FBOS-RL: FEEDBACK-DRIVEN BI-OBJECTIVE SYNERGISTIC REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning (RL) improves the reasoning capabilities of large-scale models. Group Relative Policy Optimization (GRPO) and its variants alternate between *rollout sampling* and *policy update*, updating the policy using sampled rollouts and their advantages. Unlike supervised learning with explicit ground-truth targets, these policy updates rely on high-quality rollouts as an implicit "teacher" to guide their direction. However, GRPO and similar algorithms sample only from the original prompt. For tasks beyond the policy's current capability, high-quality rollouts are rare, leaving updates without a meaningful gradient direction and causing training to stall. To address this issue, we propose **FBOS-RL**, a *Feedback-Driven Bi-Objective Synergistic Reinforcement Learning* framework. Specifically, we let the model perform Feedback-Guided Exploration Enhancement based on the feedback provided by the environment, and on top of this we design two mutually reinforcing training objectives: *Exploitation-oriented Policy Alignment* (EPA) and *Exploration-oriented Capability Cultivation* (ECC). Extensive experiments show improved rollout efficiency and higher peak validation performance. We match the total number of generated training rollouts with GRPO: GRPO samples 72 responses per original prompt, whereas FBOS-RL allocates the same budget to 8 initial and 64 feedback-guided responses. On TravelPlanner with Qwen3-8B, FBOS-RL achieves a relative gain of $+45.81\%$ in peak final pass rate within the shared rollout budget. To reach GRPO's best final pass rate, GRPO requires approximately $3.29\times$ as many training rollouts as FBOS-RL. Additional experiments compare against On-Policy Self-Distillation (OPSD). FBOS-RL also exhibits higher policy entropy and lower gradient norms than GRPO throughout training. Code will be made publicly available.

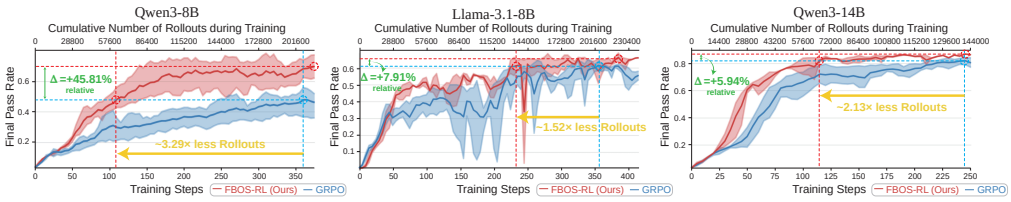


Figure 1: Final pass rate evaluated during training on 1,000 TravelPlanner test samples: **FBOS-RL** (red) and vanilla GRPO (blue). Bottom/top x-axes show training steps/cumulative rollouts during training. Solid curves and shaded bands denote the mean and min–max range over three independent runs. The *horizontal* dashed lines mark each method's best (peak) final pass rate (gap = relative gain Δ, up to $+45.81\%$); the *vertical* dashed lines mark the cumulative rollouts during training at which each method first reaches GRPO's best (ratio = rollout speedup, up to $3.29\times$ fewer). Under both an identical total number of rollouts during training and the same number of training steps, FBOS-RL attains a higher ceiling *and* reaches GRPO's best with far fewer training rollouts.

1 INTRODUCTION

Reinforcement learning (RL) plays a pivotal role in the alignment and reasoning fine-tuning of large-scale models (Ouyang et al., 2022; Bai et al., 2022; Rafailov et al., 2023; Guo et al., 2025; Shao et al., 2024). In essence, the training process of Group Relative Policy Optimization (GRPO) and its variants alternates between two stages, *rollout sampling* and *policy update* (Schulman et al., 2017;

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Shao et al., 2024; Tang et al., 2025): the model first samples in its action space to draw multiple rollouts, and then updates its policy parameters according to these rollouts and their advantages. Unlike supervised learning, where the model is directly updated toward an explicit ground truth, in this setting the model does not know the optimal gradient direction a priori when its parameters are updated. The high-quality rollouts that the model happens to draw during the sampling stage in fact play the role of a "teacher" that guides the parameter update. Only when the model samples high-quality rollouts can the parameter update obtain a better gradient direction, thereby driving the model's capability forward.

GRPO and its variants (Shao et al., 2024; Yu et al., 2025; Guo et al., 2025) commonly condition rollout sampling solely on the original prompt. Before policy optimization, vanilla GRPO samples $q \sim \mathcal{D}$ and then rollouts from the pre-update policy $\pi_{\theta_{\text{old}}}(\cdot \mid q)$ (Shao et al., 2024). This involves two distributions: the task distribution $\mathcal{D}$ and the conditional rollout distribution $\pi_{\theta_{\text{old}}}(\cdot \mid q)$. When high-quality rollouts have low probability under the latter, they may be difficult to discover within a finite sampling budget. In this situation, a single scalar reward can only tell the model that its current policy "performs poorly", but cannot indicate "why" it is poor or "how to improve" (akin to a student drilling problems without any explanation of past mistakes, whose progress is slow) (Huang et al., 2023). Because the model cannot obtain high-quality rollouts as optimization anchors during the sampling stage, it loses a reliable gradient direction when its parameters are updated, which eventually leads to low training efficiency and even prolonged training stagnation (Yue et al., 2025; Cui et al., 2025b; Liu et al., 2025).

To address this issue, we propose **FBOS-RL**: **F**eedback-Driven **B**i-**O**bjective **S**ynergistic **R**einforcement **L**earning. FBOS-RL retains the original task distribution while using feedback to expand the conditioning contexts for rollout exploration beyond the original prompts. On top of this enhanced sampling stage, we design two mutually promoting training objectives, namely *Exploitation-oriented Policy Alignment* (EPA) and *Exploration-oriented Capability Cultivation* (ECC). By alternately optimizing these two objectives, FBOS-RL forms a positive bootstrapping flywheel, thereby substantially improving both rollout efficiency and the best performance attained within the evaluated budget. Although several recent works have attempted to introduce external feedback during the sampling stage, most of them merely treat feedback as a means of data augmentation, that is, they use feedback to generate higher-quality data and then let the model imitate it (Madaan et al., 2023; Shinn et al., 2023; Kumar et al., 2024; Qu et al., 2024), while overlooking the cultivation of the model's own capability to sample better rollouts according to feedback. In contrast, FBOS-RL not only leverages feedback to enhance the quality of the rollouts obtained during the sampling stage, but also explicitly sets "understanding feedback and correcting errors" itself as an additional reinforcement learning objective. The two objectives are optimized alternately to improve rollout efficiency and validation performance. As illustrated in Figure 1, GRPO requires approximately $3.29\times$, $1.52\times$, and $2.13\times$ as many generated training rollouts as FBOS-RL to reach GRPO's best Final Pass Rate on Qwen3-8B, Llama-3.1-8B-Instruct, and Qwen3-14B, respectively. Separately, within the same maximum rollout budget for each backbone, FBOS-RL improves the peak Final Pass Rate by $+45.81\%$, $+7.91\%$, and $+5.94\%$, respectively. Both comparisons count the initial and feedback-guided rollouts; Section 4.1 specifies the shared budget. Beyond these main results, FBOS-RL also exhibits stronger exploration and better training stability: as shown in Figure 13 and Figure 14 in Appendix H, FBOS-RL does not suffer from entropy collapse and consistently maintains a higher policy entropy than vanilla GRPO throughout training, while at the same time exhibiting a lower gradient norm.

Our contributions are summarized as follows:

- We point out the inefficiency and blindness of mainstream reinforcement learning algorithms (such as GRPO) during the rollout-sampling stage, and propose an interactive sampling mechanism, namely Feedback-Guided Exploration Enhancement, which effectively breaks the inefficient-sampling bottleneck on complex reasoning tasks.

- We propose the **Feedback-Driven Bi-Objective Synergistic Reinforcement Learning (FBOS-RL)** framework. By designing two mutually promoting training objectives (Exploitation-oriented Policy Alignment and Exploration-oriented Capability Cultivation), it forms a positive bootstrapping flywheel that substantially improves both rollout efficiency and the best performance attained within the evaluated budget.

- Extensive experiments on different datasets and on models of different families and scales validate that FBOS-RL can substantially improve both rollout efficiency and the best performance attained within the evaluated budget. On TravelPlanner with Qwen3-8B, FBOS-RL improves peak Final Pass Rate by 45.81% relative to vanilla GRPO under the same maximum rollout budget, while reaching GRPO's peak with approximately $3.29\times$ fewer training rollouts. Meanwhile, FBOS-RL also avoids entropy collapse, maintains higher policy entropy, and exhibits a lower gradient norm, indicating its stronger exploration capability and better training stability.

