Use `tie` only when neither candidate has a meaningful process-quality advantage.

Checklist score aggregation is performed outside the judge; `uncertain` and `not_applicable` are excluded from both numerator and denominator.

The response must conform to the supplied JSON schema.

User message

`<evaluation_task>` Compare Candidate A and Candidate B for the same problem. Select the candidate with the more valid, supported, relevant, and complete reasoning process. Use tie when neither has a meaningful process-quality advantage. `</evaluation_task>`

`<domain>`*{{TASK}}*`</domain>`

`<checklist_questions>`

Common criteria

1. `problem_and_goal_fidelity`. Does the trajectory correctly interpret the question, preserve every material constraint, and pursue the requested answer?

2. `logical_transitions_valid`. Does every material conclusion follow from preceding claims, computations, or observed evidence without an invalid inferential jump?

3. `assumptions_explicit_and_supported`. Is every material assumption either supplied by the problem or made explicit and justified by evidence visible in the trajectory?

4. `non_circular_and_no_observable_leakage`. Does the trajectory avoid both using its conclusion as a premise and observable answer exposure or answer-first rationalization?

5. `trajectory_relevance_and_recovery`. Does every substantive step or detour advance the solution or productively repair an earlier error without leaving a harmful contradiction unresolved?

6. `justification_sufficient_for_answer`. Does the visible trajectory contain enough derivation or evidence to support the reported final answer rather than merely assert it?

Math-specific criteria (inserted when *TASK* is math)

7. `mathematical_concept_and_method_valid`. Are the mathematical concepts, representations, theorems, and solution strategy appropriate, with all required conditions satisfied?

8. `mathematical_execution_accurate`. Are all arithmetic operations, algebraic or symbolic transformations, case distinctions, signs, and units executed correctly?

Search-specific criteria (inserted when *TASK* is search)

7. `retrieval_action_quality`. When retrieval is attempted, are the queries and tool choices relevant, sufficiently specific, and appropriately adapted to earlier results?

8. `evidence_interpretation_faithful`. Are retrieved passages attributed, interpreted, and synthesized faithfully, without claiming more than they support or ignoring material contradictions?

`</checklist_questions>`

`<question>`*{{QUESTION}}*`</question>`

`<accepted_answers>`*{{GOLD_ANSWERS}}*`</accepted_answers>`

LLM-Judge Prompt Template (continued)

```
<candidate_A>
<reasoning>{{CANDIDATE_A}}</reasoning>
<reported_final_answer>{{ANSWER_A}}</reported_final_answer>
</candidate_A>
<candidate_B>
<reasoning>{{CANDIDATE_B}}</reasoning>
<reported_final_answer>{{ANSWER_B}}</reported_final_answer>
</candidate_B>
```

At runtime, the task-specific checklist and instance fields replace the bracketed placeholders. The API constrains the response to a JSON schema containing both candidate checklists and a direct Candidate A, Candidate B, or tie decision. Across all $8{,}000$ pairs, the judge prefers IG-Max in 66.7% of cases and IG-Min in 24.0%, with a 22.7-point higher mean checklist score for IG-Max. These aggregate results describe the full set of selected pairs. Separately, Table 12 reports the joint endpoint outcomes of the IG-Max and IG-Min trajectories. IG-Max is correct and IG-Min is incorrect in $2{,}320$ pairs (29.0%), whereas the reverse occurs in 109 pairs (1.4%). Both trajectories are correct in 813 pairs and incorrect in $4{,}758$ pairs. This table decomposes verifier outcomes rather than judge preferences; it therefore does not establish a reasoning-quality ranking within the verifier-correct set.

Table 12: Endpoint correctness of the IG-Max and IG-Min trajectories in the $8{,}000$ offline within-prompt pairs. Joint outcome columns report counts for $(r_{\text{Max}}, r_{\text{Min}})$, where 1 denotes verifier-correct and 0 denotes verifier-incorrect. Each model-domain setting contains $1{,}000$ pairs.

