Reducing Pre-Normalization Centered-Advantage Variance Through Counterfactual Reasoning Paths

Fei Ding
Nanchang University
China

Yongkang Zhang
Peking University
China

Youwei Wang
Tsinghua University
China

Zijian Zeng
Tsinghua University
China

Abstract

Reinforcement learning for multi-step reasoning with large language models (LLMs) typically relies on sparse terminal rewards, which can create a poorly conditioned credit-assignment problem: the final feedback is propagated uniformly across all intermediate decisions. This can lead to high gradient variance, unstable training, and many ineffective updates, potentially limiting sustained model improvement. We propose a counterfactual-comparison framework for credit assignment. For each input, the framework samples multiple reasoning trajectories and treats their differences as implicit approximations to alternative decisions. This yields an implicit process-level advantage estimator that converts sparse terminal rewards into step-sensitive learning signals. Building on this framework, we introduce Implicit Behavior Policy Optimization (IBPO), which improves performance on mathematical and code-reasoning benchmarks. Our results point to a promising direction for unlocking the reasoning potential of LLMs.

1 Introduction

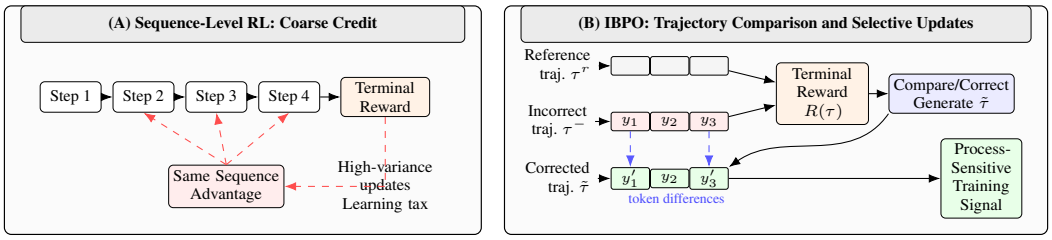

Figure 1: Overview of IBPO. Standard sequence-level RL propagates sparse terminal rewards coarsely to the whole trajectory, which can produce unstable updates. IBPO first generates an incorrect trajectory and a reference trajectory and obtains terminal feedback from a verifier; it then generates a corrected third trajectory through comparison/correction. The training signal is induced by the terminal reward and the token-level differences between the incorrect and corrected trajectories, converting sparse terminal feedback into process-sensitive training signals with optional structural token selection.

Recent advances in large language models (LLMs) have produced substantial progress on complex multi-step reasoning tasks, especially after reinforcement-learning (RL) fine-tuning. RL has become a central paradigm for scaling LLM capabilities, enabling models to solve increasingly difficult problems, such as competition-level mathematics and program synthesis, through deeper and longer reasoning chains.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

However, scaling RL for reasoning requires training to remain stable and sample-efficient under growing compute budgets. Despite this need, mainstream RL methods, such as Group Relative Policy Optimization (GRPO) Shao et al. [2024], still optimize policies with sequence-level or trajectory-level rewards. This creates a fundamental mismatch between the learning signal and the inherently stepwise nature of reasoning.

In multi-step reasoning, correctness depends on a sequence of intermediate decisions. Sequence-level supervision, however, rewards an entire trajectory only according to the final answer: a trajectory with flawed reasoning can still receive a positive reward if it happens to end with the correct answer, whereas a largely correct trajectory with a single local mistake may be discarded wholesale. This paper focuses primarily on the latter class of failed but recoverable trajectories. They contain useful intermediate reasoning fragments, yet under sequence-level rewards they are treated entirely as negative samples because their final answers are wrong. Such coarse feedback weakens the model's ability to distinguish early from late mistakes, disrupts credit assignment, destabilizes learning, and limits exploration of alternative reasoning paths. The problem is especially pronounced in long-horizon or difficult tasks. Moreover, even a single local error may require extensive sampling and many updates before it is corrected statistically, creating a substantial efficiency bottleneck often referred to as a *learning tax*.

In this work, we propose a counterfactual learning approach to credit assignment under sparse terminal rewards. Even without step-level supervision, differences among reasoning trajectories sampled for the same input naturally contain process-level information. Here, counterfactual refers to alternative trajectories sampled for the same input. Disagreements between these trajectories provide structural information about alternative reasoning paths and their outcomes. By systematically comparing these counterfactual paths and aligning their differences with final outcomes, we construct informative learning signals that are more sensitive to intermediate decisions.

Building on this insight, we introduce *Implicit Behavior Policy Optimization (IBPO)*, a framework for process-level credit assignment induced by counterfactual trajectory comparison. IBPO defines a general multi-trajectory comparison operator and uses it to construct an implicit advantage estimator. The estimator reweights terminal rewards according to trajectory-level differences, improving the within-group structure of the pre-normalization advantage signal and amplifying learning signals at frequent decision errors. IBPO does not require step-level annotations, process reward models, or an additional value network. It can be integrated seamlessly with existing sequence-level RL optimizers to use these process-sensitive training signals.

Contributions. Our main contributions are as follows:

- **Counterfactual modeling of credit assignment.** We introduce a counterfactual-learning perspective on credit assignment in LLM reinforcement learning, treating multiple reasoning trajectories for the same input as approximations to alternative decisions. We use inconsistencies among these trajectories to construct process-sensitive learning signals, even in the absence of step-level rewards.

- **Implicit process-level advantages and the IBPO framework.** We formalize a general multi-trajectory comparison operator and use it to construct an implicit process-level advantage estimator, yielding the IBPO framework.

- **Analysis of variance reduction and positive transfer.** We provide a conditional variance analysis showing that, when the counterfactual comparison signal complements terminal-reward noise and the shaping weight is moderate, IBPO can reduce the dispersion of the pre-normalization, within-group centered advantage terms. Because GSPO further applies within-group standard-deviation normalization, the full policy-gradient variance also depends on the policy-gradient terms themselves. We therefore analyze how this mechanism improves the structure of update signals and mitigates the learning tax.

- **Mechanism-driven empirical validation.** We evaluate IBPO on multiple mathematical and code-reasoning benchmarks. Under matched training compute, IBPO improves benchmark accuracy and reaches a fixed training-reward threshold earlier than the GSPO baseline.

2 Related Work

Group Relative Policy Optimization (GRPO). GRPO Shao et al. [2024] is a recent reinforcement-learning algorithm for fine-tuning LLMs on reasoning tasks and has achieved strong results in systems such as DeepSeek-R1 Guo et al. [2025a]. GRPO estimates group-relative advantages from within-group sampling, replacing the explicit value modeling used in PPO and enabling faster, more efficient training. However, GRPO suffers from entropy collapse, reward collapse, and unstable convergence Yu et al. [2025]. These issues largely stem from its reliance on the assumption that the *final reward is sufficient to characterize the reasoning trajectory*. This assumption often fails in long-horizon reasoning, where success depends on a sequence of interdependent steps, leading to ill-posed credit assignment and inflated gradient variance. GSPO Zheng et al. [2025] improves on GRPO by computing the importance ratio at the sequence level.

Self-correction strategies. Self-correction has emerged as a promising direction for strengthening reasoning. For example, selective reflection tuning Li et al. [2024] enables models to reflectively evaluate multiple candidate responses and then fine-tune on the best response with supervised learning.

Reward modeling. Reward models are crucial for robust System-2 reasoning, but they are difficult to build. Recent directions include LLM-as-a-Judge frameworks Zheng et al. [2023], Qi et al. [2025], outcome reward models Qwen et al. [2025], Yu et al. [2024], and process reward models (PRMs) that provide step-level feedback for complex tasks Lightman et al. [2024], Luo et al. [2024], Wang et al. [2024b]. PRMs, however, face important limitations: high annotation cost, weak generalization, and noisy signals from automated methods such as Monte Carlo sampling or MCTS Kang et al. [2024], Wang et al. [2024a]. Human-annotated datasets such as PRM800k Lightman et al. [2024] are difficult to scale, and existing automatic annotation methods often produce noisy or inconsistent reward scores. In contrast, IBPO bypasses the need for fine-grained annotation through implicit comparison while still providing process-sensitive training signals. Unlike prior methods, our approach does not assume that the reward decomposes into stepwise reward signals.

Rubric rewards and relational counterfactual shaping. Recent work has also explored using explicit rubrics as reward functions to provide denser feedback for tasks that lack easily verifiable answers Gunjal et al. [2026]. These methods are typically *criterion-based single-output scoring*: given one candidate response, an external rubric or judge scores its quality according to predefined criteria. IBPO focuses on a complementary source of signal: *relational counterfactual differences among multiple on-policy trajectories for the same input*. In other words, rubric methods densify the reward for a single output through external criteria, whereas IBPO constructs process-sensitive shaping signals by comparing differences between successful and failed trajectories under the same prompt. Combining the two is a future direction: calibrated rubric rewards could serve as terminal evaluators, while IBPO's comparison operator $\mathcal{M}$ could extract cross-trajectory credit-assignment information on top of those rewards.

SCoRe Kumar et al. [2025] iteratively reuses previously generated responses and prompts the model to identify errors in earlier outputs. It improves reasoning accuracy through multiple rounds of reinforcement learning, but its training pipeline relies on multi-stage generation and optimization cycles. IBPO also includes correction generation, but this generation is used only to construct shaping signals and auxiliary correction objectives within a single round of joint training. All additional generation, verification, and comparison costs are included in the training-compute budget.

3 Method

3.1 Problem Formulation

We consider multi-step reasoning reinforcement learning with terminal rewards. Given an input x, the policy π_θ generates a reasoning trajectory as a token sequence of length T:

$$\tau = (y_1, y_2, \ldots, y_T), \quad y_t \sim \pi_\theta(\cdot \mid x, y_{<t}), \tag{1}$$

where y_t denotes the token generated at step t.

The environment provides a **sequence-level reward** only when the trajectory is complete:

$$R(\tau) \in [-1, 1]. \tag{2}$$

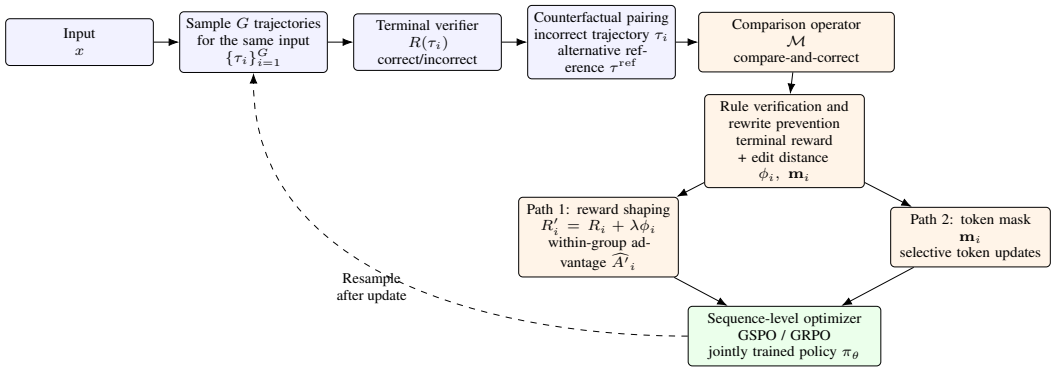

Figure 2: IBPO method diagram. For the same input, the policy first samples multiple candidate trajectories, and a terminal verifier provides sequence-level feedback. Incorrect trajectories are paired counterfactually with alternative reference trajectories; when a correct trajectory is available, it is used as the reference, otherwise an incorrect trajectory from the same group is used. The comparison operator $\mathcal{M}$ produces process-sensitive signals, which are passed through rule verification and rewrite-prevention filters before being injected into a standard sequence-level optimizer through sequence-level reward shaping or token-level gradient masking.