2 Related works

Reinforcement Learning for LLM Reasoning. LLM alignment (Ouyang et al., 2022; Bai et al., 2022; Rafailov et al., 2023) and RL reasoning (Schulman et al., 2017; Shao et al., 2024; Yu et al., 2025; Guo et al., 2025) benefit from process supervision (Lightman et al., 2023; Wang et al., 2024; Cui et al., 2025a), prolonged training (Liu et al., 2025), and entropy-aware optimization (Cui et al., 2025b). Exploration limitations (Yue et al., 2025; Tang et al., 2025; Wang et al., 2025b) motivate curricula (Xi et al., 2024), tree search (Zheng et al., 2025), negative-data learning (Setlur et al., 2024), test-time scaling (Snell et al., 2024), and self-play/self-rewarding (Chen et al., 2024; Yuan et al., 2024).

Feedback-Driven Self-Refinement and Self-Correction. Feedback-based refinement (Madaan et al., 2023; Shinn et al., 2023; Saunders et al., 2022; Welleck et al., 2023; Gou et al., 2024) supports output revision, while SCoRe (Kumar et al., 2024) and recursive introspection (Qu et al., 2024) train correction. External grounding remains important (Huang et al., 2023). LUFFY (Yan et al., 2025) transfers off-policy guidance through importance-ratio reweighting.

On-Policy Distillation and Self-Distillation. On-policy distillation (OPD) (Agarwal et al., 2024), introduced in GKD, trains a student on its own rollouts using a teacher's token-level distributions, reducing training–inference mismatch. On-policy self-distillation (OPSD) (Zhao et al., 2026) is a special case: teacher and student originate from the same base model, but only the teacher receives privileged reference solutions. FBOS-RL instead uses verifier feedback to sample additional rollouts and jointly trains feedback-free alignment and feedback-conditioned exploration through reward-based EPA and ECC.

3 Method

In this section, we introduce FBOS-RL, the Feedback-Driven Bi-Objective Synergistic Reinforcement Learning framework, whose overall pipeline is illustrated in Figure 2. FBOS-RL performs reinforcement learning on a single policy model π_θ and does not rely on any external model. Its two mutually reinforcing objectives improve rollout efficiency and the best performance attained within the evaluated budget. Each training step denotes a complete sampling-and-optimization iteration, comprising *Initial Exploration*, *Feedback-Guided Exploration Enhancement*, and *Bi-Objective Synergistic Training*; the last stage contains the EPA and ECC updates. The complete procedure is summarized in Algorithm 1 in Appendix A. Let $q \sim \mathcal{D}$ be a prompt and π_θ the policy. The behavior snapshot θ_{old} is fixed before sampling and throughout both updates at each training step. For a response a, a_t and $a_{<t}$ denote its token at position t and preceding prefix; $\epsilon > 0$ ensures numerical stability, and ε is the clipping radius.

3.1 Initial Exploration

In the *Initial Exploration* stage, corresponding to the "Step 1: Initial Exploration" block in Figure 2, the policy model π_θ independently samples, in parallel, n initial rollouts $\{\text{ans}_i^1\}_{i=1}^n \sim \pi_\theta(\cdot \mid q)$ conditioned solely on the original prompt q. Each rollout ans_i^1 with $i = 1, \ldots, n$ is then evaluated by a rule-based verifier that returns both a scalar reward r_i^1 and a natural-language feedback F_i^1 that explicitly identifies the flaws in the current answer (e.g., a specific reasoning step that violates a constraint, an incorrect intermediate result, or a violation of the required output format). Unlike a single scalar reward, F_i^1 provides actionable, error-localized information that tells the model not only that ans_i^1 is suboptimal, but also *why* and *where* it went wrong.

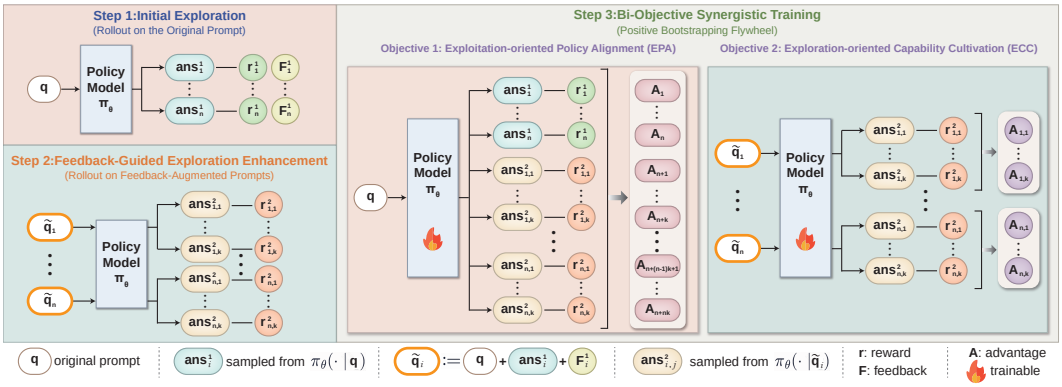


Figure 2: Overview of our FBOS-RL framework. In the sampling phase, the policy first samples n initial rollouts $\{\text{ans}_i^1\}_{i=1}^n$ from $\pi_\theta(\cdot \mid q)$. For each rollout ans_i^1, a rule-based verifier produces natural-language feedback F_i^1. The original prompt q, rollout ans_i^1, and feedback F_i^1 are concatenated to form a Feedback-Augmented Prompt (FAP), $\tilde{q}_i := (q \oplus \text{ans}_i^1 \oplus F_i^1)$, for second-round sampling from $\pi_\theta(\cdot \mid \tilde{q}_i)$. In the optimization phase, the shared policy π_θ is trained sequentially with two complementary objectives: Exploitation-oriented Policy Alignment (EPA), which optimizes $\pi_\theta(\cdot \mid q)$, and Exploration-oriented Capability Cultivation (ECC), which optimizes $\pi_\theta(\cdot \mid \tilde{q}_i)$. Together, the two objectives induce a positive bootstrapping flywheel.

3.2 Feedback-Guided Exploration Enhancement

To enhance the quality of the rollouts sampled during the sampling stage, so that the model can obtain higher-quality gradient guidance when its parameters are updated in the subsequent training phase, we proceed from the Initial Exploration stage to the *Feedback-Guided Exploration Enhancement* stage, corresponding to the "Step 2: Feedback-Guided Exploration Enhancement" block in Figure 2. For each initial rollout ans_i^1, we concatenate the original prompt q, the rollout itself, and its corresponding feedback F_i^1 to construct a *Feedback-Augmented Prompt (FAP)*: $\tilde{q}_i := (q \oplus \text{ans}_i^1 \oplus F_i^1)$. The FAP is then used to re-prompt the policy model for a second round of sampling: for each $\tilde{q}_i$, the model samples k rollouts $\{\text{ans}_{i,j}^2\}_{j=1}^k \sim \pi_\theta(\cdot \mid \tilde{q}_i)$, and each $\text{ans}_{i,j}^2$ is again scored by the same rule-based verifier to obtain a reward $r_{i,j}^2$.

In total, the two stages produce $N := n(1+k)$ new rollouts per original query: n from $\pi_{\theta_{\text{old}}}(\cdot \mid q)$ and nk from $\pi_{\theta_{\text{old}}}(\cdot \mid \tilde{q}_i)$. In our rollout-matched comparisons, GRPO's group size is set to $N_{\text{G}} = N = n(1+k)$: it samples all N responses from q, whereas FBOS-RL distributes the same budget across the two stages. Both stages count toward the sampling budget; ECC subsequently reuses the second-round responses without generating new rollouts.

From a distributional perspective, FBOS-RL retains $q \sim \mathcal{D}$ while expanding the conditioning contexts used for exploration from q to the feedback-augmented prompts $\tilde{q}_i$. Both rounds use the same frozen θ_{old}, so feedback changes the conditional rollout distribution before any policy update. This allows the model to explore under targeted guidance when high-quality rollouts are difficult to discover from $\pi_{\theta_{\text{old}}}(\cdot \mid q)$ within a finite sampling budget. EPA then transfers the reward-weighted learning signals from these feedback-guided rollouts to the original-prompt policy $\pi_\theta(\cdot \mid q)$, while ECC trains the feedback-conditioned policy $\pi_\theta(\cdot \mid \tilde{q}_i)$ to make better use of feedback.

3.3 Bi-Objective Synergistic Training

Building upon the two sampling stages described above, we design two mutually reinforcing training objectives, namely *Exploitation-oriented Policy Alignment* (EPA) and *Exploration-oriented Capability Cultivation* (ECC), as illustrated in the "Step 3: Bi-Objective Synergistic Training" block in Figure 2. At each training step, we sequentially update the policy parameters using these two objectives. In the objectives below, $\mathbb{E}$ is taken over $q \sim \mathcal{D}$, the initial rollouts sampled from $\pi_{\theta_{\text{old}}}(\cdot \mid q)$, the resulting verifier feedback and FAPs, and the second-round rollouts sampled from $\pi_{\theta_{\text{old}}}(\cdot \mid \tilde{q}_i)$.