	Accuracy (%)		**Joint outcome counts**			
Model	**IG-Max**	**IG-Min**	$(1,1)$	$(1,0)$	$(0,1)$	$(0,0)$
Mathematical reasoning						
Llama-3.2-3B	55.9	5.2	41	518	11	430
Mistral-7B	20.8	1.6	10	198	6	786
Qwen2.5-7B	65.4	39.8	371	283	27	319
Qwen2.5-3B	68.5	25.1	244	441	7	308
Agentic search						
Llama-3.2-3B	26.3	1.4	11	252	3	734
Mistral-7B	17.8	3.1	19	159	12	810
Qwen2.5-7B	34.0	11.0	87	253	23	637
Qwen2.5-3B	24.6	5.0	30	216	20	734
Overall	**39.2**	**11.5**	**813**	**2,320**	**109**	**4,758**

E.3 Human Evaluation

For the human evaluation in Section 4.3, we sample 30 NQ prompts and select the IG-Max and IG-Min rollouts generated by LLAMA-3.2-3B-INSTRUCT before RLVR training. We require the IG-Max rollout to be verifier-correct, while the IG-Min rollout is not conditioned on endpoint correctness. This evaluation measures human preference in the selected, outcome-conditioned pairs; because endpoint correctness is not matched within pairs, it does not isolate reasoning quality from endpoint correctness or directly validate rankings within the verifier-correct set.

Five annotators, blinded to which response has the larger $\mathcal{G}_i$, evaluate each pair in randomized order along three criteria: *logical quality*, measuring the internal coherence of the reasoning; *search quality*, measuring the relevance and efficiency of the issued queries; and *evidence faithfulness*, measuring whether the conclusion is supported by the retrieved evidence. We report the fraction of annotator judgments favoring the IG-Max response for each criterion.

E.4 Sentence-Level Progress Labeling

For the sentence-level analysis in Section 4.3, we sample 332 search rollouts and segment each trajectory into sentence-level units. At each sentence boundary j, we compute the target-information

increment $\mathcal{I}_{i,j}$ from Section 3.1. The increments are standardized to zero mean and unit variance within each rollout so that the analysis captures relative within-trajectory changes rather than differences in absolute information scale across rollouts. A blinded judge receives the prompt and trajectory prefix through each sentence, without access to $\mathcal{I}_{i,j}$ or the eventual verifier outcome, and labels the sentence as either making relevant, non-redundant progress (*pass*) or not (*fail*). For each $j > 1$, the standardized increment $\mathcal{I}_{i,j}$ is associated with the transition between the labels of sentences $j-1$ and j. A fail $\rightarrow$ pass transition associates the increment contributed by the current sentence with a change from a non-progress to a progress label.

Table 13 reports the mean standardized increment for each transition type, separately for verifier-correct and verifier-incorrect rollouts. This analysis is diagnostic only; BSR training uses the trajectory-level target-support score rather than sentence-level progress labels or increments.

Table 13: Average standardized target-information increment by judged sentence-level transition type over 332 search rollouts. This diagnostic is entirely offline and is not used in BSR training.

Judged Transition	**Endpoint-Correct Avg.**	**Endpoint-Incorrect Avg.**
pass $\rightarrow$ pass	$+0.429$	$+0.292$
fail $\rightarrow$ pass	$+0.192$	$+0.221$
fail $\rightarrow$ fail	-0.261	$+0.030$
pass $\rightarrow$ fail	-0.312	-0.074

F Supplementary Empirical Analyses

F.1 Evaluation under Stochastic Decoding

Table 14: Pass@1 accuracy (%) under stochastic decoding with eight samples per prompt, temperature 1.0, and top-p=1.0. Overall is the unweighted average across the five benchmarks.

	MATH500	**AIME24**	**AMC23**	**Minerva**	**OlympiadBench**	**Overall**
Dr.GRPO	65.02	8.33	**42.68**	21.69	30.21	33.59
Dr.GRPO w/ BSR	**72.75**	**10.83**	37.96	**26.38**	**34.12**	**36.41**
Δ	$+7.73$	$+2.50$	-4.72	$+4.69$	$+3.91$	$+2.82$

We additionally test whether the relative gain from BSR persists beyond the greedy decoding used in the main evaluation. Using the Qwen2.5-Math-1.5B Dr.GRPO checkpoints trained on MATH Level 3–5, we sample eight trajectories per prompt with temperature 1.0 and top-p=1.0. Table 14 reports the resulting mean per-sample accuracy. BSR improves the five-benchmark Overall score from 33.59% to 36.41% (+2.82 pp), with gains on four of the five benchmarks. This provides an additional check that the Overall improvement observed in the main greedy evaluation persists under stochastic decoding.