In most reasoning tasks, $R(\tau)$ is sparse, for example the binary correctness of the final answer, and provides no explicit step-level supervision. To unify notation, the theoretical derivations in the main text use a symmetric binary reward $Y = R(\tau) \in \{-1, 1\}$, whereas the experimental implementation and algorithm pseudocode use the correctness indicator $r(x, y) \in \{0, 1\}$. The two are related by $Y = 2r - 1$, and all within-group normalized advantages preserve the same ranking structure under affine transformations. For the same comparison signal, coefficients on these scales satisfy $\lambda_Y = 2\lambda_r$.

The standard policy-gradient objective is:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[A(\tau)\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(y_t \mid x, y_{<t})\right]. \tag{3}$$

In multi-step reasoning tasks, the central challenge is not merely reward sparsity itself, but the fact that **terminal rewards assign credit to early decisions in an extremely unstable way**. When a local error occurs at an early step, its effect often cascades through and is amplified by later reasoning steps. Yet the error is reflected only indirectly through the terminal reward, producing highly noisy gradient signals whose variance can grow substantially with trajectory length.

IBPO as a framework rather than an implementation. We emphasize that IBPO is a *training framework* for credit assignment under sparse terminal rewards, not a specific correction or rewriting algorithm. Its core contribution is to establish counterfactual trajectory comparison as a general mechanism for inducing implicit process-level learning signals. Concretely, the multi-trajectory comparison operator $\mathcal{M}$ in our framework is an abstract operator whose role is to extract cross-trajectory differences and generate learning signals that reflect differences in process-level decisions. The IBPO framework does not depend on implementation details such as how counterfactual differences are computed or what comparison mechanism is used. The operator $\mathcal{M}$ can be instantiated in many ways, such as consistency scoring, relative ranking, or error detection, but these are implementation details rather than constituent parts of the IBPO framework itself. Thus, IBPO's core contribution lies in its framework-level design, while concrete instantiations can be customized to task requirements.

Concrete comparison mechanisms such as correction, verifier-based ranking, or consistency scoring should be viewed as *instantiations* of the comparison operator used within the IBPO framework. We implement and evaluate compare-and-correct; ranking and consistency scoring are future instantiations. The variance analysis applies when the comparison signals satisfy its stated sufficient conditions.

3.2 Counterfactual Trajectory Comparison

Counterfactual trajectory comparison and the role of $\mathcal{M}$. The core idea of IBPO is to sample multiple reasoning trajectories for the same input and use their differences as counterfactual approximations for constructing process-sensitive learning signals. Specifically, we sample G trajectories from the policy. Fully correct trajectories require no additional signal. For each incorrect target trajectory τ_i, we pair it with $K-1$ alternative reference trajectories to form a target-centered K-tuple. When correct trajectories exist in the group, references are preferentially sampled from the correct trajectories; when there are too few correct trajectories, we replicate them to reach the required number. If no correct trajectory exists, we randomly select other incorrect trajectories as alternative references while excluding the target trajectory itself.

$$\mathcal{T}_i = \left(\tau_i; \tau_{i,\mathrm{ref}}^{(1)}, \ldots, \tau_{i,\mathrm{ref}}^{(K-1)}\right), \qquad K \geq 2. \tag{4}$$

We introduce a **multi-trajectory comparison operator**

$$\mathcal{M}(\mathcal{T}_i) = s_i \quad \text{or} \quad \mathcal{M}(\mathcal{T}_i) = (s_i, \mathbf{m}_i), \qquad s_i \in [0,1], \quad \mathbf{m}_i \in \{0,1\}^{T_i}. \tag{5}$$

where s_i is a comparison signal for the target trajectory τ_i. In compare-and-correct, recoverability is evaluated conditional on the sampled reference, correction policy, and terminal verifier. When a token-level variant is used, $\mathcal{M}$ also outputs a token mask $\mathbf{m}_i$ for the target trajectory τ_i; otherwise only the scalar signal s_i is used. The compare-and-correct outcome is checked by rule-based terminal rewards. The IBPO framework assumes only that the target-trajectory-dependent output of $\mathcal{M}$ is sensitive to the counterfactual differences between τ_i and its reference trajectories.

The output of $\mathcal{M}$ is not limited to sequence-level scalar signals. It can also produce token-level signals, such as the fraction of unchanged tokens used for reward shaping (IBPO-ratio), or a token-level mask used to block gradients on unchanged tokens (IBPO-mask). Detailed definitions of these variants are provided in Appendix I.2.

Per-trajectory shaping function. Given the scalar comparison output s_i from $\mathcal{M}(\mathcal{T}_i)$, we define the per-trajectory shaping function:

$$\phi_i = \begin{cases} 0 & \text{if } \tau_i \text{ is correct,} \\ s_i \in [0,1] & \text{otherwise.} \end{cases} \tag{6}$$

The correction outcome underlying ϕ is checked by rule-based terminal rewards. The function maps the comparison signal to a scalar shaping term for trajectory τ_i. Importantly, ϕ_i depends on τ_i only through its relationship to other counterfactual trajectories, and it requires neither explicit step-level annotation nor value estimation.

Token-level masking. When the operator $\mathcal{M}$ produces a structural token-selection signal, the comparison signal can be refined from the sequence level to a token-level mask. Let $\mathbf{m}_i = (m_{i,1}, \ldots, m_{i,T}) \in \{0,1\}^T$ be the token-level mask produced by $\mathcal{M}$, where $m_{i,t} = 1$ indicates that the t-th token is selected for updating and $m_{i,t} = 0$ indicates that it is not selected. The policy gradient can then selectively update only the specified tokens:

$$\nabla_\theta \mathcal{J}_i^{\mathrm{mask}} = \frac{1}{M_i} \sum_{t=1}^{T} m_{i,t} \cdot \widehat{A'}_i \cdot \nabla_\theta \log \pi_\theta(y_t \mid y_{<t}, x), \qquad M_i = \max\left(1, \sum_{t=1}^{T} m_{i,t}\right). \tag{7}$$

This masking mechanism uses structural differences between a trajectory and its verified correction to select tokens for updating; the update direction is determined by the sequence-level advantage. Concrete mask construction methods, such as edit-distance-based masking, are described in Appendix I.2.

3.3 Implicit Process-Level Advantage Estimation

To inject comparison signals into optimization, we provide two complementary paths that replace the coarse feedback determined solely by $R(\tau)$.

Path 1: sequence-level reward shaping. For a candidate trajectory $\tau_i^{(k)}$, we define its shaped reward as:

$$R'_i(x) = R(\tau_i) + \lambda\,\phi_i \tag{8}$$

where $0 \le \lambda\phi_i < 1$, $\lambda\phi_i = 0$ when $R(\tau_i) = 1$, and $\lambda\phi_i$ is a nonnegative value below 1 when $R(\tau_i) = -1$. Although R'_i remains a sequence-level scalar, its value is conditioned on counterfactual comparisons among multiple trajectories and therefore statistically encodes process-level credit information.

We center $\lambda\phi_i$ through within-group advantage normalization:

$$\widehat{A'}_i = \frac{R'_i(x) - \text{mean}\left(\{R'_i(x)\}_{i=1}^{G}\right)}{\text{std}\left(\{R'_i(x)\}_{i=1}^{G}\right)}. \tag{9}$$

This normalization fixes the empirical scale of each within-group normalized advantage. Therefore, our theoretical analysis focuses on how the pre-normalization centered shaped rewards alter the within-group signal structure. The full post-normalization policy-gradient variance also depends on sequence-level log-probability gradients, trajectory length, and the correlation between the shaping signal and effective update directions.

Path 2: token-level gradient masking. When $\mathcal{M}$ produces a token-level mask $\mathbf{m}_i$ (as described above), the policy gradient can perform selective updates at token granularity (Eq. (7)). This path combines the sequence-level advantage $\widehat{A'}_i$ with a token-level mask: the advantage is still determined at the sequence level by counterfactual comparison, but gradients flow only through tokens marked for update. Neither path requires explicit step-level annotation or a value model.

3.4 Hypothesis: Positive Backward Transfer in Multi-Task Learning

Counterfactual trajectory comparison does not directly provide explicit error labels to the model. Instead, by contrasting differences among multiple reasoning paths, latent errors become more salient during comparison. When this comparison behavior is jointly optimized with the base reasoning task, it may induce *positive backward transfer* from the auxiliary comparison task to the original reasoning task from the perspective of multi-task learning: the model may learn to suppress local errors more quickly, thereby accelerating convergence on the base task. This hypothesis motivates reducing ineffective updates and the *learning tax* commonly observed in long-horizon reinforcement learning.

Testable prediction. This hypothesis motivates examining gains on difficult reasoning tasks, particularly when correct trajectories are extremely scarce and training signals are dominated by sparse terminal rewards. Concretely, we expect faster convergence and more stable training dynamics on challenging benchmarks. The process-level advantage-estimation mechanism and conditional variance analysis of IBPO are provided in Appendix G.

4 Experiments

Instantiating IBPO. As discussed above, IBPO provides a reward and advantage-construction framework based on multi-trajectory counterfactual comparison, rather than introducing a new sequence-level policy optimizer. Therefore, in concrete experiments, IBPO must be instantiated on top of an existing sequence-level reinforcement-learning method.

In this work, we instantiate IBPO on top of GSPO. IBPO focuses on constructing process-sensitive advantage estimates from counterfactual multi-trajectory comparison under terminal rewards, while GSPO serves purely as the sequence-level optimizer that carries and applies these advantage signals. As shown in Appendix K, we provide a detailed IBPO+GSPO framework. Appendix D reports the GRPO results. The concrete instantiation of the comparison operator $\mathcal{M}$ is described in Appendix I.

Tasks and datasets. We evaluate the proposed method on a collection of mathematical and code-reasoning benchmarks. The selected tasks are designed to assess symbolic manipulation, multi-step reasoning, domain-specific mathematical understanding, and code reasoning.

HMMT25 Balunovic et al. [2025], *AIME25* Art of Problem Solving [2025] and *LiveCodeBench v6 (25.02-25.05)* Jain et al. [2025] **Base models.** Qwen3-32B Yang et al. [2025]. Qwen3-Next-80B-A3B-Thinking Qwen Team [2025].

We configure Qwen3-32B with a 32k-token context length and Qwen3-Next-80B-A3B-Thinking with a 256k-token context length. Inference is performed with the VLLM engine (version 0.11.2).

Training data and selection protocol. RL training prompts come from DeepMath-103K He et al. [2025] and OpenCodeReasoning Ahmad et al. [2025]. AIME24, AIME25, HMMT25, MATH-500, GPQA-Diamond, and LiveCodeBench v6 are reserved for evaluation and are not used for RL training, validation, checkpoint selection, or hyperparameter tuning. We follow the decontamination analyses released by the dataset publishers. We use each run's last training checkpoint, with hyperparameters fixed before training. Mathematical tasks use answer matching and code tasks use programmatic tests. The additional comparisons include AIME24 Art of Problem Solving [2024], MATH-500 Hendrycks et al. [2021b], and GPQA-Diamond Rein et al. [2024].

Direct credit-assignment baselines. We additionally compare with PRIME Cui et al. [2026], Math-Shepherd Wang et al. [2024b], VinePPO Kazemnejad et al. [2025], SPO Guo et al. [2025b], GiGPO Feng et al. [2025], and High-Entropy Token Selection Wang et al. [2025], using Qwen3-32B, the same training data and terminal verifiers, and matched end-to-end training compute. Math-Shepherd additionally uses PRM supervision. Table 2 reports the results.