In this paper, the qualifiers "exploitation-oriented" and "exploration-oriented" carry operational meanings specific to our framework. By "exploitation-oriented" we mean exploiting the high-quality rollouts already collected as a learning signal for policy alignment. By "exploration-oriented" we mean improving the sampling capability of $\pi_\theta(\cdot \mid \tilde{q}_i)$. These usages are scoped to GRPO-style

training and are not intended to coincide with the classical action-selection trade-off in textbook reinforcement learning.

Objective 1: Exploitation-oriented Policy Alignment (EPA).
EPA updates the shared policy under the original prompt, $\pi_\theta(\cdot \mid q)$. The second-round rollouts provide additional training signals, but their current-policy likelihoods for EPA are evaluated without feedback under q. We aggregate the n initial rollouts $\{\mathrm{ans}_i^1\}_{i=1}^n$ (drawn from $\pi_{\theta_{\text{old}}}(\cdot \mid q)$) and the $n \cdot k$ rollouts $\{\mathrm{ans}_{i,j}^2\}_{i=1,j=1}^{n,k}$ produced in the Feedback-Guided Exploration Enhancement stage (drawn from $\pi_{\theta_{\text{old}}}(\cdot \mid \tilde{q}_i)$) into a single group $\mathcal{G}_q \coloneqq \{\mathrm{ans}_i^1\}_{i=1}^n \cup \{\mathrm{ans}_{i,j}^2\}_{i=1,j=1}^{n,k}$, of size $N \coloneqq n + n \cdot k$. Based on the rewards $\mathcal{R}_q \coloneqq \{r_i^1\}_{i=1}^n \cup \{r_{i,j}^2\}_{i=1,j=1}^{n,k}$ associated with these rollouts, we compute a group-normalized advantage following GRPO (Shao et al., 2024) for every rollout $a \in \mathcal{G}_q$ and reward $r(a) \in \mathcal{R}_q$ as

$$\hat{A}^{\text{EPA}}(a) \;=\; \frac{r(a) - \mu_{\mathcal{G}_q}}{\sigma_{\mathcal{G}_q} + \epsilon},$$
$$\text{with} \quad \mu_{\mathcal{G}_q} = \frac{1}{N}\sum\nolimits_{a' \in \mathcal{G}_q} r(a'), \quad \sigma^2_{\mathcal{G}_q} = \frac{1}{N}\sum\nolimits_{a' \in \mathcal{G}_q} \big(r(a') - \mu_{\mathcal{G}_q}\big)^2.$$

For an initial rollout ans_i^1 drawn from $\pi_{\theta_{\text{old}}}(\cdot \mid q)$, we use the standard GRPO same-prompt ratio, with the superscript 1 indicating the first sampling stage:

$$\rho_{i,t}^1(\theta) = \frac{\pi_\theta(\mathrm{ans}_{i,t}^1 \mid q, \mathrm{ans}_{i,<t}^1)}{\pi_{\theta_{\text{old}}}(\mathrm{ans}_{i,t}^1 \mid q, \mathrm{ans}_{i,<t}^1)}.$$

However, for a rollout $\mathrm{ans}_{i,j}^2$ collected in the Feedback-Guided Exploration Enhancement stage, the behavior policy that actually drew it is $\pi_{\theta_{\text{old}}}(\cdot \mid \tilde{q}_i)$, whereas the policy we ultimately deploy at inference time only sees the original prompt q. To transfer learning signals from these feedback-guided samples to the feedback-free policy, we introduce the following token-level cross-context likelihood ratio as a practical surrogate:

$$\rho_{i,j,t}^2(\theta) \;=\; \frac{\pi_\theta(\mathrm{ans}_{i,j,t}^2 \mid q,\, \mathrm{ans}_{i,j,<t}^2)}{\pi_{\theta_{\text{old}}}(\mathrm{ans}_{i,j,t}^2 \mid \tilde{q}_i,\, \mathrm{ans}_{i,j,<t}^2)}.$$

Inspired by Yan et al. (Yan et al., 2025), we additionally introduce a reweighting function $f(\rho) = \rho/(\rho + 0.1)$ that acts on $\rho_{i,j,t}^2(\theta)$, in order to strengthen the learning signal on low-probability tokens that the model has not yet mastered but that may correspond to critical reasoning steps. We then treat $\mathcal{G}_q$ as a single group and perform GRPO training over it. Putting the above ingredients together, and applying the reweighting function $f(\cdot)$ exclusively to $\rho_{i,j,t}^2(\theta)$, the EPA objective is given by

$$\mathcal{L}_{\text{EPA}}(\theta) \;=\; \mathcal{L}_{\text{EPA}}^{\text{init}}(\theta) \;+\; \mathcal{L}_{\text{EPA}}^{\text{FAP}}(\theta), \tag{3.1}$$

where $\mathcal{L}_{\text{EPA}}^{\text{init}}(\theta)$ accounts for the n initial rollouts drawn from $\pi_{\theta_{\text{old}}}(\cdot \mid q)$ and $\mathcal{L}_{\text{EPA}}^{\text{FAP}}(\theta)$ accounts for the $n \cdot k$ rollouts drawn from $\pi_{\theta_{\text{old}}}(\cdot \mid \tilde{q}_i)$, respectively,

$$\mathcal{L}_{\text{EPA}}^{\text{init}}(\theta) = -\,\mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{n}\frac{1}{|\mathrm{ans}_i^1|}\sum_{t=1}^{|\mathrm{ans}_i^1|}\min\Big(\rho_{i,t}^1(\theta)\,\hat{A}^{\text{EPA}}(\mathrm{ans}_i^1),\ \mathrm{clip}_{\text{init}}\hat{A}^{\text{EPA}}(\mathrm{ans}_i^1)\Big)\right],$$
$$\mathcal{L}_{\text{EPA}}^{\text{FAP}}(\theta) = -\,\mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{n}\sum_{j=1}^{k}\frac{1}{|\mathrm{ans}_{i,j}^2|}\sum_{t=1}^{|\mathrm{ans}_{i,j}^2|} f\big(\rho_{i,j,t}^2(\theta)\big)\,\hat{A}^{\text{EPA}}(\mathrm{ans}_{i,j}^2)\right].$$

Note that $\mathrm{clip}_{\text{init}} = \mathrm{clip}\big(\rho_{i,t}^1(\theta),\, 1-\varepsilon,\, 1+\varepsilon\big)$. Appendix M provides exact token-level and finite-batch FAP gradients with step-by-step proofs. The essence of EPA is therefore to pull the model's *feedback-free* policy $\pi_\theta(\cdot \mid q)$ toward the high-quality rollouts discovered during the feedback-guided sampling stage, thereby achieving highly efficient exploitation of the collected high-quality rollouts. We empirically verify that EPA can be optimized in a stable manner: as shown in Appendices P, on both Llama-3.1-8B-Instruct and Qwen3-14B, the EPA training-set score steadily rises while its std steadily decreases, and the same trend holds across training subsets of different difficulty levels.

Objective 2: Exploration-oriented Capability Cultivation (ECC).
ECC updates the same shared policy under each FAP, $\pi_\theta(\cdot \mid \tilde{q}_i)$. To improve the quality of rollouts sampled from $\pi_\theta(\cdot \mid \tilde{q}_i)$, we treat the k rollouts generated in the second round for *each* specific FAP $\tilde{q}_i$ as an independent group $\mathcal{G}_{\tilde{q}_i} = \{\text{ans}^2_{i,j}\}_{j=1}^k$ and perform standard GRPO training. This objective focuses exclusively on cultivating the model's **feedback-conditioned sampling capability**, i.e., its ability to sample better rollouts from $\pi_\theta(\cdot \mid \tilde{q}_i)$.