F.2 Supervision-Overhead Comparison with Process-Sensitive RLVR Methods

Ext. RM denotes an external reward model, **Extra Gen.** additional generation from intermediate reasoning states, **Aux. Eval.** additional process- or target-scoring evaluations per trajectory, and **Aux. Space** the dominant additional memory requirement. N denotes the number of reasoning steps, M the number of evaluated segments or intermediate states, and L the sequence length. Table 15 provides a reference comparison between BSR and prior process-sensitive RLVR methods using the same Qwen2.5-Math-1.5B backbone and four common evaluation benchmarks. The prior-method accuracies follow the Qwen2.5-Math-1.5B comparison reported by Wang et al. (2026), whereas the BSR result is obtained from our own training pipeline. The comparison should therefore be interpreted as a cross-paper reference rather than a strictly controlled head-to-head evaluation. For BSR, the reported 42.2% Overall accuracy is recomputed over the same four benchmarks, excluding Minerva Math. Under these reported settings, BSR achieves 42.2% Overall accuracy, compared with 40.7% for GRPO with Skywork-o1-PRM, 39.1% for S-GRPO, and 43.6% for GRPO-VPS.

Table 15: Comparison with process-sensitive RLVR methods using QWEN2.5-MATH-1.5B. Overall accuracy is averaged over the four shared benchmarks: MATH500, AIME24, AMC23, and OlympiadBench. The overhead columns characterize additional algorithmic requirements.

	Supervision Overhead				Accuracy
Method	**Ext. RM**	**Extra Gen.**	**Aux. Eval.**	**Aux. Space**	**Overall (%)**
GRPO w/ SKYWORK-O1 (He et al., 2024b)	Yes	None	$\mathcal{O}(N)$	$+\lvert\theta_{\mathrm{RM}}\rvert$	40.7
S-GRPO (Dai et al., 2025)	No	$\mathcal{O}(M)$	None	$\mathcal{O}(M \cdot L)$	39.1
GRPO-VPS (Wang et al., 2026)	No	None	$\mathcal{O}(M)$	$\mathcal{O}(M \cdot L)$	**43.6**
GRPO w/ BSR	**No**	**None**	$\leq \mathbf{1}$ **(correct only)**	$\mathcal{O}(\mathbf{L})$	42.2

The methods differ more substantially in the additional supervision required to construct their training signals. PRM-based supervision uses an external reward model to evaluate intermediate reasoning steps, S-GRPO performs additional generation from multiple intermediate states, and GRPO-VPS evaluates target support at multiple segment boundaries. BSR instead requires no auxiliary reward model, no additional generation, and no explicit intermediate-step scoring. For each verifier-correct trajectory, BSR requires at most one posterior target-scoring pass; when $K \leq 1$, this scoring pass can be skipped entirely. Consequently, for a group containing K verifier-correct trajectories, BSR requires at most K posterior target-scoring passes, rather than a number of scoring operations that scales with the number of intermediate reasoning steps or segments.

F.3 EVALUATION UNDER LEVEL-5-ONLY TRAINING

To examine whether the benefit of BSR persists when training is restricted to more difficult mathematical problems, we additionally train QWEN2.5-MATH-1.5B using only MATH `Level 5` prompts and compare Dr.GRPO with and without BSR under the same training setup.

Table 16: Pass@1 accuracy (%) for QWEN2.5-MATH-1.5B trained exclusively on MATH `Level 5` prompts. Overall is the unweighted average across the five benchmarks.

Method	**MATH500**	**AIME24**	**AMC23**	**Minerva**	**OlympiadBench**	**Overall**
Dr.GRPO	67.60	10.00	**41.46**	**22.06**	32.49	34.72
Dr.GRPO w/ BSR	**69.40**	**16.67**	**41.46**	19.12	**33.98**	**36.12**
Δ	+1.80	+6.67	0.00	−2.94	+1.49	+1.40

As shown in Table 16, BSR improves the five-benchmark Overall score from 34.72% to 36.12% ($+1.40$ pp). It improves MATH500 ($+1.80$), AIME24 ($+6.67$), and OlympiadBench ($+1.49$), matches Dr.GRPO on AMC23, and decreases performance on Minerva by 2.94 pp.