Baselines and comparison setup. We compare IBPO instantiated on GSPO Zheng et al. [2025] (denoted **IBPO+GSPO**) with the following baselines: (1) vanilla GSPO; (2) **GSPO with prompt**, which introduces an additional correction prompt only at inference time after GSPO training to generate a revised output; (3) **GSPO with SCoRe**, which uses a SCoRe-style two-stage self-correction pipeline: it first generates an initial answer, then constructs a correction input from that answer and optimizes the corrected output. Its additional generation and optimization costs are counted under the same training-compute budget; and (4) **GSPO Best-of-N (oracle)**, which samples $N = 8$ candidate answers for the same input and uses the same terminal verifier as the main experiments to select a candidate that passes verification. If multiple candidates pass, the first passing candidate is selected; if none pass, the first sample is retained. It serves as an oracle reference selected by the final evaluator.

Experiments are conducted on 32 Nvidia A800 (80G) GPUs. Training hyperparameters are as follows: initial learning rate 5×10^{-7}; cosine-annealing learning-rate schedule with minimum learning-rate ratio 0.1; linear warmup for 3% of the total training steps; entropy-regularization coefficient $\beta = 0$; 64 rollouts per input for GSPO and 8 rollouts per input for IBPO+GSPO; and mini-batch size 32.

Compute-matching protocol. We align the budgets of different methods by estimated training compute based on GPU runtime and GPU utilization. Details are provided in Appendix H.

Full-rewrite filtering. To prevent the model from completely rewriting rather than locally repairing a solution during correction, which would lead to reward hacking, we detect full rewrites using edit distance and set their shaping rewards to zero. This mechanism is a defensive safeguard rather than a core component; see Appendix I.3.

Our experiments focus on mathematical and code-reasoning tasks. The applicability of the IBPO framework primarily depends on whether verifiable terminal feedback is available during training; further discussion is provided in Appendix A.

5 Results and Analysis

Table 1 reports performance comparisons across multiple reasoning benchmarks. In addition to the base IBPO variant with sequence-level reward shaping, we evaluate two token-level edit-distance-based counterfactual analysis variants, IBPO(ratio) and IBPO(mask), whose detailed definitions appear in Appendix I.2. Under matched training compute, GSPO+IBPO consistently outperforms GSPO, test-time prompt correction, and SCoRe-style self-correction baselines across both base models and all three datasets; IBPO(mask) achieves the highest performance. Compared with GSPO, GSPO+IBPO improves Qwen3-32B by 8.2, 10.7, and 7.0 percentage points, and Qwen3-Next by 3.7, 4.4, and 4.5 percentage points.

Method	Qwen3-32B			Qwen3-Next		
	AIME25	**LiveCodeBench**	**HMMT25**	**AIME25**	**LiveCodeBench**	**HMMT25**
GSPO + IBPO	85.3 ± 1.2	75.3 ± 1.1	62.6 ± 1.4	93.8 ± 1.5	75.3 ± 1.3	80.4 ± 1.6
GSPO + IBPO(ratio)	85.0 ± 1.3	74.9 ± 1.2	62.2 ± 1.5	93.4 ± 1.4	74.7 ± 1.4	80.0 ± 1.5
GSPO + IBPO(mask)	85.9 ± 1.1	76.0 ± 1.0	63.1 ± 1.3	94.4 ± 1.3	75.8 ± 1.2	81.1 ± 1.4
GSPO	77.1 ± 1.4	64.6 ± 1.5	55.6 ± 1.3	90.1 ± 1.1	70.9 ± 1.7	75.9 ± 1.4
GSPO with prompt	78.2 ± 0.8	65.3 ± 0.9	56.4 ± 1.6	90.6 ± 1.2	71.4 ± 1.3	76.5 ± 1.8
GSPO with SCoRe	78.3 ± 1.2	65.2 ± 1.2	56.7 ± 0.8	90.7 ± 1.5	71.9 ± 1.2	76.3 ± 0.9
GSPO Best-of-N (oracle)	77.9 ± 0.9	66.1 ± 1.7	57.1 ± 0.9	91.5 ± 0.9	71.6 ± 1.5	77.2 ± 1.4

Table 1: Experimental results with Qwen3-32B and Qwen3-Next-80B-A3B-Thinking. For each test set, we conduct 64 independent evaluations and report average accuracy; each evaluation submits one final answer per problem. Non-oracle rows use pass@1 without majority voting. We report means over 5 random seeds with 95% bootstrap confidence intervals (mean $\pm$ 95% CI). All methods are budget-aligned by the same training compute, and IBPO's base generation, correction generation, verification, and comparison costs are all included in the total budget. The Best-of-N method uses $N = 8$.

Method	**AIME25**	**HMMT25**	**LiveCodeBench v6**	**AIME24**	**MATH-500**	**GPQA-Diamond**
PRIME	78.5 ± 1.1	58.1 ± 1.0	64.8 ± 0.9	85.1 ± 0.8	97.7 ± 0.3	68.3 ± 1.2
Math-Shepherd	77.6 ± 1.1	57.1 ± 1.2	64.5 ± 0.9	84.7 ± 1.0	97.6 ± 0.4	68.2 ± 1.1
SPO	79.9 ± 1.2	59.8 ± 0.9	66.9 ± 1.0	83.9 ± 0.9	97.6 ± 0.3	68.5 ± 1.0
GiGPO	76.4 ± 0.9	56.9 ± 1.0	63.4 ± 1.1	83.5 ± 1.0	97.3 ± 0.4	68.2 ± 1.1
VinePPO	78.1 ± 1.0	57.5 ± 1.1	64.4 ± 0.9	85.0 ± 0.8	97.9 ± 0.4	68.4 ± 1.1
High-Entropy Token Selection	77.5 ± 0.7	57.2 ± 0.6	63.8 ± 0.8	84.6 ± 0.9	97.4 ± 0.5	$\mathbf{68.7 \pm 1.2}$
GSPO+IBPO (mask)	$\mathbf{85.9 \pm 1.1}$	$\mathbf{63.1 \pm 1.3}$	$\mathbf{76.0 \pm 1.0}$	$\mathbf{88.1 \pm 1.1}$	$\mathbf{98.2 \pm 0.3}$	68.5 ± 1.2

Table 2: Direct credit-assignment and token-selection comparisons with Qwen3-32B under the common training and compute-accounting protocol. The methods share the base model, training data, and terminal verifiers; Math-Shepherd additionally uses PRM supervision. Values are means with 95% bootstrap confidence intervals.

These results show improved benchmark accuracy under matched training compute; on Qwen3-Next, IBPO reaches the fixed training-reward threshold with less cumulative compute. Figure 3 further compares the training dynamics. The main experiments in Table 1 use a unified training-compute matching protocol: to avoid treating two-stage generation with different context lengths as having the same cost, we do not use the number of rollouts as the matching criterion. GSPO samples 64 base responses per input prompt, whereas IBPO samples 8 base responses per prompt and performs comparison-correction generation for incorrect responses. The two methods are aligned by unified training compute, and IBPO's base generation, correction generation, verification, rewrite filtering, and parameter-update costs are all included in the total budget. The training reward always refers to the batch average of the original terminal reward $r(x, y)$ rather than the shaped reward $r'(x, y)$; therefore, the reward curves for GSPO and IBPO in Fig. 3 use the same measurement.

Under budgets aligned by training compute, GSPO+IBPO consistently achieves higher training rewards throughout training and enters the high-reward regime earlier under the same compute constraint. Table 3 shows that GSPO+IBPO requires only $0.63\times$ the training compute of GSPO to reach a fixed training-reward threshold of 0.75. This indicates that IBPO converts terminal rewards into more effective parameter updates *per unit of compute*, improving training-compute efficiency.

Beyond average performance, GSPO+IBPO also exhibits a smoother training-reward trajectory with smaller fluctuations than GSPO. With only sequence-level terminal rewards, local reasoning errors are often propagated uniformly through the entire generated sequence via shared terminal feedback, causing gradient estimates to be affected by noise from irrelevant time steps. The shaping signals produced by counterfactual trajectory comparison help the model identify error-prone regions more quickly and suppress ineffective updates during optimization. The evolution of correction success rate, the positive-backward-transfer explanation, and a detailed comparison with test-time prompt correction are provided in Appendix F.

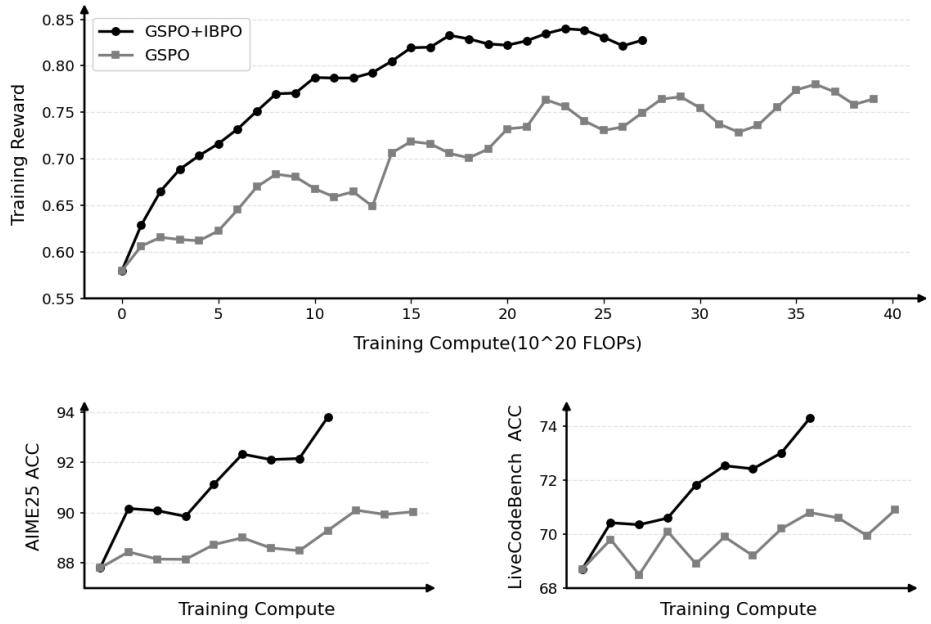

Figure 3: Training curves for fine-tuning Qwen3-Next-80B-A3B-Thinking. The horizontal axis is normalized estimated training compute; training reward is the batch average of the original terminal reward $r(x, y)$.

Method	**Compute@Reward=0.75**
GSPO	1.00×
GSPO + IBPO	**0.63**×

Table 3: With Qwen3-Next-80B-A3B-Thinking, we measure the training budget required to reach a fixed training-reward threshold under a **training-compute matched** setting. Results are normalized relative to GSPO.

Ablation experiments further show that multi-trajectory counterfactual comparison, joint training, and the shaping weight all contribute to final performance. Full results and the λ sensitivity analysis are given in Appendix C.

6 Conclusion

We introduced Implicit Behavior Policy Optimization (IBPO), a reinforcement-learning paradigm that extracts implicit process-level learning signals from sparse terminal rewards through counterfactual comparison of reasoning trajectories. By sampling multiple trajectories per input and comparing their outcomes, IBPO constructs process-sensitive training signals without step-level supervision or an auxiliary value model.

Experiments show that, when combined with sequence-level optimizers such as GSPO, IBPO improves performance on mathematical and code-reasoning benchmarks under matched training compute. Its formulation is agnostic to the underlying reinforcement-learning algorithm and is directly compatible with GRPO variants and other policy-gradient methods, offering a scalable path toward more robust multi-step reasoning in LLMs.

Limitations

IBPO incurs additional compute overhead because it samples multiple counterfactual trajectories for each input. Although our results show higher sample efficiency under a fixed budget, reducing this

overhead remains important for large-scale deployment. In addition, if the counterfactual trajectories contain systematic errors, the comparison signal may weaken. Increasing trajectory diversity or incorporating external verifiers can mitigate this issue.