Based on the rewards $\{r^2_{i,j}\}_{j=1}^k$ associated with these rollouts, we compute a group-normalized advantage for each rollout $a \in \mathcal{G}_{\tilde{q}_i}$ following GRPO (Shao et al., 2024) as

$$\hat{A}^{\text{ECC}}(a) \;=\; \frac{r(a) - \mu_{\tilde{q}_i}}{\sigma_{\tilde{q}_i} + \epsilon}, \qquad \text{with} \quad \mu_{\tilde{q}_i} = \frac{1}{k}\sum\nolimits_{a' \in \mathcal{G}_{\tilde{q}_i}} r(a'), \quad \sigma^2_{\tilde{q}_i} = \frac{1}{k}\sum\nolimits_{a' \in \mathcal{G}_{\tilde{q}_i}} \big(r(a') - \mu_{\tilde{q}_i}\big)^2.$$

Note that $r(a)$ denotes the reward corresponding to the rollout a. Because every rollout in $\mathcal{G}_{\tilde{q}_i}$ shares the same conditioning prompt $\tilde{q}_i$, the standard GRPO importance sampling ratio applies without modification: $\tilde{\rho}_{i,j,t}(\theta) \;=\; \frac{\pi_\theta(\text{ans}^2_{i,j,t} \mid \tilde{q}_i,\, \text{ans}^2_{i,j,<t})}{\pi_{\theta_{\text{old}}}(\text{ans}^2_{i,j,t} \mid \tilde{q}_i,\, \text{ans}^2_{i,j,<t})}$. The ECC objective is defined as follows:

$$\mathcal{L}_{\text{ECC}}(\theta) = -\,\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{k}\sum_{j=1}^{k}\frac{1}{|\text{ans}^2_{i,j}|}\sum_{t=1}^{|\text{ans}^2_{i,j}|}\min\Big(\tilde{\rho}_{i,j,t}(\theta)\,\hat{A}^{\text{ECC}}(\text{ans}^2_{i,j}),\; \text{clip}_{\text{ECC}}\hat{A}^{\text{ECC}}(\text{ans}^2_{i,j})\Big)\right], \tag{3.2}$$

where $\text{clip}_{\text{ECC}} = \text{clip}\big(\tilde{\rho}_{i,j,t}(\theta),\, 1-\varepsilon,\, 1+\varepsilon\big)$. By optimizing $\mathcal{L}_{\text{ECC}}$, ECC improves the model's ability to sample higher-quality rollouts from $\pi_\theta(\cdot \mid \tilde{q}_i)$, so that it learns how to better correct its mistakes according to the feedback. We empirically verify that ECC can be optimized in a stable manner as well: as shown in Appendices Q, on both Llama-3.1-8B-Instruct and Qwen3-14B, the ECC training-set score steadily rises while its std steadily decreases, and the same trend holds across training subsets of different difficulty levels. The two objectives are closely analogous to how a student tackles practice problems: ECC corresponds to the act of refining answers based on the explanations of past mistakes, while EPA corresponds to gradually internalizing those lessons into the realization that "ah, so next time I encounter this problem, I should solve it this way!" Together, they enable the student to learn more efficiently and make rapid progress.

3.4 The Positive Bootstrapping Flywheel Effect

Although EPA and ECC update the same parameters θ, they act on different conditional distributions: EPA optimizes $\pi_\theta(\cdot \mid q)$, whereas ECC optimizes $\pi_\theta(\cdot \mid \tilde{q}_i)$. ECC improves this feedback-conditioned sampling capability, allowing $\pi_\theta(\cdot \mid \tilde{q}_i)$ to supply useful higher-reward rollouts. EPA incorporates these rollouts into $\mathcal{G}_q$ and transfers their reward-weighted learning signals to $\pi_\theta(\cdot \mid q)$. A possible mechanism for these benefits is the progression of errors and feedback across training. By improving $\pi_\theta(\cdot \mid q)$, EPA enables the initial rollouts to overcome some low-level errors. Their remaining errors may then involve deeper logical or constraint-related failures, prompting the verifier to produce F_i^1 that targets these more advanced errors. The informal mechanism is discussed in Appendix B; the two directions are empirically evaluated in Appendices N.1 and N.2.

4 Experiments

4.1 Experimental Setup

Datasets. We conduct experiments on the TravelPlanner dataset (Xie et al., 2024) and the MiniF2F dataset (Zheng et al., 2022) (Lean4 version), respectively. The TravelPlanner dataset evaluates the ability of LLMs to perform complex, multi-constraint planning in real-world scenarios, while MiniF2F-Lean4 benchmarks their capacity for formal mathematical reasoning and proof generation within the Lean 4 formalization environment. Detailed descriptions of both datasets are provided in Appendix C. For the TravelPlanner dataset, we use its training set (45 samples) and validation set (180 samples), totaling 225 samples, as our training set, and adopt its test set (1000 samples) as our validation set. For the MiniF2F-Lean4 dataset, we use its validation set (244 samples) as our training set, and use its test set (244 samples) as our validation set.

Models On the TravelPlanner dataset, we conduct experiments on Qwen3-8B, Llama-3.1-8B-Instruct, and Qwen3-14B. On the MiniF2F-Lean4 dataset, we conduct experiments on Qwen3.5-27B. All experiments involving Qwen3-8B are conducted on NVIDIA H100 GPUs, while all other experiments are conducted on NVIDIA H200 GPUs.

Baseline Methods. We compare FBOS-RL with GRPO (Shao et al., 2024), OPSD (On-Policy Self-Distillation) (Zhao et al., 2026), FBOS-RL w/o EPA, FBOS-RL w/o ECC, GRPO w/ Extra Update, FBOS-RL w/o Cross-Prompt IS Ratio, FBOS-RL w/o IS Ratio Transformation, and FBOS-RL w/ Modified IS Ratio Transformation. The TravelPlanner baselines also include Greedy Search, Direct, CoT, ReAct, Reflexion, and prompt reflect. Detailed definitions of all baseline methods are provided in Appendix E.

Rollout budget. GRPO and FBOS-RL use the same original-prompt order and $N = 72$ newly generated responses per prompt per iteration. GRPO draws all 72 from q; FBOS-RL uses $n = k = 8$, allocating $8 + 64 = 72$ draws across its two sampling stages. With $b = 8$ original prompts per batch, both generate $bN = 576$ training rollouts per iteration, so the cumulative count after t iterations is $C(t) = 576t$. Appendix R gives the training settings.

New responses per original prompt	From q	From FAPs	Total
GRPO	72	0	72
FBOS-RL	8	64	72

We independently repeat the GRPO and FBOS-RL experiments three times. Their solid curves show pointwise arithmetic means, and shaded regions span the pointwise minimum and maximum across runs.

Evaluation Metrics. We adopt standard metrics (Xie et al., 2024) to evaluate the performance of our method, including Commonsense Constraint Pass Rate, Hard Constraint Pass Rate, and Final Pass Rate. Detailed mathematical definitions of all the above metrics are provided in Appendix D. For each sample in the MiniF2F-Lean4 dataset, we adopt the following evaluation scheme: if the model gives a correct proof, it is awarded $+1$ point; if the model gives a response in the correct format but the proof is wrong or some steps are not finished, it is awarded 0 point; if the model's response is in an incorrect format, it is awarded -1 point. We apply the above evaluation scheme to every sample, and finally average the scores as the average score.

4.2 Main Results

Rollout Efficiency and Best Performance within the Shared Budget. We distinguish *rollout efficiency*, measured by the cumulative generated training responses needed to reach a target validation score, from *best performance within a shared rollout budget*. Table 1 compares TravelPlanner performance with prompting-based baselines and GRPO on the same backbones. FBOS-RL achieves a *Final Pass Rate* of **69.65** on Qwen3-8B, **66.15** on Llama-3.1-8B-Instruct, and **86.9** on Qwen3-14B, compared with 47.77, 61.3, and 82.03 for GRPO, respectively. These correspond to relative improvements of approximately $+45.81\%$, $+7.91\%$, and $+5.94\%$.

Figure 3 plots validation trajectories against training iterations t and cumulative training rollouts $C(t) = 576t$, counting both FBOS-RL sampling stages.

For equal-performance comparisons, we measure the first crossing of GRPO's peak final pass rate on each method's pointwise mean validation curve. GRPO requires approximately $3.29\times$, $1.52\times$, and $2.13\times$ as many training rollouts as FBOS-RL on Qwen3-8B, Llama-3.1-8B-Instruct, and Qwen3-14B, respectively. The Qwen3-8B estimate linearly interpolates between the recorded validation points surrounding the target. Under the same maximum cumulative rollout budget per backbone, FBOS-RL also attains higher peaks (Table 1). Rollout efficiency does not establish a wall-clock speedup.

The Easy, Medium, and Hard results and the Commonsense Constraint Pass Rate (Micro/Macro) and Hard Constraint Pass Rate (Micro/Macro) curves are provided in Appendix G, Figures 11–12.

Under identical feedback, FBOS-RL also substantially outperforms our OPSD adaptation on Llama-3.1-8B-Instruct. Appendix F reports the validation curves and OPSD training diagnostics.

Table 1: Experimental results comparing FBOS-RL with different baselines on TravelPlanner (1,000 evaluation samples) with five criteria.