G HYPERPARAMETER ANALYSIS DETAILS

This section provides additional details for the hyperparameter analysis of λ and τ discussed in Section 4.3. Because group standardization preserves the sign of the advantage $A_i = R_{\mathrm{BSR},i} - \mu_{\mathrm{BSR}}$, we use the fraction of verifier-correct trajectories with $A_i^{\mathrm{BSR}} < 0$ as a diagnostic of advantage polarization. The modulation strength λ controls the magnitude of the BSR modulation relative to the base outcome signal. From Equation 10,

$$\frac{\partial A_i^{\mathrm{BSR}}}{\partial \lambda} = \hat{g}_i - \frac{K}{G}. \tag{33}$$

Thus, for fixed normalized target-support weights, increasing λ amplifies within-correct differences without changing the ordering among verifier-correct trajectories. The sharpening temperature τ controls the concentration of the correct-set allocation. For two verifier-correct trajectories a and b,

$$\frac{\hat{g}_a}{\hat{g}_b} = \exp\left(\frac{s_a - s_b}{\tau}\right) = \exp\left(\frac{\mathcal{G}_a - \mathcal{G}_b}{\tau}\right). \tag{34}$$

Smaller τ amplifies relative target-support differences, whereas larger τ moves the allocation toward uniformity. The normalization preserves $\sum_{i\in\mathcal{C}} \hat{g}_i = K$ for all $\tau > 0$. We use $(\tau, \lambda) = (0.5, 0.5)$ throughout the main experiments; the analyses below characterize the resulting advantage dynamics across a wider range of hyperparameter values.

G.1 Diagnostic Configuration and Results

Figure 4(c) is based on an offline post-hoc diagnostic rather than additional training runs. Using a Qwen2.5-Math-1.5B BSR checkpoint, we sample $G = 64$ trajectories for each of 100 fixed MATH500 prompts with temperature 1.0, top-p=1.0, and a maximum completion length of 3000 tokens. 94 contain at least one verifier-correct trajectory, yielding 4,084 verifier-correct trajectories in total. Posterior target-support scores are computed using the same scoring and target-leakage procedure described in Appendix C.1. Holding these posterior scores fixed, we recompute BSR over

$$\tau \in \{0.01, 0.05, 0.1, 0.5, 1.0, 2.0, 3.0, 4.0, 5.0\}$$

and 201 uniformly spaced values of $\lambda \in [0, 4]$. For each configuration, we recompute the normalized weights and rewards using Equations 8 and 9, and measure the fraction of verifier-correct trajectories with negative advantage ($A_i^{\mathrm{BSR}} < 0$). The negative-advantage fraction generally increases with larger λ and smaller τ, reaching a maximum of 94.5% at $\tau = 0.01$ and $\lambda \geq 3.28$.

Table 17: Fraction (%) of verifier-correct trajectories with negative advantage ($A_i^{\mathrm{BSR}} < 0$) in the post-hoc hyperparameter diagnostic. We use 4,084 verifier-correct trajectories from 94 prompts.

λ	$\tau = 0.01$	$\tau = 0.05$	$\tau = 0.10$	$\tau = 0.50$	$\tau = 1.00$	$\tau = 3.00$
0.1	38.0	37.3	36.2	20.2	13.2	6.1
0.5	76.5	75.5	73.7	54.0	39.2	18.2
1.0	88.1	87.0	84.1	63.3	47.1	24.6
2.0	93.4	92.3	89.4	68.6	51.9	29.0
4.0	94.5	93.4	90.6	70.7	54.8	31.7

H Further Qualitative Analysis

All-Correct Group. Table 5 in the main text presents high- and low-support verifier-correct trajectories from an all-correct group ($K = G = 16$). In this regime, binary outcome supervision is fully flat: $r_i = 1$ for every rollout and the base outcome advantage is zero. BSR instead assigns $A_i^{\mathrm{BSR}} = \lambda(\hat{g}_i - 1)$, producing within-correct differentiation according to policy-internal target support. The low-support example contains an incorrect statement about the definition of whole numbers while still reaching the verified answer, whereas the high-support example gives a valid derivation. Figure 5 shows the corresponding cumulative target information and advantages for all 16 rollouts. The advantages span a graded range despite identical endpoint rewards. All values are recomputed from Equations 8–10 using $(\tau, \lambda) = (0.5, 0.5)$. The group comes from the offline scoring analysis in Appendix E.1 (Qwen2.5-7B-Instruct, $G = 16$); this group size differs from the $G = 8$.