Our experiments focus primarily on mathematics and code because these tasks have relatively reliable terminal verifiers. For open-ended question answering, writing, dialogue, and other settings that require LLM-as-a-Judge or rubric-model scoring, verifier noise directly affects recoverability judgments and shaping rewards, potentially amplifying erroneous preferences or weakening counterfactual comparison signals. Practical applications can reduce bias from noisy verifiers by using multi-judge consistency, confidence weighting, calibration-set filtering, or enabling shaping rewards only for high-confidence samples.

Our code evaluation uses LiveCodeBench v6; broader code coverage, instruction following, and output diversity are directions for further evaluation.

Societal Impact

IBPO aims to improve the sample efficiency and training stability of reinforcement learning for multi-step reasoning. Potential positive impacts include lowering training-compute costs, improving the reliability of mathematical and code-reasoning systems, and reducing dependence on human step-level annotation. At the same time, stronger reasoning and code-generation capabilities could be used for unreliable automation, for generating incorrect but plausible explanations, or for assisting harmful code writing. Because the method depends on terminal verifiers, deployment to open-ended tasks may also inherit or amplify judge bias. Practical use should be paired with task risk assessment, verifier calibration, safety filtering, and human review.

Acknowledgments and Disclosure of Funding

This work received no third-party funding or in-kind support. The authors declare no competing financial interests relevant to this work.

References

Wasi Uddin Ahmad, Sean Narenthiran, Somshubra Majumdar, Aleksander Ficek, Siddhartha Jain, Jocelyn Huang, Vahid Noroozi, and Boris Ginsburg. Opencodereasoning: Advancing data distillation for competitive coding, 2025. URL https://arxiv.org/abs/2504.01943.

Art of Problem Solving. 2024 AIME I Problems and 2024 AIME II Problems, 2024. URL https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions. Problems copyrighted by the Mathematical Association of America. Accessed: 2026-08-06.

Art of Problem Solving. 2025 AIME I Problems and 2025 AIME II Problems, 2025. URL https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions. Problems copyrighted by the Mathematical Association of America. Accessed: 2026-05-23.

Mislav Balunovic, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Matharena: Evaluating llms on uncontaminated math competitions. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen, editors, *Advances in Neural Information Processing Systems*, volume 38. Curran Associates, Inc., 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/1d27c01ebd3e3aebe226b44fc970d803-Paper-Datasets_and_Benchmarks_Track.pdf.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, Jiarui Yuan, Huayu Chen, Kaiyan Zhang, Xingtai Lv, Shuo Wang, Yuan Yao, Xu Han, Hao Peng, Yu Cheng, Zhiyuan Liu, Maosong Sun, Bowen Zhou, and Ning Ding. Process reinforcement through implicit rewards. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=9SkkifLopZ.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen, editors, *Advances in Neural Information Processing Systems*, volume 38, pages 46375–46408. Curran Associates, Inc., 2025. doi: 10.52202/085713-1544. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/420c9f777c0b4f78d515e53cf74d58b2-Paper-Conference.pdf.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean M. Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=c1bTcrDmt4.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu,

Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, September 2025a. doi: 10.1038/s41586-025-09422-z. URL https://www.nature.com/articles/s41586-025-09422-z.

Yiran Guo, Lijie Xu, Jie Liu, DAN YE, and Shuang Qiu. Segment policy optimization: Effective segment-level credit assignment in rl for large language models. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen, editors, *Advances in Neural Information Processing Systems*, volume 38, pages 114399–114431. Curran Associates, Inc., 2025b. doi: 10.52202/085713-3815. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/a6536243037d1e32c20de85137d478da-Paper-Conference.pdf.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning, 2025. URL https://arxiv.org/abs/2504.11456.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021a. URL https://arxiv.org/abs/2009.03300.

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. *NeurIPS Datasets and Benchmarks*, 2021b. URL https://arxiv.org/abs/2103.03874.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=chfJJYC3iL.

Jikun Kang, Xin Zhe Li, Xi Chen, Amirreza Kazemi, Qianyi Sun, Boxing Chen, Dong Li, Xu He, Quan He, Feng Wen, Jianye Hao, and Jun Yao. Mindstar: Enhancing math reasoning in pre-trained llms at inference time, 2024. URL https://arxiv.org/abs/2405.16265.

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. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 29557–29590. PMLR, 2025. URL https://proceedings.mlr.press/v267/kazemnejad25a.html.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal Behbahani, and Aleksandra Faust. Training language models to self-correct via reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=CjwERcAU7w.

Ming Li, Lichang Chen, Jiuhai Chen, Shwai He, Jiuxiang Gu, and Tianyi Zhou. Selective reflection-tuning: Student-selected data recycling for LLM instruction-tuning. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 16189–16211, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.958. URL https://aclanthology.org/2024.findings-acl.958/.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

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, 2024. URL https://arxiv.org/abs/2406.06592.

OpenAI. gpt-oss-120b & gpt-oss-20b Model Card, 2025. URL https://openai.com/index/gpt-oss-model-card/. Released under the Apache 2.0 license subject to the gpt-oss usage policy.

Zhenting Qi, Mingyuan MA, Jiahang Xu, Li Lyna Zhang, Fan Yang, and Mao Yang. Mutual reasoning makes smaller LLMs stronger problem-solver. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=6aHUmotXaw.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Qwen Team. Qwen3-next: Towards ultimate training & inference efficiency. Qwen Blog, September 2025. URL https://qwen.ai/blog?id=4074cca80393150c248e508aa62983f9cb7d27cd&from=research.latest-advancements-list.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://arxiv.org/abs/2311.12022.

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

Chaojie Wang, Yanchen Deng, Zhiyi Lyu, Liang Zeng, Jujie He, Shuicheng Yan, and Bo An. Q*: Improving multi-step reasoning for llms with deliberative planning, 2024a. URL https://arxiv.org/abs/2406.14283.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439, Bangkok, Thailand, August 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.510. URL https://aclanthology.org/2024.acl-long.510/.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen, editors, *Advances in Neural Information Processing Systems*, volume 38, pages 115452–115486. Curran Associates, Inc., 2025. doi: 10.52202/085713-3850. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/a797c2d2e0c1fdabf4d1ab8cd0b465c6-Paper-Conference.pdf.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Fei Yu, Anningzhe Gao, and Benyou Wang. OVM, outcome-supervised value models for planning in mathematical reasoning. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 858–875, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.55. URL https://aclanthology.org/2024.findings-naacl.55/.

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

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

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=uccHPGDlao.

A Scope of Applicability

Although our experiments focus primarily on mathematical and code-reasoning tasks, the design and applicability of IBPO are not limited to a particular task domain. Its core mechanism is not multi-draft generation or explicit correction per se. Rather, IBPO constructs multiple counterfactual reasoning trajectories for the same input and uses their differences in terminal outcomes and intermediate decisions to induce learning signals that are more sensitive to the reasoning process. From this perspective, IBPO is fundamentally a training paradigm based on counterfactual trajectory comparison. Extensions can be studied when objective or verifiable terminal feedback is available during training, beyond the task formats evaluated here.

In mathematics and programming tasks, the correctness of the final solution has a clear and automatically verifiable definition, making these tasks convenient and reliable testbeds for studying the role of counterfactual trajectory comparison in multi-step reasoning RL. For other task types, such as factual question answering, structured knowledge reasoning, or multi-step decision problems with clear termination conditions, one can likewise design appropriate verifiable reward functions or evaluation criteria to distinguish the terminal quality of different counterfactual trajectories. By comparing multiple counterfactual trajectories sampled for the same input during training and injecting the resulting differences into reward shaping or advantage estimation, the model can statistically learn which intermediate decisions are more likely to lead to success and which are more likely to lead to failure.

Structurally, the key advantage of IBPO lies in its modeling of counterfactual trajectory independence and the resulting implicit process-level credit-assignment mechanism. This mechanism does not rely on explicit step-level annotations or an additional value model. Instead, with only terminal rewards, it introduces more discriminative learning signals for the reasoning process through multi-trajectory counterfactual comparison. This mechanism provides a basis for studying trajectory comparison and advantage construction in other reasoning or decision tasks with verifiable terminal outcomes.

B Existing Assets and Licenses

The existing models, datasets, and evaluation tools used in this paper are all taken from public release pages. Based on the official model cards, dataset cards, or code repositories, we record the following license information:

- **Qwen3-32B**: base model. The official Hugging Face page is `https://huggingface.co/Qwen/Qwen3-32B`, and the license is marked as Apache-2.0.
- **Qwen3-Next-80B-A3B-Thinking**: base model. The official Hugging Face page is `https://huggingface.co/Qwen/Qwen3-Next-80B-A3B-Thinking`, and the license is marked as Apache-2.0. The official Qwen repository also states that its open-weight models use the Apache 2.0 license.
- **MathArena/aime_2025**: AIME 2025 evaluation data. The official Hugging Face dataset page is `https://huggingface.co/datasets/MathArena/aime_2025`, and the license is marked as CC BY-NC-SA 4.0. The AoPS AIME 2025 page also states that the original problems are copyrighted by the Mathematical Association of America.
- **MathArena/hmmt_feb_2025**: HMMT February 2025 evaluation data. The official Hugging Face dataset page is `https://huggingface.co/datasets/MathArena/hmmt_feb_2025`, and the license is marked as CC BY-NC-SA 4.0.
- **LiveCodeBench**: code-evaluation benchmark. The official GitHub repository is `https://github.com/LiveCodeBench/LiveCodeBench`, and the code license is MIT.
- **vLLM**: inference engine. The official GitHub repository is `https://github.com/vllm-project/vllm`, and the license is Apache-2.0.

Beyond the public assets listed above, this paper does not release a new dataset or benchmark.

C Additional Ablation Experiments

To evaluate the contribution of each IBPO component, we conduct ablation experiments with Qwen3-32B on AIME25, LiveCodeBench, and HMMT25. Table 4 summarizes the results. The GSPO, shaping-only, correction-training-only, and full-IBPO rows respectively disable both components, enable shaping alone, enable correction training alone, and enable both, giving the full 2×2 decomposition. The incorrect-reference variant changes only the reference policy; it scores 80.2, 69.6, and 59.2, compared with 85.3, 75.3, and 62.6 for correct-reference-first sampling.

Model Variant	AIME25 (%)	LiveCodeBench (%)	HMMT25 (%)
Full IBPO($K = 2$)	**85.3**	**75.3**	**62.6**
GSPO (baseline)	77.1	64.6	55.6
GSPO + test-time prompt	78.2	65.3	56.4
Self-correction ($K = 1$)	78.6	65.9	57.1
IBPO (correction training only)	79.5	67.5	57.8
IBPO (incorrect references only)	80.2	69.6	59.2
IBPO (shaping only)	80.3	70.2	59.7

Table 4: Ablation results for Qwen3-32B. The $K = 2$ variants use no token mask and share the base model, data, reference policy, configuration, and end-to-end compute budget except for the named factor.

GSPO + test-time prompt. Multi-trajectory comparison is performed only at inference time through prompting. Without joint training, the gains introduced by IBPO disappear, leading to a substantial performance drop. This result is consistent with the *positive transfer* interpretation of IBPO and the effectiveness of implicit process-level rewards.

Self-correction control ($K = 1$). This variant uses only the target answer itself to construct the correction input and provides no alternative reference from other candidate trajectories under the same prompt. It therefore degenerates into a self-correction-only training setup: the model can still attempt to revise its own output, but it cannot use cross-trajectory differences to localize local errors or construct relational counterfactual signals. Accuracy drops substantially without comparison among multiple counterfactual trajectories, indicating that inconsistencies among such trajectories play a key role in error identification.