Methods	**Commonsense Constraint**		**Hard Constraint**		**Final Pass Rate**
	Micro	Macro	Micro	Macro	
Greedy Search	72.0	0	52.4	31.8	0
Mixtral-8x7B-MoE$_{\text{Direct}}$	67.0	3.7	3.9	1.6	0.7
GPT-3.5-Turbo$_{\text{Direct}}$	59.5	2.7	9.5	4.4	0.6
GPT-3.5-Turbo$_{\text{CoT}}$	64.4	2.3	9.8	3.8	0.4
GPT-3.5-Turbo$_{\text{ReAct}}$	45.9	2.5	10.7	3.1	0.7
GPT-3.5-Turbo$_{\text{Reflexion}}$	52.1	2.2	9.9	3.8	0.6
GPT-4-Turbo$_{\text{Direct}}$	80.6	15.2	44.3	23.1	4.4
GPT-4o-2024-11-20$_{\text{CoT}}$	84.18	25.9	49.87	26.6	7
Gemini Pro$_{\text{Direct}}$	64.7	7.9	10.6	4.7	2.1
DeepSeek-R1$_{\text{CoT}}$	92.65	55.2	74.15	62	40
Qwen3-8B	66.21	6.73	23.77	17.12	3.43
Qwen3-8B$_{\text{CoT}}$	72.04	10.7	28	21.8	5.9
Qwen3-8B$_{\text{prompt reflect}}$	73.84	14.6	27.95	24	8.3
Qwen3-8B$_{\text{GRPO}}$	**96.43**	73.8	78.71	62.3	47.77
Qwen3-8B$_{\text{FBOS-RL}}$	93.57	**87.85**	**82.21**	**73.15**	**69.65**
Llama-3.1-8B-Instruct	57.12	1.13	6.75	3.53	0.43
Llama-3.1-8B-Instruct$_{\text{GRPO}}$	98.3	90.33	81.65	66.27	61.3
Llama-3.1-8B-Instruct$_{\text{FBOS-RL}}$	**98.58**	**92.65**	**82.84**	**69.4**	**66.15**
Qwen3-14B	66.62	6.13	19.03	15.5	2.9
Qwen3-14B$_{\text{GRPO}}$	97.51	90.33	92.88	88.63	82.03
Qwen3-14B$_{\text{FBOS-RL}}$	**98.7**	**91.3**	**95.94**	**93.8**	**86.9**

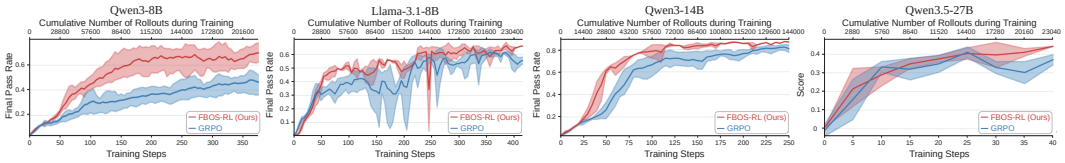


Figure 3: Validation performance of FBOS-RL and vanilla GRPO. From left to right: Final Pass Rate on TravelPlanner with Qwen3-8B, Llama-3.1-8B-Instruct, and Qwen3-14B, followed by validation score on MiniF2F-Lean4 with Qwen3.5-27B. Bottom: training iterations t; top: cumulative generated training rollouts $C(t) = 576t$. Both methods use 72 responses per original prompt (GRPO: 72; FBOS-RL: $8+64$) and 8 prompts per iteration. Both FBOS-RL sampling stages are counted; optimization reuse adds no rollouts.

Stronger Exploration and Better Training Stability.

Beyond the validation accuracy itself, we further measure the actor policy entropy and the gradient norm during training.

Stronger exploration. GRPO can be prone to the entropy collapse phenomenon (Cui et al., 2025b; Yu et al., 2025; Wang et al., 2025a), in which the actor's output distribution rapidly contracts onto a narrow set of high-probability actions during training, causing the policy to become overly deterministic and to lose the capacity to discover new, potentially higher-quality rollouts. Figure 13 in Appendix H compares four settings: Qwen3-8B, Llama-3.1-8B-Instruct, and Qwen3-14B on TravelPlanner, and Qwen3.5-27B on MiniF2F-Lean4. FBOS-RL generally maintains higher actor entropy than vanilla GRPO, with a particularly clear separation on Qwen3-8B and Qwen3-14B over most of training. These observations are consistent with FBOS-RL preserving broader output distributions during training.

Better training stability. We further examine the gradient norm during training. As shown in Figure 14 in Appendix H, across the four settings, including Qwen3-8B on TravelPlanner, FBOS-RL exhibits a lower gradient norm than vanilla GRPO for most training steps. This observation is consistent with more stable optimization dynamics under the same training settings.

Out-of-Distribution Generalization. We further evaluate the GPQA-Diamond performance of each model before training and after FBOS-RL training, with results shown in Figure 15 in Appendix I. All

models use greedy decoding during inference. The Llama-3.1-8B-Instruct and Qwen3-14B models are trained only on TravelPlanner with FBOS-RL, while the Qwen3.5-27B model is trained only on MiniF2F-Lean4 with FBOS-RL; none of these models are trained on GPQA-Diamond. Compared with the untrained Llama-3.1-8B-Instruct model, the Llama-3.1-8B-Instruct model trained with FBOS-RL on TravelPlanner improves by 10.1 percentage points on GPQA-Diamond, corresponding to a relative improvement of 76.92%. For Qwen3-14B, training with FBOS-RL on TravelPlanner does not lead to any performance degradation on GPQA-Diamond. For Qwen3.5-27B, training with FBOS-RL on MiniF2F-Lean4 improves GPQA-Diamond accuracy by 4.54 percentage points over the untrained Qwen3.5-27B model, corresponding to a relative improvement of 8.99%.

4.3 Ablation Studies

To demonstrate that training Objective 1 (Exploitation-oriented Policy Alignment, EPA) and Objective 2 (Exploration-oriented Capability Cultivation, ECC) can mutually reinforce each other, forming a positive flywheel (bootstrapping) effect, we conduct in-depth experiments on the Qwen3-14B model. Figure 4 shows that our method significantly outperforms the baseline that only trains Objective 1 (EPA), and Figure 5 shows the comparison with the baseline that only trains Objective 2 (ECC). The complete analysis, rollout-quality curves, and difficulty- and constraint-specific results are provided in Appendix N.

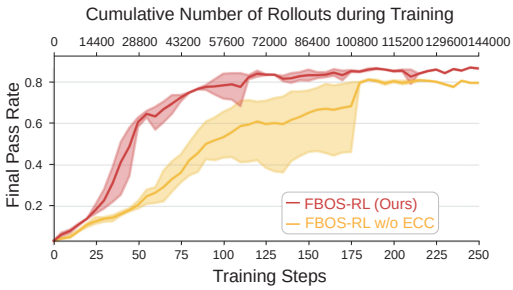


Figure 4: Final pass rate on the TravelPlanner validation set: our method vs. the baseline that only trains Objective 1 (EPA).

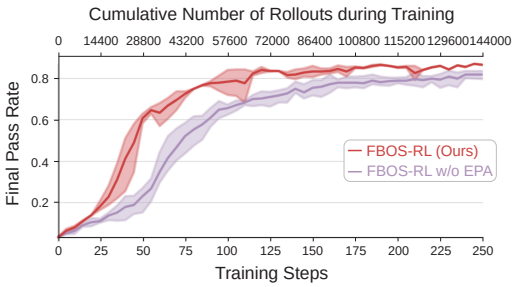


Figure 5: Final pass rate on the TravelPlanner validation set: our method vs. the baseline that only trains Objective 2 (ECC).

FBOS-RL outperforms the GRPO w/ Extra Update baseline on both Qwen3-14B and Llama-3.1-8B-Instruct. These results indicate that the gains of FBOS-RL cannot be explained solely by the additional parameter update introduced by ECC (Appendix O, Figure 42). The cross-prompt ratio and IS ratio transformation comparisons are reported in Appendices J and K. These comparisons evaluate joint surrogate changes and do not isolate the effect of either operation alone. Replacing only f with the piecewise transformation g further improves Qwen3-8B validation performance, especially on medium and hard tasks (Appendix L); Theorems 1 and 2 give the exact role of the transformation derivative in EPA's FAP gradient.

5 Conclusion

We propose **FBOS-RL**, a Feedback-Driven Bi-Objective Synergistic Reinforcement Learning framework. Its two objectives, *Exploitation-oriented Policy Alignment* (EPA) and *Exploration-oriented Capability Cultivation* (ECC), mutually reinforce each other to improve rollout efficiency and peak validation performance. Experiments on TravelPlanner and MiniF2F-Lean4 demonstrate these gains over GRPO under matched rollout budgets, alongside comparisons with strong controlled baselines and OOD generalization to GPQA Diamond. FBOS-RL also maintains higher policy entropy and lower gradient norms than GRPO, consistent with stronger exploration and better training stability.

Limitations. FBOS-RL relies on reliable feedback. Matching rollout counts does not match computational cost: feedback processing and ECC introduce additional work. The reported rollout-efficiency gains do not establish lower wall-clock time or GPU-hours (Appendix R.1).