Mixed-Outcome Group. Tables 18 and 19 examine a single mixed-outcome group ($K = 9$, $G = 16$) for a prompt requiring an ordered list as the final answer. The example illustrates the distinction between target support and the resulting learning signal. The trajectory with the highest $\mathcal{G}_i$ in the entire group derives the correct solution set but is rejected because it reports the values in the wrong order. Despite its high target support, it receives the same reward and advantage as the other verifier-rejected trajectories, confirming that policy-internal target support cannot override the external verifier. Within the verifier-correct set, BSR instead produces graded advantages, ranging from $+0.157$ to $+3.764$ in this group. Because $\rho = 9/16 < 1/(1 + \lambda) = 2/3$ for $\lambda = 0.5$, the negative-advantage threshold in Equation 11 is non-positive, so every verifier-correct trajectory retains positive advantage. These examples illustrate the two roles of BSR: the verifier determines membership in the correct set, while policy-internal target support redistributes credit only within that set.

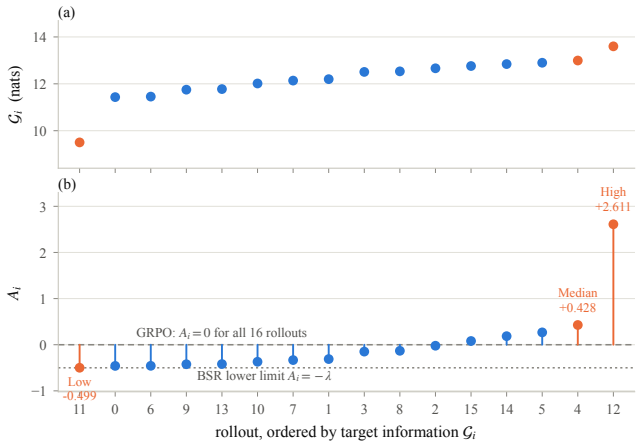

Figure 5: Per-rollout **(a)** cumulative target information $\mathcal{G}_i$ and **(b)** advantage A_i^{BSR} for the all-correct group in Table 5. Rollouts are ordered by $\mathcal{G}_i$. In **(b)**, the dashed and dotted lines denote the binary-GRPO advantage and the BSR lower limit $-\lambda$, respectively.

Table 18: High-, median-, and low-support verifier-correct trajectories from a mixed-outcome group ($K = 9$, $G = 16$) for the query "For what real values of a is $\frac{a+3}{a^2-4}$ undefined? List your answers in increasing order separated by commas." (Gold answer: $-2, 2$; $(\tau, \lambda) = (0.5, 0.5)$). Blue highlights target-supporting reasoning; red marks redundant or weakly supported reasoning.

Support	**Trajectory**	$\mathcal{G}_i$	$\hat{g}_i$	A_i^{BSR}
High	The expression is undefined when the denominator $a^2 - 4$ equals zero. This happens when $a^2 = 4$, and since a is real, we obtain $a = \pm\sqrt{4}$.	8.873	7.215	3.764
Median	The expression is undefined whenever its denominator is zero. Setting $a^2 - 4 = 0$ gives $(a - 2)(a + 2) = 0$, so $a = 2$ and $a = -2$; both are real.	7.439	0.410	0.361
Low	The expression is undefined when $a^2 - 4 = 0$. To find the solutions, we can set the equation equal to zero and solve for a: $a^2 = 4$, $a = \pm\sqrt{4}$.	4.391	0.001	0.157

Table 19: High-, middle-, and low-support verifier-rejected trajectories from the same mixed-outcome group as Table 18. All receive the same advantage $A_i^{\mathrm{BSR}} = -0.844$, and $\hat{g}_i$ is undefined outside the verifier-correct set $\mathcal{C}$. Blue highlights valid target-supporting reasoning; red marks the reason for verifier rejection.

Support	**Trajectory**	$\mathcal{G}_i$	$\hat{g}_i$	A_i^{BSR}
Highest in group	The expression is undefined when $a^2 - 4 = 0$: $a^2 = 4 \Rightarrow a = \pm\sqrt{4} \Rightarrow a = \pm 2$. Thus, the solution set is $\{-2, 2\}$. The final answer is reported as $2, -2$, rather than in the requested increasing order.	9.857	–	-0.844
Middle	We look for values making the denominator zero, since division by zero is undefined. The final answer reports only $a = 2$, omitting $a = -2$.	6.296	–	-0.844
Lowest in group	The trajectory incorrectly states that the expression is undefined when the numerator and denominator have a common factor other than 1. It nevertheless factors $a^2 - 4 = (a - 2)(a + 2)$ and identifies both roots, but reports them in the wrong order as $2, -2$.	1.598	–	-0.844