IBPO (shaping only). In this ablation, we disable joint training for multi-trajectory comparison while retaining the reward-shaping term. This isolates the positive-transfer effect brought by joint training.

λ	**0.4**	**0.6**	**0.8**	**1.0**	**1.2**
Accuracy (%)	83.1	**85.3**	83.6	82.1	81.5

Table 5: Evaluation results under different λ values (IBPO score; Qwen3-32B; AIME25).

We conduct a sensitivity analysis over λ and observe the best performance around 0.6. Overall, the ablation results clearly show that each IBPO component makes a meaningful contribution to the final performance.

D Optimizer Transfer

E Cross-Family Evaluation

We evaluate gpt-oss-20b OpenAI [2025] on AIME24, AIME25, and MMLU Hendrycks et al. [2021a], using the same experimental configuration as the main paper and matched end-to-end GPU time. Table 7 reports the results.

GSPO+IBPO(mask) has point estimates 3.1 and 2.9 percentage points above GSPO on AIME 2025 and AIME 2024, respectively. Its MMLU point estimate matches the Base model at 85.1. These results provide evidence beyond the Qwen family.

Dataset	GRPO	GRPO+IBPO (mask)
AIME25	75.8 ± 1.0	$\mathbf{83.7 \pm 1.2}$
HMMT25	54.6 ± 1.1	$\mathbf{61.4 \pm 1.3}$
LiveCodeBench v6	61.2 ± 0.8	$\mathbf{73.2 \pm 1.1}$
AIME24	83.1 ± 0.9	$\mathbf{87.2 \pm 1.2}$
MATH-500	97.4 ± 0.5	$\mathbf{98.0 \pm 0.4}$
GPQA-Diamond	68.4 ± 1.1	68.4 ± 0.9

Table 6: IBPO instantiated on GRPO with Qwen3-32B using the common training data, deterministic terminal verifiers, and end-to-end compute protocol. Values are means over 5 seeds with 95% confidence intervals.

Base Model	Benchmark	Base	GSPO	GSPO+IBPO (mask)
gpt-oss-20b	AIME 2025	91.5 ± 0.9	92.2 ± 0.8	$\mathbf{95.3 \pm 0.7}$
gpt-oss-20b	AIME 2024	91.8 ± 0.7	92.6 ± 0.9	$\mathbf{95.5 \pm 0.6}$
gpt-oss-20b	MMLU	85.1 ± 0.8	85.0 ± 0.7	$\mathbf{85.1 \pm 0.8}$

Table 7: Cross-family evaluation with gpt-oss-20b under the main-paper configuration and matched end-to-end GPU time.

F Training Dynamics and Additional Result Analysis

Evolution of correction success rate. The informativeness of the shaping signal depends on whether the correction success rate remains in a meaningful intermediate range: if correction almost always succeeds or almost always fails, the signal degenerates into a constant. We track the evolution of correction success rate during Qwen3-32B training on batches from DeepMath-103K and OpenCodeReasoning. The correction success rate is about 12% early in training and gradually rises to about 67% as training progresses. This indicates that the shaping signal remains in an information-rich regime throughout training, neither constantly zero nor constantly one, and thus continues to provide discriminative process-level feedback for policy optimization.

F.1 Positive Backward Transfer and Difficult Tasks

As shown in Fig. 3, introducing IBPO leads to faster performance improvement and faster convergence. We interpret this through the hypothesis of a *positive backward transfer* effect. In multi-task learning, positive backward transfer refers to the phenomenon in which learning a later task (task B) improves performance on an earlier task (task A), reflecting strong generalization. By introducing an auxiliary task based on counterfactual comparison of reasoning trajectories, IBPO may induce a positive transfer effect on the main reasoning task during training.

Concretely, under GSPO training the model receives only a sequence-level reward $R(y)$, which is propagated uniformly across the entire reasoning sequence. When final failure is caused by only a few local tokens, this supervision signal cannot indicate where the error occurred, forcing the model to rely on extensive sampling and iterative updates to gradually internalize these local errors statistically. This substantially increases the sample complexity of learning and gives rise to the *learning tax* commonly observed in reinforcement learning. IBPO introduces an auxiliary task based on counterfactual comparison of reasoning trajectories. By contrasting inconsistencies across different reasoning paths, it increases the *observability* of local errors and accelerates the model's internalization of fine-grained reasoning mistakes.

In difficult reasoning tasks, correct responses are often extremely rare, and the vast majority of model-generated trajectories are incorrect. This distribution makes sequence-level rewards highly sparse and causes policy-gradient estimates to be dominated by negative samples, slowing convergence and potentially destabilizing training. In contrast, without changing the original sampling distribution, IBPO converts a small number of correct reasoning paths into multiple informative learning signals

through counterfactual trajectory comparison, thereby alleviating the learning bottleneck caused by positive-sample scarcity in difficult tasks.

F.2 Comparison with Prompted GSPO

The core difference between GSPO+prompt and IBPO+GSPO is whether joint training is performed. The former conducts multi-trajectory comparison and correction through prompting only at inference time after training is complete, whereas the latter performs multi-trajectory comparison during training and jointly optimizes the model. The experimental results validate the effectiveness of implicit process-level rewards. Overall, these results support learning from trajectory comparisons during training to improve benchmark accuracy.

G Theoretical Analysis: Conditional Variance-Reduction Properties of IBPO

To characterize the role of IBPO in credit assignment, we use representative GSPO-style methods as a baseline and analyze how counterfactual comparison signals affect the variance of within-group centered reward terms. The analysis is conditional: it does not claim that IBPO unconditionally reduces the full policy-gradient variance in all settings, nor that the sample-advantage variance after standard-deviation normalization is further reduced. Instead, it shows that when the comparison signal complements terminal-reward noise and the shaping weight is moderate, the pre-normalization advantage terms constructed by IBPO can have lower within-group dispersion.

G.1 Process-Level Advantage-Estimation Mechanism

Remark G.1 (IBPO's process-level advantage-estimation mechanism). For a multi-step reasoning task with verifiable terminal feedback and comparison signals satisfying the conditions stated below, suppose the policy π_θ generates G trajectories $\{\tau_i\}_{i=1}^{G}$ for a given input x, and each trajectory τ_i receives a sequence-level reward $R(\tau_i)$. For each incorrect target trajectory τ_i, a target-trajectory-dependent comparison signal s_i is generated by counterfactual comparison with $K-1$ alternative reference trajectories; these reference trajectories are correct trajectories when available and otherwise other incorrect trajectories in the group. The comparison signal is injected into optimization through two paths: (i) it is mapped to a shaped reward $R'_i(x)$; and (ii) when $\mathcal{M}$ produces a token-level mask $\mathbf{m}_i$, only the marked tokens are selectively updated.

When the shaping function ϕ_i provides informative process signals for suboptimal trajectories and this signal complements terminal-reward noise, Path 1 can reduce the within-group dispersion of pre-normalization centered reward terms under the sufficient conditions in Proposition G.4 and improve the update-signal structure induced by coarse episode-level rewards. Specifically, in Path 1, ϕ_i provides finer process feedback through reward shaping; in Path 2, the token-level mask further selects updates according to structural differences between the original and corrected trajectories. These two paths motivate the hypothesis of more stable training dynamics.

By introducing the multi-trajectory comparison operator $\mathcal{M}$, we obtain target-trajectory-dependent comparison signals $s_i(x)$ from the G trajectories. Each signal is sensitive to the counterfactual differences between the target trajectory and its reference trajectories. These differences provide structural information about alternative reasoning paths and are injected into optimization through two paths: (i) ϕ_i is mapped to the shaped reward $R'_i(x)$, enabling finer sequence-level credit assignment; and (ii) the token-level mask $\mathbf{m}_i$ restricts gradient updates to modified tokens (Eq. (7)), enabling selective token-granular credit assignment. Thus, IBPO does not rely on an unconditional variance-reduction guarantee. Instead, it uses more informative comparison signals to reduce interference from irrelevant tokens and irrelevant trajectories.

Concretely, ϕ_i provides process-level feedback for incorrect trajectories rather than relying only on terminal rewards; $\mathbf{m}_i$ further filters out gradient contributions from tokens not selected by the structural mask. The following subsection gives a conditional finite-sample variance analysis for the reward-shaping path. The token-masking path is defined as one possible output form of $\mathcal{M}$ in Appendix I.2 and is integrated into GSPO through the masked ratio in Appendix K.2.

G.2 Conditional Variance Reduction for Centered Advantages

We consider a group of trajectories $\{\tau_i\}_{i=1}^G$ sampled from the policy π_θ for a fixed input x. To make the finite-sample variance calculation clear, we analyze an idealized but standard setting: conditional on x, (Y_i, ϕ_i) are exchangeable within the group and share the same marginal variances and covariances. This assumption allows ϕ_i to have weak dependence on other samples through within-group reference selection. Therefore, the conclusion below should be understood as a conditional analysis of the direction in which the IBPO shaping term acts, not as an unconditional guarantee over all implementation details. Let:

- $Y_i = R(\tau_i) \in \{-1, 1\}$ denote the terminal reward of the i-th trajectory; for simplicity, we assume a binary reward. The implementation uses the correctness reward $r(x, y_i) \in \{0, 1\}$; Appendix G uses the affine recoding $Y_i = 2r(x, y_i) - 1$ only to simplify covariance notation.
- $\phi_i \in [0, 1]$ be the counterfactual comparison signal introduced by IBPO, satisfying:

$$\phi_i = \begin{cases} 0, & \text{if } Y_i = 1 \text{ (correct trajectory)}; \\ \geq 0, & \text{if } Y_i = -1 \text{ (incorrect trajectory)}. \end{cases}$$

  Here, ϕ_i measures reference-conditioned, terminally verified recoverability under the sampled reference and correction policy.

Based on this, GSPO and IBPO define the following within-group centered advantage terms; the standard-deviation normalization factor is omitted for now and discussed later:

$$A_i^{\mathrm{GSPO}} = Y_i - \bar{Y}, \quad \text{where } \bar{Y} = \frac{1}{G}\sum_{j=1}^{G} Y_j, \tag{10}$$

$$A_i^{\mathrm{IBPO}} = (Y_i + \lambda\phi_i) - (\bar{Y} + \lambda\bar{\phi}) = A_i^{\mathrm{GSPO}} + \lambda(\phi_i - \bar{\phi}), \tag{11}$$

where $\lambda > 0$ is the shaping weight and $\bar{\phi} = \frac{1}{G}\sum_{j=1}^{G} \phi_j$.

For the general exchangeable setting, we use the following structural condition:

Condition G.2 (Effective negative correlation and non-degeneracy)**.** Let

$$C_{\mathrm{in}} = \mathrm{Cov}(Y_i, \phi_i), \qquad C_{\mathrm{out}} = \mathrm{Cov}(Y_i, \phi_j), \quad j \neq i.$$

The terminal reward Y_i and comparison signal ϕ_i satisfy $C_{\mathrm{in}} - C_{\mathrm{out}} < 0$. In addition, let

$$V_{\mathrm{in}} = \mathrm{Var}(\phi_i), \qquad V_{\mathrm{out}} = \mathrm{Cov}(\phi_i, \phi_j), \quad j \neq i,$$

and require $V_{\mathrm{in}} - V_{\mathrm{out}} > 0$, meaning that the comparison signal is not constant within the group. When cross-sample correlation introduced by within-group reference selection is negligible, $C_{\mathrm{out}} \approx 0$, and the condition reduces to $\mathrm{Cov}(Y_i, \phi_i) < 0$. Intuitively, correct trajectories ($Y_i = 1$) force $\phi_i = 0$, whereas incorrect trajectories ($Y_i = -1$) may receive nonnegative, reference-conditioned correction signals.