REFERENCES

R. Agarwal, N. Vieillard, Y. Zhou, P. Stanczyk, S. Ramos Garea, M. Geist, and O. Bachem, "On-policy distillation of language models: Learning from self-generated mistakes," in *International Conference on Learning Representations (ICLR)*, 2024.

Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon *et al.*, "Constitutional ai: Harmlessness from ai feedback," *arXiv preprint arXiv:2212.08073*, 2022.

Z. Chen, Y. Deng, H. Yuan, K. Ji, and Q. Gu, "Self-play fine-tuning converts weak language models to strong language models," *International Conference on Machine Learning (ICML)*, 2024.

G. Cui, L. Yuan, Z. Wang, H. Wang, W. Li, B. He, Y. Fan, T. Yu, Q. Xu, W. Chen *et al.*, "Process reinforcement through implicit rewards," *arXiv preprint arXiv:2502.01456*, 2025a.

G. Cui, Y. Zhang, J. Chen, L. Yuan, Z. Wang, Y. Zuo, H. Li, Y. Fan, H. Chen, W. Chen *et al.*, "The entropy mechanism of reinforcement learning for reasoning language models," *arXiv preprint arXiv:2505.22617*, 2025b.

Z. Gou, Z. Shao, Y. Gong, Y. Shen, Y. Yang, N. Duan, and W. Chen, "Critic: Large language models can self-correct with tool-interactive critiquing," in *International Conference on Learning Representations (ICLR)*, 2024.

D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi *et al.*, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," *arXiv preprint arXiv:2501.12948*, 2025.

J. Huang, X. Chen, S. Mishra, H. S. Zheng, A. W. Yu, X. Song, and D. Zhou, "Large language models cannot self-correct reasoning yet," *arXiv preprint arXiv:2310.01798*, 2023.

A. Kumar, V. Zhuang, R. Agarwal, Y. Su, J. D. Co-Reyes, A. Singh, K. Baumli, S. Iqbal, C. Bishop, R. Roelofs *et al.*, "Training language models to self-correct via reinforcement learning," 2024.

H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe, "Let's verify step by step," *arXiv preprint arXiv:2305.20050*, 2023.

M. Liu, S. Diao, X. Lu, J. Hu, X. Dong, Y. Choi, J. Kautz, and Y. Dong, "Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models," *arXiv preprint arXiv:2505.24864*, 2025.

A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang *et al.*, "Self-refine: Iterative refinement with self-feedback," in *Advances in Neural Information Processing Systems*, 2023.

L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, "Training language models to follow instructions with human feedback," in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 27 730–27 744.

Y. Qu, T. Zhang, N. Garg, and A. Kumar, "Recursive introspection: Teaching language model agents how to self-improve," *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn, "Direct preference optimization: Your language model is secretly a reward model," in *Advances in Neural Information Processing Systems*, 2023.

W. Saunders, C. Yeh, J. Wu, S. Bills, L. Ouyang, J. Ward, and J. Leike, "Self-critiquing models for assisting human evaluators," *arXiv preprint arXiv:2206.05802*, 2022.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

A. Setlur, S. Garg, X. Geng, N. Garg, V. Smith, and A. Kumar, "Rl on incorrect synthetic data scales the efficiency of llm math reasoning by eight-fold," *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. Li, Y. Wu *et al.*, "Deepseekmath: Pushing the limits of mathematical reasoning in open language models," *arXiv preprint arXiv:2402.03300*, 2024.

N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: Language agents with verbal reinforcement learning," in *Advances in Neural Information Processing Systems*, 2023.

C. Snell, J. Lee, K. Xu, and A. Kumar, "Scaling llm test-time compute optimally can be more effective than scaling model parameters," *arXiv preprint arXiv:2408.03314*, 2024.

Y. Tang *et al.*, "Exploration–exploitation trade-off in reinforcement learning for large language models," *arXiv preprint arXiv:2506.10202*, 2025.

C. Wang, Z. Li, J. Bai, Y. Zhang, S. Cui, Z. Zhao, and Y. Wang, "Arbitrary entropy policy optimization: Entropy is controllable in reinforcement finetuning," *arXiv e-prints*, 2025a.

P. Wang, L. Li, Z. Shao, R. Xu, D. Dai, Y. Li, D. Chen, Y. Wu, and Z. Sui, "Math-shepherd: Verify and reinforce llms step-by-step without human annotations," *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Y. Wang, Q. Yang, Z. Zeng, L. Ren, L. Liu, B. Peng, H. Cheng, X. He, K. Wang, J. Gao *et al.*, "Reinforcement learning for reasoning in large language models with one training example," *arXiv preprint arXiv:2504.20571*, 2025b.

S. Welleck, X. Lu, P. West, F. Brahman, T. Shen, D. Khashabi, and Y. Choi, "Generating sequences by learning to self-correct," in *International Conference on Learning Representations (ICLR)*, 2023.

Z. Xi, W. Yang, R. Chen, B. Ding, Y. Liu, J. Liu, R. Zheng, W. Zhou, T. Gui, Q. Zhang, and X. Huang, "Training large language models for reasoning through reverse curriculum reinforcement learning," 2024.

J. Xie, K. Zhang, J. Chen, T. Zhu, R. Lou, Y. Tian, Y. Xiao, and Y. Su, "Travelplanner: A benchmark for real-world planning with language agents," *arXiv preprint arXiv:2402.01622*, 2024.

J. Yan, Y. Li, Z. Hu, Z. Wang, G. Cui, X. Qu, Y. Cheng, and Y. Zhang, "Learning to reason under off-policy guidance," *arXiv preprint arXiv:2504.14945*, 2025.

Q. Yu, Z. Zhang, R. Zhu, Y. Yuan, X. Zuo, Y. Yue, T. Fan, G. Liu, L. Liu, X. Liu *et al.*, "Dapo: An open-source llm reinforcement learning system at scale," *arXiv preprint arXiv:2503.14476*, 2025.

W. Yuan, R. Y. Pang, K. Cho, S. Sukhbaatar, J. Xu, and J. Weston, "Self-rewarding language models," *International Conference on Machine Learning (ICML)*, 2024.

Y. Yue, Z. Chen, R. Lu, A. Zhao, Z. Wang, Y. Yue, S. Song, and G. Huang, "Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model?" *arXiv preprint arXiv:2504.13837*, 2025.

S. Zhao, Z. Xie, M. Liu, J. Huang, G. Pang, F. Chen, and A. Grover, "Self-distilled reasoner: On-policy self-distillation for large language models," *arXiv preprint arXiv:2601.18734*, 2026.

K. Zheng, J. M. Han, and S. Polu, "Minif2f: a cross-system benchmark for formal olympiad-level mathematics," in *International Conference on Learning Representations (ICLR)*, 2022.

T. Zheng *et al.*, "Beyond grpo: Tree-search enhanced reinforcement learning for reasoning," *arXiv preprint arXiv:2502.10717*, 2025.

APPENDIX CONTENTS

A FBOS-RL Algorithm **13**

B The Positive Bootstrapping Flywheel Effect **14**

C Detailed Descriptions of Datasets **15**

D Detailed Definitions of Evaluation Metrics **15**

E Detailed Baseline Methods **16**

F Comparison with OPSD under Identical Feedback **19**

G Full Results by Difficulty Level and Constraint Type **21**

H Stronger Exploration and Better Training Stability **23**

I Out-of-Distribution Generalization **24**

J FBOS-RL w/o Cross-Prompt IS Ratio **25**

K FBOS-RL w/o IS Ratio Transformation **27**

L FBOS-RL w/ Modified IS Ratio Transformation **28**

M Exact Gradients of the FAP Branch in EPA **30**
M.1 Fixed data and the empirical objective 30
M.2 Token-level gradient 31
M.3 The complete finite-batch FAP gradient 32
M.4 Relation to the population objective 33

N Ablation Study of Mutual Reinforcement between EPA and ECC **34**
N.1 Objective 2 (ECC) Boosts Objective 1 (EPA) 34
N.2 Objective 1 (EPA) Boosts Objective 2 (ECC) 36

O Controlling for the Number of Parameter Updates **37**

P Training Dynamics of the EPA Objective **40**

Q Training Dynamics of the ECC Objective **41**

R Hyperparameters **42**
R.1 Rollout 42
R.2 TravelPlanner 42
R.3 MiniF2F-Lean4 42

S Exact Prompt Templates **44**
S.1 TravelPlanner Initial Prompt 44
S.2 MiniF2F Initial Prompt 45
S.3 TravelPlanner Feedback-Augmented Prompt 46
S.4 MiniF2F Feedback-Augmented Prompt 46