Lemma G.3 (Design-induced negative correlation without cross-sample dependence)**.** *Suppose the comparison signal depends only on the target trajectory, so there is no cross-sample dependence and* $\mathrm{Cov}(Y_i, \phi_j) = 0$ *for* $j \neq i$*. If* $0 < \mathbb{P}(Y_i = 1) < 1$*,* $\phi_i = 0$ *when* $Y_i = 1$*, and* $\mathbb{E}[\phi_i \mid Y_i = -1] > 0$*, then*

$$\mathrm{Cov}(Y_i, \phi_i) < 0.$$

Therefore, the negative-correlation part of Condition G.2 holds when $C_{\mathrm{out}} = 0$*.*

Proof. Let $p = \mathbb{P}(Y_i = 1)$ and $m = \mathbb{E}[\phi_i \mid Y_i = -1]$. Since $Y_i \in \{-1, 1\}$ and $\phi_i = 0$ on correct trajectories, we have

$$\mathbb{E}[Y_i\phi_i] = -(1-p)m, \quad \mathbb{E}[Y_i] = 2p - 1, \quad \mathbb{E}[\phi_i] = (1-p)m.$$

Therefore,

$$\mathrm{Cov}(Y_i, \phi_i) = \mathbb{E}[Y_i\phi_i] - \mathbb{E}[Y_i]\mathbb{E}[\phi_i] = -2p(1-p)m < 0.$$

□

Proposition G.4 (Conditional variance reduction for centered advantages)**.** *Under Condition G.2 and group size $G \geq 2$, there exists $\lambda_{\max} > 0$ such that for any $\lambda \in (0, \lambda_{\max})$:*

$$\mathrm{Var}\left(A_i^{\mathrm{IBPO}}\right) < \mathrm{Var}\left(A_i^{\mathrm{GSPO}}\right).$$

Proof. From $A_i^{\mathrm{IBPO}} = A_i^{\mathrm{GSPO}} + \lambda(\phi_i - \bar{\phi})$, expanding the variance gives:

$$\begin{aligned}\mathrm{Var}(A_i^{\mathrm{IBPO}}) &= \mathrm{Var}\big(A_i^{\mathrm{GSPO}} + \lambda(\phi_i - \bar{\phi})\big) \\ &= \mathrm{Var}(A_i^{\mathrm{GSPO}}) + \lambda^2 \mathrm{Var}(\phi_i - \bar{\phi}) + 2\lambda\, \mathrm{Cov}\big(A_i^{\mathrm{GSPO}},\, \phi_i - \bar{\phi}\big). \end{aligned} \tag{12}$$

Note that $A_i^{\mathrm{GSPO}} = Y_i - \bar{Y}$. Because the within-group samples are exchangeable conditional on the input x, the finite-sample covariance can be computed exactly without taking G to infinity. Let

$$C_{\mathrm{in}} = \mathrm{Cov}(Y_i, \phi_i), \qquad C_{\mathrm{out}} = \mathrm{Cov}(Y_i, \phi_j), \quad j \neq i,$$

Then:

$$\mathrm{Cov}\big(A_i^{\mathrm{GSPO}},\, \phi_i - \bar{\phi}\big) = \mathrm{Cov}\big(Y_i - \bar{Y},\, \phi_i - \bar{\phi}\big) = \left(1 - \frac{1}{G}\right)(C_{\mathrm{in}} - C_{\mathrm{out}}) < 0,$$

where the strict inequality is guaranteed by Condition G.2. Similarly, let

$$V_{\mathrm{in}} = \mathrm{Var}(\phi_i), \qquad V_{\mathrm{out}} = \mathrm{Cov}(\phi_i, \phi_j), \quad j \neq i,$$

Then

$$\mathrm{Var}(\phi_i - \bar{\phi}) = \left(1 - \frac{1}{G}\right)(V_{\mathrm{in}} - V_{\mathrm{out}}).$$

By the non-degeneracy part of Condition G.2, $V_{\mathrm{in}} - V_{\mathrm{out}} > 0$; otherwise, the shaping term would not provide distinguishable within-group information.

Let $C = -(C_{\mathrm{in}} - C_{\mathrm{out}}) > 0$ and $V_\phi = V_{\mathrm{in}} - V_{\mathrm{out}} > 0$. Substituting into the expression above yields:

$$\mathrm{Var}(A_i^{\mathrm{IBPO}}) = \mathrm{Var}(A_i^{\mathrm{GSPO}}) + \left(1 - \frac{1}{G}\right)\left(\lambda^2 V_\phi - 2\lambda C\right).$$

Because $G \geq 2$ and $1 - \frac{1}{G} > 0$, when

$$0 < \lambda < \lambda_{\max} = \frac{2C}{V_\phi}$$

the quadratic term in parentheses is strictly negative, and thus $\mathrm{Var}(A_i^{\mathrm{IBPO}}) < \mathrm{Var}(A_i^{\mathrm{GSPO}})$.

This conclusion characterizes only the within-group centered advantage term before standard-deviation normalization. GSPO implementations typically also divide by the empirical within-group standard deviation, so the sample variance after normalization is not itself the object explained by this theoretical result. The full policy-gradient variance also depends on the correlation between $\nabla_\theta \log \pi_\theta(\tau_i \mid x)$ and the shaping signal, trajectory length, the sampling distribution, and optimizer implementation. Therefore, the proposition should be understood as a consistency analysis showing how the IBPO shaping term improves the structure of update signals, rather than as an unconditional guarantee on the full policy-gradient variance. □

Discussion. The proposition shows that, under Condition G.2, the shaping term $\lambda\phi_i$ introduced by IBPO's counterfactual comparison can reduce the variance of pre-normalization centered advantage terms. It is important to emphasize that the effective negative correlation here is not an independently discovered empirical law, but a structural condition partially induced by the design of the shaping term: correct trajectories typically do not require recoverability rewards, while incorrect but repairable trajectories receive additional shaping signals. Lemma G.3 explicitly demonstrates this in the target-trajectory-local case without cross-sample dependence. The C_{out} term in Condition G.2 extends the statement to a more general exchangeable setting, capturing weak dependence that may arise from within-group reference selection. Thus, the proposition does not prove that IBPO necessarily reduces variance in all settings. Rather, it shows that when the shaping signal genuinely distinguishes recoverable from unrecoverable errors and λ is not too large, the IBPO advantage estimate is not merely adding arbitrary noise, but has a plausible mechanism for improving the structure of within-group update signals.

Relation to training dynamics. Proposition G.4 provides a directional explanation, not an unconditional guarantee on the full policy-gradient variance. It shows that when the shaping signal truly distinguishes recoverable from unrecoverable errors and the shaping weight is not too large, IBPO's advantage estimate need not merely add arbitrary noise and may instead improve the within-group update-signal structure. The smoother reward evolution and faster convergence in Fig. 3 are consistent with this mechanistic explanation.

H Training Compute Accounting

To ensure fair comparison, we match the total training-compute budget of different methods by considering: (1) the number of sampled trajectories and (2) the total compute cost. Therefore, we compare performance only under the same training-compute budget.

In our experiments, the compute budgets of IBPO+GSPO and GSPO are matched through a unified training-compute estimate. Specifically, for each prompt x, IBPO+GSPO first generates 8 responses y. For each incorrect response y_i, it concatenates y_i with the original input x and a reference response to form a new input, then generates a corrected output. When a correct response exists in the group, the reference is preferentially sampled from correct responses; otherwise, it is randomly sampled from other incorrect responses in the group, excluding the target response y_i itself. The correction-stage input sequence is longer because it concatenates additional context, and the quadratic complexity of attention makes each correction trajectory more expensive than base sampling. Therefore, we do not use rollout count as the budget-alignment criterion. The horizontal axis in Fig. 3 corresponds to this training-compute budget.

On a sample of 10 prompts, the correction stage took 1.6 times the GPU time of base generation.

I Instantiation Details of IBPO

This appendix presents one concrete instantiation of IBPO used in our experiments, namely the *compare-and-correct* mechanism, along with the integrated training pipeline and implementation details when combined with GSPO. We emphasize that the following design is one concrete choice for the comparison operator $\mathcal{M}$ in the main text; the core formulation of IBPO does not depend on this particular instantiation.

I.1 Operator

Instantiation choice. In our experiments, we instantiate the general comparison operator $\mathcal{M}$ with a *compare-and-correct* mechanism. We emphasize that this is *one concrete implementation* of IBPO, chosen for its simplicity and effectiveness on verifiable reasoning tasks, rather than a requirement of the IBPO formulation itself.

Concretely, we first generate multiple candidate reasoning trajectories, then use the model itself to compare these trajectories and rewrite them into corrected outputs. This implementation can be viewed as one concrete choice of $\mathcal{M}$ that maps counterfactual differences to a computable shaping term $\phi(\cdot)$. To avoid tying the contribution of this work to a specific implementation, we defer details such as how trajectory diversity is induced and how comparison inputs are constructed to this appendix.

Given two candidate reasoning trajectories/responses for the same input x, namely the target response y and the reference response y^{ref}, we construct a correction input $\tilde{x} = (x; y, y^{\mathrm{ref}})$ and let the model generate a revised output conditioned on $\tilde{x}$:

$$\hat{y} \sim \pi_\theta(\cdot \mid \tilde{x}), \qquad \hat{y} = \mathcal{C}(x; y, y^{\mathrm{ref}}). \tag{13}$$

Prompt template. In all experiments, we use the following compare-and-correct instruction template, with minor adjustments for task format when needed:

> You are given two candidate solutions to the same problem. Compare them step by step, identify any inconsistencies or errors, and then produce a corrected solution and final answer.

Reference sampling. For each target response y_i to be corrected, we sample the reference response y_i^{ref} as follows. If the group contains at least one correct response, we sample uniformly from the set of correct responses. Otherwise, we sample uniformly from the set of other incorrect responses $\{y_j : r(x, y_j) = 0,\ j \neq i\}$. Thus, the fallback reference response is never the target response itself. This strategy is designed to provide a relatively stronger, or at least different, counterfactual reference for comparison without introducing external supervision.

I.2 Shaping Instance: Recoverability-Induced Reward

We use a shaping instance based on *recoverability* to define $\phi(\cdot)$. Let $r(x, y) \in \{0, 1\}$ denote the terminal correctness reward, i.e., whether the final answer is correct. For an incorrect response y and its corrected output $\hat{y} = \mathcal{C}(x; y, y^{\text{ref}})$, we define the implicit process-shaping term as:

$$\Delta(x; y, y^{\text{ref}}) = \rho \cdot \mathbb{I}[r(x, y) = 0 \ \wedge \ r(x, \hat{y}) = 1], \qquad \rho = 0.5. \tag{14}$$

The resulting shaped sequence-level reward is

$$r'(x, y) = r(x, y) + \lambda \, \Delta(x; y, y^{\text{ref}}). \tag{15}$$

Token-level edit-distance variants. When correction succeeds, i.e., $r(x, y) = 0 \wedge r(x, \hat{y}) = 1$, we can further localize the modified tokens by computing the token-level edit distance between the original response y and the corrected output $\hat{y}$. Let $y = (y_1, \ldots, y_T)$ and $\hat{y} = (\hat{y}_1, \ldots, \hat{y}_T)$ after alignment, and define the set of unchanged tokens as $\mathcal{U} = \{t : y_t = \hat{y}_t\}$.

IBPO-ratio (unchanged-ratio reward shaping): use the fraction of unchanged tokens $|\mathcal{U}|/T$ as the recoverability measure, replacing the fixed scale ρ in Eq. (14):

$$\Delta^{\text{ratio}}(x; y, y^{\text{ref}}) = \frac{|\mathcal{U}|}{T} \cdot \mathbb{I}[r(x, y) = 0 \ \wedge \ r(x, \hat{y}) = 1]. \tag{16}$$

A higher unchanged ratio indicates a more local verified correction and is used as a structural recoverability signal for shaping.

IBPO-mask (token-level gradient mask): mask the gradient contribution of unchanged tokens and apply policy-gradient updates only to modified tokens. Define the token-level mask $m_t = \mathbb{I}[t \notin \mathcal{U}]$; then the gradient coefficient for the t-th token in the policy gradient is multiplied by m_t:

$$\nabla_\theta \mathcal{J}^{\text{mask}} \propto \frac{1}{M} \sum_{t=1}^{T} m_t \cdot \widehat{A}(y) \cdot \nabla_\theta \log \pi_\theta(y_t \mid y_{<t}, x), \qquad M = \max\left(1, \sum_{t=1}^{T} m_t\right), \tag{17}$$

Here $\widehat{A}(y)$ is the within-group normalized advantage obtained from the sequence-level reward or shaped reward and broadcast to the selected tokens. It is not a token-level value model and is not an advantage estimated token by token. Unchanged tokens ($m_t = 0$) are excluded from this masked update; edit distance supplies the structural update support. Its integration with GSPO is given in the masked-ratio form in Appendix K.2.

Why not penalize failed correction. When $r(x, y) = 0$ and $r(x, \hat{y}) = 0$, we do not apply an additional negative penalty. This avoids misattributing insufficient correction ability or poor reference quality to the intrinsic quality of the original reasoning. This design choice helps prevent unnecessary bias and training instability.

I.3 Full-Rewrite Detection and Suppression

Problem. During counterfactual compare-and-correct generation, the model may not locally repair the original incorrect trajectory. Instead, it may ignore the original reasoning entirely and generate a new solution from scratch. This "full rewrite" behavior can lead to reward hacking: when the corrected output happens to be correct, the shaping reward is incorrectly attributed to the "recoverability" of the original trajectory, even though the original reasoning process was not actually used.

Detection mechanism. We use Python's token-level edit distance to detect full rewrites. Let $d(a, b)$ denote the normalized edit distance between sequences a and b, with values in $[0, 1]$. When correction

succeeds, i.e., $r(x, y) = 0 \wedge r(x, \hat{y}) = 1$, we classify the case as a full rewrite and set $\Delta = 0$ if both of the following conditions hold:

$$d(y, \hat{y}) > \alpha \quad \text{and} \quad d(y, \hat{y}) > d(\hat{y}, y^{\text{ref}}), \tag{18}$$

where α is the edit-distance threshold. The first condition detects how far the corrected output deviates from the original trajectory. The second condition confirms that the corrected output is closer to the reference answer than to the original trajectory, meaning that the model tends to copy the reference rather than repair the original reasoning.

Combining this with Eq. (14), the complete shaping term with rewrite filtering is:

$$\Delta(x; y, y^{\text{ref}}) = \rho \cdot \mathbb{I}\big[r(x, y) = 0 \wedge r(x, \hat{y}) = 1 \wedge \neg\text{rewrite}(y, \hat{y}, y^{\text{ref}})\big]\,. \tag{19}$$

Applicability to code tasks. The applicability of IBPO does not depend on whether the output is natural-language reasoning text; it depends on whether the training task provides verifiable terminal feedback. Code-generation tasks naturally satisfy this condition: candidate programs can receive binary or scalar rewards through unit tests, execution results, or compilation feedback. Therefore, IBPO can directly treat multiple candidate programs for the same programming problem as counterfactual trajectories and use differences between programs that pass and fail tests to construct process-sensitive shaping signals.

In our LiveCodeBench experiments, program semantic correctness is always determined by execution-based test feedback, not by edit distance. Edit distance is used only as a structural-fidelity check in the current compare-and-correct instantiation: it determines whether the corrected program still preserves much of the local structure of the original failing program, thereby avoiding the erroneous attribution of a fully rewritten correct program to the "recoverability" of the original failed trajectory. Because program semantics are often expressed through local token dependencies, syntactic structure, and control-flow organization, higher token/structural overlap in practice means that the correction process is more likely repairing the original trajectory rather than generating an unrelated solution from scratch.

At the same time, the IBPO framework itself is not tied to token-level edit distance. For code tasks, the comparison operator $\mathcal{M}$ can naturally be replaced or extended with code-specific signals such as AST distance, control-flow-graph similarity, execution-trace similarity, failing-test localization, or compiler-error localization. We use edit distance in this paper to keep the instantiation simple, task-agnostic, and computationally inexpensive; code-specific comparison operators are a direct extension.

Threshold sensitivity analysis. We conduct a sensitivity analysis over the threshold α on Qwen3-32B (AIME25). Results are shown in Table 8.

Threshold α	Flagged as Rewrite (%)	AIME25 (%)
50%	27.45	85.3
55%	23.36	85.5
60%	18.42	85.6
65%	13.37	85.5
70%	9.82	85.4
75%	5.73	85.3
80%	2.06	85.3

Table 8: Rewrite-detection rate and AIME25 accuracy under different edit-distance thresholds α (Qwen3-32B). Performance remains stable in the 55%–70% range.

Adaptive threshold. A possible extension is to choose an adaptive threshold from the batch edit-distance distribution, for example using its mean μ_d and standard deviation σ_d to set $\alpha = \mu_d + 2\sigma_d$.

Suppressing rewrites at the source through reinforcement learning. A possible future extension is to incorporate an edit-distance-dependent correction penalty and contrastive optimization of multiple corrected outputs, assigning higher return to local repairs than to full rewrites.

Positioning. The edit-distance check is a defensive safeguard against reward hacking, not a core component of the IBPO framework. When the model has sufficient base capability, full rewrites are intrinsically rare events, and the specific threshold choice has negligible influence on the main experimental results. The additional rewrite-control mechanisms above are directions for future study.

I.4 Trajectory Diversity and Coupling Reduction

IBPO relies on sampling sufficiently diverse trajectories for the same input. In our experiments, we use the following strategies to increase trajectory diversity and reduce trajectory coupling:

- **Stochastic decoding.** We use different random seeds, temperatures, and top-p/top-k sampling parameters.
- **Prompt perturbation (optional).** We apply slight perturbations to the system prompt or formatting prompt to induce trajectory-level differences.

J Implementation Details

Infrastructure. Experiments are conducted on 32 Nvidia A800 (80G) GPUs.

Optimization. We use an initial learning rate of 5×10^{-7} with cosine decay, a minimum ratio of 0.1, and linear warmup for 3% of the total steps. The entropy-regularization coefficient is set to 0.

Sampling. GSPO uses $G = 64$ rollouts per prompt, while IBPO uses $G = 8$ rollouts to approximately match the overall compute budgets of the methods.

K One Instantiation of IBPO: Integration with GSPO

GSPO is used only as a carrier optimizer; replacing GSPO with GRPO or PPO does not change the IBPO formulation. In the preceding sections, we introduced the general IBPO formulation and its reward-shaping definition. IBPO is a training formulation orthogonal to the underlying sequence-level reinforcement-learning method. This section describes how to integrate this formulation seamlessly into GSPO, a representative sequence-level RL algorithm.

K.1 Preliminaries: Sequence-Level Reinforcement Learning

We view the autoregressive language model parameterized by θ as a policy π_θ. Let $\mathcal{D}$ denote the query set. Given a query x, the model generates a full response $y = (y_1, \ldots, y_{|y|})$, with sequence probability

$$\pi_\theta(y \mid x) = \prod_{t=1}^{|y|} \pi_\theta(y_t \mid x, y_{<t}). \tag{20}$$

We consider a general class of sequence-level policy-optimization objectives:

$$J(\theta) = \mathbb{E}_{x\sim\mathcal{D},\, y\sim\pi_{\theta_{\text{old}}}(\cdot|x)}\Big[\mathcal{L}\big(s(\theta; x, y), \hat{A}(x, y)\big)\Big], \tag{21}$$

where $s(\theta; x, y)$ denotes the sequence-level importance-sampling weight, and $\hat{A}(x, y)$ is constructed from sequence-level rewards. Methods such as GSPO and GRPO can be viewed as specific instantiations of this formulation.

IBPO does not change the optimization form in Eq. (21). Instead, it shapes the original sequence-level reward through the model's self-correction process.

K.2 IBPO and GSPO: Single-Pass Joint Training

At each iteration, we perform a **single** policy update. For the same batch of queries x, we simultaneously construct the candidate response set for sequence-level RL and the self-correction results used to evaluate recoverability. IBPO modifies only the reward definition, shaping it into r', while keeping the GSPO surrogate objective unchanged.

(A) GSPO backbone with shaped rewards. For each query x, sample G responses $\{y_i\}_{i=1}^{G}$ from the old policy. The sequence-level importance ratio in GSPO is defined as

$$s_i(\theta) = \left(\frac{\pi_\theta(y_i \mid x)}{\pi_{\theta_{\text{old}}}(y_i \mid x)} \right)^{\frac{1}{|y_i|}}. \tag{22}$$

We use the **shaped reward** to construct within-group advantages:

$$\hat{A}_i = \frac{r'(x, y_i) - \mu'}{\sigma'}, \tag{23}$$

where μ' and σ' denote the mean and standard deviation of the shaped rewards within the group. The GSPO objective is

$$\begin{aligned} J_{\text{GSPO}}(\theta) = \mathbb{E}_{x \sim \mathcal{D},\, \{y_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(\cdot|x)} \Bigg[\frac{1}{G} \sum_{i=1}^{G} \min\Big(s_i(\theta)\, \hat{A}_i, \\ \text{clip}\big(s_i(\theta),\, 1-\epsilon,\, 1+\epsilon\big)\, \hat{A}_i \Big) \Bigg]. \end{aligned} \tag{24}$$

Masked GSPO variant. For IBPO-mask, we keep the sequence-level advantage $\hat{A}_i$ unchanged and restrict GSPO's sequence-level log-ratio to tokens selected by the mask. Let $\mathbf{m}_i = (m_{i,1}, \ldots, m_{i,|y_i|})$ and $M_i = \max(1, \sum_t m_{i,t})$. Define

$$s_i^{\text{mask}}(\theta) = \exp\left(\frac{1}{M_i} \sum_{t=1}^{|y_i|} m_{i,t} \left[\log \pi_\theta(y_{i,t} \mid x, y_{i,<t}) - \log \pi_{\theta_{\text{old}}}(y_{i,t} \mid x, y_{i,<t}) \right] \right). \tag{25}$$

Then replace $s_i(\theta)$ in the GSPO backbone objective above with $s_i^{\text{mask}}(\theta)$. On the active unclipped branch at $\theta = \theta_{\text{old}}$, the gradient locally reduces to broadcasting the same sequence-level advantage $\hat{A}_i$ to the selected tokens and multiplying by $m_{i,t}$. It introduces neither a token-level value model nor a token-level advantage estimator.