A FBOS-RL ALGORITHM

Algorithm 1 FBOS-RL

Require: Dataset $\mathcal{D}$, policy π_θ, rule verifier, initial group size n, feedback group size k, clipping radius ε, optimizer

for each training iteration with original-prompt batch $B \subset \mathcal{D}$ **do**
  Fix the behavior snapshot $\theta_{\text{old}} \leftarrow \theta$.
  Stage 1: Initial Exploration
  for all $q \in B$ in parallel **do**
    Sample $\{\text{ans}_i^1\}_{i=1}^n \sim \pi_{\theta_{\text{old}}}(\cdot \mid q)$ and obtain rewards $\{r_i^1\}_{i=1}^n$ and feedback $\{F_i^1\}_{i=1}^n$.
  end for
  Stage 2: Feedback-Guided Exploration Enhancement
  for all $q \in B$ in parallel **do**
    Construct $\tilde{q}_i = q \oplus \text{ans}_i^1 \oplus F_i^1$ for every $i = 1, \ldots, n$ with the implemented deterministic prompt construction function.
    Sample $\{\text{ans}_{i,j}^2\}_{j=1}^k \sim \pi_{\theta_{\text{old}}}(\cdot \mid \tilde{q}_i)$ and obtain rewards $\{r_{i,j}^2\}_{j=1}^k$ for every $i = 1, \ldots, n$.
    This original prompt now has $N = n + nk$ newly generated rollouts.
  end for
  Before any parameter update, cache the sampled token probabilities or log probabilities under $\pi_{\theta_{\text{old}}}$, conditioned on q for the initial rollouts and on each $\tilde{q}_i$ for the second-round rollouts. Prepare the feedback-free inputs for the EPA current numerator, which is computed during the EPA forward pass.
  Stage 3: Bi-Objective Synergistic Training
  Objective 1, EPA: Starting from θ_{old}, optimize $\mathcal{L}_{\text{EPA}}$ in Equation 3.1 and denote the resulting parameters by θ_{EPA}.
  Objective 2, ECC: Reuse the nk second-round responses per original prompt without further sampling. Starting from θ_{EPA}, optimize $\mathcal{L}_{\text{ECC}}$ in Equation 3.2, using the current numerator computed during the ECC forward pass and the cached FAP behavior probabilities or log probabilities in the denominator.
  Denote the resulting parameters by θ_{new} and set $\theta \leftarrow \theta_{\text{new}}$.
end for

B THE POSITIVE BOOTSTRAPPING FLYWHEEL EFFECT

Although EPA and ECC update the same parameters θ, they act on different conditional distributions: EPA optimizes $\pi_\theta(\cdot \mid q)$, whereas ECC optimizes $\pi_\theta(\cdot \mid \tilde{q}_i)$. Figure 6 illustrates how these objectives can mutually reinforce each other across training stages.

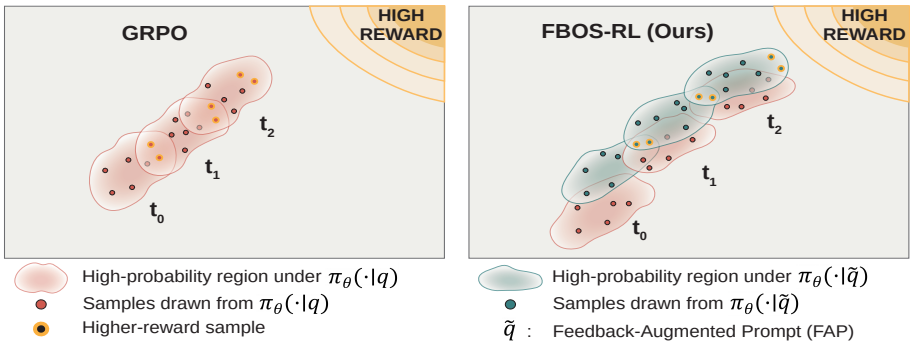


Figure 6: An informal, nonrigorous probability-distribution view of GRPO and FBOS-RL. GRPO samples from and optimizes only $\pi_\theta(\cdot \mid q)$. For FBOS-RL, red and green denote high-probability regions under $\pi_\theta(\cdot \mid q)$ and $\pi_\theta(\cdot \mid \tilde{q}_i)$, respectively. Dots denote sampled rollouts, and gold outlines identify higher-reward samples. The snapshots t_0, t_1, and t_2 show a qualitative progression, not exact dynamics, monotonic convergence, or a theoretical guarantee. We encourage reviewers to watch the corresponding animated video demonstration in the Supplementary Material.

ECC boosts EPA (empirically evaluated in Appendix N.1)**:** Given the error-localized feedback F_i^1 in $\tilde{q}_i$, sampling from $\pi_\theta(\cdot \mid \tilde{q}_i)$ can discover higher-reward rollouts that are unlikely under $\pi_\theta(\cdot \mid q)$. ECC improves this feedback-conditioned sampling capability, allowing $\pi_\theta(\cdot \mid \tilde{q}_i)$ to supply useful higher-reward rollouts. EPA incorporates these rollouts into $\mathcal{G}_q$ and transfers their reward-weighted learning signals to $\pi_\theta(\cdot \mid q)$. Consequently, at a subsequent stage, $\pi_\theta(\cdot \mid q)$ is more likely to generate behaviors previously discovered under $\pi_\theta(\cdot \mid \tilde{q}_i)$. As ECC further trains $\pi_\theta(\cdot \mid \tilde{q}_i)$, it can discover improved rollouts for later EPA updates. Thus, as depicted by the green-to-red progression in Figure 6, $\pi_\theta(\cdot \mid \tilde{q}_i)$ guides $\pi_\theta(\cdot \mid q)$ toward higher-reward behaviors.

EPA elevates ECC (empirical benefits evaluated in Appendix N.2)**:** A possible mechanism for these benefits is the progression of errors and feedback across training. Early in training, rollouts sampled from $\pi_\theta(\cdot \mid q)$ often contain relatively low-level errors, so the resulting F_i^1 and $\tilde{q}_i = q \oplus \mathrm{ans}_i^1 \oplus F_i^1$ may primarily expose ECC to correction tasks involving such errors. By improving $\pi_\theta(\cdot \mid q)$, EPA enables the initial rollouts to overcome some low-level errors. Their remaining errors may then involve deeper logical or constraint-related failures, prompting the verifier to produce F_i^1 that targets these more advanced errors. Because each new F_i^1 becomes part of $\tilde{q}_i$, ECC can learn from a broader and potentially harder set of correction tasks under $\pi_\theta(\cdot \mid \tilde{q}_i)$. The improved $\pi_\theta(\cdot \mid \tilde{q}_i)$ can in turn provide stronger higher-reward samples for EPA, closing the positive bootstrapping flywheel.

C Detailed Descriptions of Datasets

In this section, we provide the detailed descriptions of the datasets used in our experiments.

TravelPlanner. The TravelPlanner dataset is a benchmark designed to evaluate the planning capabilities of language agents in realistic travel-planning scenarios. Each query specifies a travel request (e.g., origin, destination(s), duration, budget, and personalized requirements), and the agent is required to produce a detailed multi-day itinerary.

MiniF2F-Lean4. The MiniF2F-Lean4 dataset is a benchmark for formal mathematical reasoning, consisting of Olympiad-level and high-school competition problems (drawn from sources such as AMC, AIME, and IMO) formalized as theorem-proving problems in the Lean4 proof assistant.

D Detailed Definitions of Evaluation Metrics

In this section, we provide the detailed definitions of the evaluation metrics used in the main paper, following (Xie et al., 2024).

- *Commonsense Constraint Pass Rate*: This metric covers eight commonsense dimensions: whether the cities visited in the itinerary are reasonable, whether restaurant choices are non-repetitive, whether attractions are non-repetitive, whether the accommodations are reasonable, whether the transportation modes are reasonable, whether all daily activities (meals, attractions, accommodations) take place in the city the traveler is in on that day, whether all referenced information (e.g., restaurant names, flight numbers) exists in the closed sandbox database, and whether the plan's information is complete. It evaluates whether the model can incorporate commonsense knowledge into the generated plan without being explicitly instructed.
- *Hard Constraint Pass Rate*: This metric measures whether the generated travel plan satisfies the hard constraints explicitly given in the query (e.g., dietary, accommodation, transportation, and budget constraints), aiming to test the model's ability to adjust its plan according to diverse user requirements.
- *Final Pass Rate*: This metric denotes the proportion of feasible plans (i.e., those satisfying all of the aforementioned constraints, including every commonsense constraint and every hard constraint) among all evaluated plans, and measures the model's ability to generate plans that meet practical standards.

$$\text{Final Pass Rate} = \frac{\sum_{p \in P} \mathbb{I}_{\text{IsSatisfied}(C_p^{\text{all}}, p)}}{|P|}, \tag{D.1}$$

  where P denotes the set of all plans being evaluated, and C_p^{all} denotes the set of all constraints applicable to a specific plan p (including all commonsense constraints and all hard constraints). $\mathbb{I}_{\text{IsSatisfied}(X,Y)}$ is an indicator function that returns 1 if Y satisfies the constraint X, and 0 otherwise. For both the Commonsense Constraint Pass Rate and the Hard Constraint Pass Rate, we adopt two evaluation strategies: micro and macro. The micro strategy computes the proportion of satisfied constraints over the total number of constraints, and The macro strategy computes the proportion of plans that satisfy all commonsense constraints (or all hard constraints) among the evaluated plans, Micro Pass Rate $= \frac{\sum_{p \in P} \sum_{c \in C_p} \mathbb{I}_{\text{IsSatisfied}(c,p)}}{\sum_{p \in P} |C_p|}$, Macro Pass Rate $= \frac{\sum_{p \in P} \mathbb{I}_{\text{IsSatisfied}(C_p,p)}}{|P|}$, where C_p denotes the set of constraints applicable to a specific plan p, and $|C_p|$ denotes the number of constraints in this set. These two strategies evaluate the model's ability to follow individual constraints and the full set of constraints, respectively.