(B) Self-correction shaping signal. To compute the shaped reward $r'(x, y_i)$, we construct a self-correction instance for each incorrect answer y_i. Specifically, we randomly sample a reference answer y_i^{ref} (from correct answers if the group contains at least one correct answer; otherwise from other incorrect answers satisfying $j \neq i$), and ask the model to compare and correct:

$$\hat{y}_i = \mathcal{C}(x; y_i, y_i^{\text{ref}}). \tag{26}$$

This process introduces no additional supervision and is used only to assess whether the model can correct an incorrect answer into a correct one. Combined with the full-rewrite filtering in Eq. (19), we define

$$\Delta_i = \rho \cdot \mathbb{I}\big[r(x, y_i) < 1 \ \wedge\ r(x, \hat{y}_i) = 1 \ \wedge\ \neg\text{rewrite}(y_i, \hat{y}_i, y_i^{\text{ref}}) \big], \qquad \rho = 0.5, \tag{27}$$

and obtain the sequence-level shaped reward:

$$r'(x, y_i) = r(x, y_i) + \lambda\, \Delta_i. \tag{28}$$

Although $r'(x, y_i)$ remains formally a sequence-level scalar, its value depends on the counterfactual self-correction process and therefore implicitly encodes process-level, or step-level, credit-assignment information.

(C) Joint GSPO training for correction behavior. In addition to sequence-level optimization of the original reasoning input x using the implicit process reward $r'(x, y)$, we further treat **correction behavior itself as a reasoning task for the same policy on an extended input space** and train it jointly with the **same GSPO objective**. Formally, this process does not introduce a new Markov decision process (MDP); it simply applies the policy to different conditional inputs, i.e., a prompt-conditioned policy.

Concretely, for each answer y_i whose recoverability needs to be evaluated, we construct a correction input:

$$\tilde{x}_i = (x;\ y_i,\ y_i^{\text{ref}}), \tag{29}$$

where y_i^{ref} denotes the reference answer. Conditioned on this input, the model generates a corrected output and receives a terminal reward according to final-answer correctness. Importantly, correction inputs are usually different: each incorrect answer y_i produces its own $\tilde{x}_i$. Therefore, we **do not** mix corrected outputs from different $\tilde{x}_i$ into the same GSPO group for advantage normalization; instead, each correction input forms its own group.

Specifically, for each correction input $\tilde{x}_i$ enabled for joint training, we sample G_c corrected outputs from the old policy:

$$\{z_{i,\ell}\}_{\ell=1}^{G_c} \sim \pi_{\theta_{\text{old}}}(\cdot \mid \tilde{x}_i).$$

In implementation, the $\hat{y}_i$ generated in stage (B) for computing the shaping reward can be reused as one sample in this correction group. If an additional $G_c - 1$ corrected outputs are needed, their generation cost is fully included in the training budget. The correction reward is defined as

$$r_{i,\ell}^{\text{corr}} = r(x, z_{i,\ell}),$$

which checks whether the corrected output gives the correct final answer to the original problem x. The within-correction-group advantage is

$$\hat{A}_{i,\ell}^{\text{corr}} = \frac{r_{i,\ell}^{\text{corr}} - \mu_i^{\text{corr}}}{\sigma_i^{\text{corr}} + 10^{-8}},$$

where μ_i^{corr} and σ_i^{corr} are computed only over the G_c outputs for the same correction input $\tilde{x}_i$. The auxiliary GSPO objective for correction behavior is:

$$J_{\text{corr}}(\theta) = \mathbb{E}_{\tilde{x}_i}\left[\frac{1}{G_c}\sum_{\ell=1}^{G_c}\min\Big(s_{i,\ell}^{\text{corr}}(\theta)\,\hat{A}_{i,\ell}^{\text{corr}},\ \text{clip}\big(s_{i,\ell}^{\text{corr}}(\theta),\,1-\epsilon,\,1+\epsilon\big)\,\hat{A}_{i,\ell}^{\text{corr}}\Big)\right], \tag{30}$$

where

$$s_{i,\ell}^{\text{corr}}(\theta) = \left(\frac{\pi_\theta(z_{i,\ell} \mid \tilde{x}_i)}{\pi_{\theta_{\text{old}}}(z_{i,\ell} \mid \tilde{x}_i)}\right)^{1/|z_{i,\ell}|}.$$

The final optimization objective is

$$J_{\text{total}}(\theta) = J_{\text{main}}(\theta) + \eta J_{\text{corr}}(\theta),$$

where J_{main} is the GSPO objective in part (A) based on the original input x and shaped reward $r'(x, y)$, and η is the weight of the auxiliary correction objective.

We emphasize that this joint training process **does not introduce an additional optimization stage or a separate optimizer**. Learning correction ability is simply behavioral generalization of the policy under different input conditions, enabling the model to gradually internalize comparison and correction ability during training without requiring additional multi-round calls at inference time.

K.3 IBPO + GSPO Algorithm Pseudocode

Combining the stages above, the full workflow of IBPO + GSPO is summarized in Algorithm 1 and Algorithm 2.

Algorithm 1 IBPO + GSPO Main Training Procedure

Input: Dataset $\mathcal{D}$; policy π_θ and old policy $\pi_{\theta_{\text{old}}}$; group size G; clipping parameter ϵ; shaping weight λ; auxiliary weight η; scale ρ; correction operator $\mathcal{C}$; terminal reward $r(\cdot) \in \{0, 1\}$.
Output: Updated parameters θ.

for each iteration **do**
  Sample a mini-batch of prompts $\mathcal{B} = \{x\}$ from $\mathcal{D}$.
  Initialize batch objective $\mathcal{J}_\mathcal{B}(\theta) \leftarrow 0$.
  for each prompt $x \in \mathcal{B}$ **do**
    Sample G answers $\{y_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(\cdot \mid x)$ and compute terminal rewards $r_i \leftarrow r(x, y_i)$.
    for $i = 1$ to G **do**
      if $r_i = 0$ **then**
        if there exists j such that $r(x, y_j) = 1$ **then**
          Sample reference answer y_i^{ref} uniformly from $\{y_j : r(x, y_j) = 1\}$.
        else
          Sample reference answer y_i^{ref} uniformly from $\{y_j : r(x, y_j) = 0,\ j \neq i\}$.
        end if
        Generate corrected result $\hat{y}_i \leftarrow \mathcal{C}(x; y_i, y_i^{\text{ref}})$.
        $\Delta_i \leftarrow \rho \cdot \mathbb{I}\big[r(x, \hat{y}_i) = 1 \wedge \neg\text{rewrite}(y_i, \hat{y}_i, y_i^{\text{ref}})\big]$.
      else
        $\Delta_i \leftarrow 0$.
      end if
      $r_i' \leftarrow r_i + \lambda\, \Delta_i$, and compute sequence-level ratio $s_i(\theta)$.
    end for
    $\mu' \leftarrow \frac{1}{G} \sum_{i=1}^G r_i'$.
    $\sigma' \leftarrow \sqrt{\frac{1}{G} \sum_{i=1}^G (r_i' - \mu')^2}$.
    for $i = 1$ to G **do**
      $\hat{A}_i \leftarrow \frac{r_i' - \mu'}{\sigma' + 10^{-8}}$.
    end for
    $\mathcal{J}_x(\theta) \leftarrow \frac{1}{G} \sum_{i=1}^G \min\Big(s_i(\theta)\hat{A}_i,\ \text{clip}(s_i(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_i\Big)$.
    If joint training is enabled, compute $\mathcal{J}_x^{\text{corr}}(\theta)$ using Algorithm 2; otherwise set $\mathcal{J}_x^{\text{corr}}(\theta) \leftarrow 0$.
    $\mathcal{J}_\mathcal{B}(\theta) \leftarrow \mathcal{J}_\mathcal{B}(\theta) + \mathcal{J}_x(\theta) + \eta \mathcal{J}_x^{\text{corr}}(\theta)$.
  end for
  Update θ by maximizing $\mathcal{J}_\mathcal{B}(\theta)$.
end for

Algorithm 2 Optional GSPO Auxiliary Training for Correction Behavior

Input: Original prompt x; incorrect-answer set $\{y_i : r_i = 0\}$; reference answer y_i^{ref}; generated correction $\hat{y}_i$; correction group size G_c; clipping parameter ϵ; terminal reward $r(\cdot)$.
Output: Auxiliary correction objective $\mathcal{J}_x^{\text{corr}}(\theta)$.

Initialize $\mathcal{J}_x^{\text{corr}}(\theta) \leftarrow 0$.
for each incorrect answer y_i enabled for joint training **do**
  Construct correction input $\tilde{x}_i \leftarrow (x;\ y_i,\ y_i^{\text{ref}})$.
  Reuse $\hat{y}_i$ as $z_{i,1}$; if $G_c > 1$, sample additional outputs $\{z_{i,\ell}\}_{\ell=2}^{G_c} \sim \pi_{\theta_{\text{old}}}(\cdot \mid \tilde{x}_i)$.
  Count all additional correction sampling, verification, and comparison costs in the training budget.
  for $\ell = 1$ to G_c **do**
    Compute $r_{i,\ell}^{\text{corr}} \leftarrow r(x, z_{i,\ell})$ and sequence-level ratio $s_{i,\ell}^{\text{corr}}(\theta)$.
  end for
  Compute μ_i^{corr}, σ_i^{corr}, and $\hat{A}_{i,\ell}^{\text{corr}}$ within the G_c outputs for the same $\tilde{x}_i$.
  Accumulate $\mathcal{J}_i^{\text{corr}}(\theta)$ according to Eq. (30); different $\tilde{x}_i$ do not share within-group normalization statistics.
  $\mathcal{J}_x^{\text{corr}}(\theta) \leftarrow \mathcal{J}_x^{\text{corr}}(\theta) + \mathcal{J}_i^{\text{corr}}(\theta)$.
end for

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction state the IBPO framework, its scope, and its empirical focus on math and code reasoning. The claims are further qualified in the method, theory, and limitations sections.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The paper includes a dedicated limitations section discussing compute overhead, dependence on reliable terminal verification, verifier noise, and possible mitigations.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [Yes]

Justification: The theoretical analysis states the exchangeability, effective negative correlation, and non-degeneracy conditions, and provides a finite-sample proof for the pre-normalization centered advantage term in the appendix.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: The paper reports datasets, base models, context lengths, decoding settings, rollout counts, optimization hyperparameters, evaluation protocol, baselines, and compute matching details in the main text and appendix.

Guidelines:

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

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

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

   Answer: [Yes]

   Justification: Code is publicly available at `https://github.com/ding-decision-ai/IBPO`.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: The experiments section and appendix specify datasets, models, context lengths, rollout counts, learning rate, schedule, warmup, entropy regularization, batch size, decoding parameters, verifier use, and compute matching.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: Table 1 reports means over five random seeds with 95% bootstrap confidence intervals.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: The experiments section reports the use of 32 Nvidia A800 80G GPUs and describes the training-compute matching procedure; the appendix explains the generation, correction, verification, filtering, and update costs included in the compute budget.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research uses public reasoning benchmarks and standard model fine-tuning/evaluation procedures, and does not involve human subjects, privacy-sensitive data, or the release of a new high-risk dataset.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [Yes]

Justification: The limitations and societal impact sections discuss potential positive and negative impacts. Positive impacts include improving the compute efficiency of reasoning reinforcement learning and achieving stronger performance.

Guidelines:

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The paper does not release a new high-misuse-risk pretrained model, image generator, or scraped dataset; the work focuses on a training method and evaluates it on existing benchmarks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Appendix B lists official license information for public assets including Qwen, MathArena, LiveCodeBench, and vLLM, and cites the corresponding models, datasets, and evaluation tools.

Guidelines:

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

13. **New assets**

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

    Answer: [N/A]

    Justification: The paper does not introduce a new dataset or benchmark.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The work does not involve crowdsourcing experiments or human-subject studies.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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

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

Answer: [N/A]

Justification: The work does not involve crowdsourcing or human-subject research, so IRB approval is not applicable.

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [Yes]

Justification: LLMs are central to the method and experiments. The paper describes the base models, LLM rollout generation, correction generation, and verifier-based training/evaluation setup.

Guidelines:

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