E Detailed Baseline Methods

We compare FBOS-RL with the following training baselines:

- **GRPO** (Shao et al., 2024): We use group-normalized outcome-reward advantages and a clipped policy surrogate, sampling $N_{\mathrm{G}} = n(1+k) = 72$ responses under each original prompt q without reference-policy KL regularization. This matches FBOS-RL's total across both sampling stages, rather than its initial group size $n = 8$.
- **OPSD (On-Policy Self-Distillation)** (Zhao et al., 2026): We adapt OPSD by replacing the teacher's reference-solution context with the FAP $\tilde{q}_i$. For each prompt q in an update batch $B \subset \mathcal{D}$, the student samples n responses $\mathrm{ans}_i \sim \pi_{\theta_{\mathrm{old}}}(\cdot \mid q)$. The frozen teacher π_{θ_0}, initialized from the same base model as the student, scores these same responses under their FAPs. Let $\mathcal{V}^{64}_{i,t}(q)$ contain its 64 highest-probability tokens at prefix $\mathrm{ans}_{i,<t}$. The implemented token loss and batch objective are

$$\ell^{\mathrm{OPSD}}_{i,t}(\theta; q) = \max\left\{0, \sum_{v \in \mathcal{V}^{64}_{i,t}(q)} \pi_{\theta_0}(v \mid \tilde{q}_i, \mathrm{ans}_{i,<t}) \cdot \log \frac{\pi_{\theta_0}(v \mid \tilde{q}_i, \mathrm{ans}_{i,<t})}{\pi_{\theta}(v \mid q, \mathrm{ans}_{i,<t})}\right\}, \tag{E.1}$$

$$\mathcal{L}_{\text{OPSD-FAP}}(\theta; B) = \frac{\sum_{q \in B} \sum_{i=1}^{n} \sum_{t=1}^{|\mathrm{ans}_i|} \ell^{\mathrm{OPSD}}_{i,t}(\theta; q)}{\sum_{q \in B} \sum_{i=1}^{n} |\mathrm{ans}_i|}. \tag{E.2}$$

  Here, i indexes responses for each q, and lengths count valid response tokens. The top-64 approximation retains full-vocabulary probabilities without renormalization, floors the truncated sum at zero, and averages over all valid response tokens in the batch. Only the student is updated; no task-reward policy-gradient term or second-round rollout is used. OPSD uses $n = 8$ student responses per original prompt, yielding 64 responses for a batch of 8 prompts. It shares the initial-response and feedback-instance counts with FBOS-RL, but is not included in the matched-total-rollout comparison with GRPO.

We additionally evaluate the following controlled variants:

- **FBOS-RL w/o EPA**: A variant of FBOS-RL that removes the EPA training objective, which is used to ablate the contribution of the EPA objective.
- **FBOS-RL w/o ECC**: A variant of FBOS-RL that removes the ECC training objective, which is used to ablate the contribution of the ECC objective.
- **GRPO w/ Extra Update**: A variant that reuses 64 of the 72 sampled responses per original prompt for a second update, without new generation. It controls for the additional update introduced by ECC. FBOS-RL's advantage over this baseline indicates that an extra update alone does not explain the gains; this comparison does not match FLOPs or wall-clock time. Results on Qwen3-14B and Llama-3.1-8B-Instruct are provided in Appendix O.

The following three variants examine the treatment of second-round rollouts in EPA. We retain the notation $N = n + nk$, $\hat{A}^{\mathrm{EPA}}$, $\rho^1_{i,t}(\theta)$, and $\rho^2_{i,j,t}(\theta)$ from Section 3.3. In all three variants,

$$\mathcal{L}_{\mathrm{EPA}}(\theta) = \mathcal{L}^{\mathrm{init}}_{\mathrm{EPA}}(\theta) + \mathcal{L}^{\mathrm{FAP}}_{\mathrm{EPA}}(\theta), \tag{E.3}$$

where only $\mathcal{L}^{\mathrm{FAP}}_{\mathrm{EPA}}$ changes. The initial EPA branch, mixed-group advantages, ECC objective, and sequential update $\theta_{\mathrm{old}} \to \theta_{\mathrm{EPA}} \to \theta_{\mathrm{new}}$ are retained. For reference, the full FBOS-RL method uses

$$\rho^2_{i,j,t}(\theta) = \frac{\pi_\theta\big(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t}\big)}{\pi_{\theta_{\mathrm{old}}}\big(\mathrm{ans}^2_{i,j,t} \mid \tilde{q}_i, \mathrm{ans}^2_{i,j,<t}\big)}, \tag{E.4}$$

$$\mathcal{L}^{\mathrm{FAP}}_{\mathrm{EPA}}(\theta)\big|_{\text{FBOS-RL}} = -\mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{n}\sum_{j=1}^{k}\frac{1}{|\mathrm{ans}^2_{i,j}|}\sum_{t=1}^{|\mathrm{ans}^2_{i,j}|} f\big(\rho^2_{i,j,t}(\theta)\big)\,\hat{A}^{\mathrm{EPA}}(\mathrm{ans}^2_{i,j})\right], \tag{E.5}$$

with $f(\rho) = \rho/(\rho + 0.1)$ and no PPO clipping in this second-round branch.

For a ratio ρ and advantage $\hat{A}$, the standard PPO per-token clipped loss is

$$\ell_{\mathrm{clip}}(\rho, \hat{A}) = -\min\Big\{\rho\hat{A}, \mathrm{clip}(\rho, 1-\varepsilon, 1+\varepsilon)\hat{A}\Big\}, \tag{E.6}$$

where $\varepsilon = 0.2$ is the clipping radius.

- **FBOS-RL w/o Cross-Prompt IS Ratio**: Both likelihoods condition on the same FAP, and the second-round EPA loss is

$$\tilde{\rho}_{i,j,t}(\theta) = \frac{\pi_\theta\big(\mathrm{ans}^2_{i,j,t} \mid \tilde{q}_i, \mathrm{ans}^2_{i,j,<t}\big)}{\pi_{\theta_{\mathrm{old}}}\big(\mathrm{ans}^2_{i,j,t} \mid \tilde{q}_i, \mathrm{ans}^2_{i,j,<t}\big)}, \tag{E.7}$$

$$\mathcal{L}^{\mathrm{FAP}}_{\mathrm{EPA}}(\theta)\big|_{\text{w/o Cross-Prompt IS Ratio}} = \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{n}\sum_{j=1}^{k}\frac{1}{|\mathrm{ans}^2_{i,j}|}\sum_{t=1}^{|\mathrm{ans}^2_{i,j}|}\ell_{\mathrm{clip}}\Big(\tilde{\rho}_{i,j,t}(\theta), \hat{A}^{\mathrm{EPA}}(\mathrm{ans}^2_{i,j})\Big)\right]. \tag{E.8}$$

  Relative to Equation E.5, this variant changes both the numerator's prompt and the second-round surrogate.

- **FBOS-RL w/o IS Ratio Transformation**: This variant retains the cross-prompt ratio in Equation E.4, removes the IS ratio transformation, and restores PPO clipping in EPA's FAP branch as defined in Equation E.6:

$$\mathcal{L}^{\mathrm{FAP}}_{\mathrm{EPA}}(\theta)\big|_{\text{w/o transformation}} = \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{n}\sum_{j=1}^{k}\frac{1}{|\mathrm{ans}^2_{i,j}|}\sum_{t=1}^{|\mathrm{ans}^2_{i,j}|}\ell_{\mathrm{clip}}\Big(\rho^2_{i,j,t}(\theta), \hat{A}^{\mathrm{EPA}}(\mathrm{ans}^2_{i,j})\Big)\right]. \tag{E.9}$$

  Equations E.5 and E.9 compare the unclipped surrogate with IS ratio transformation against the clipped surrogate on the untransformed ratio. Equations E.8 and E.9 differ only in $\tilde{\rho}_{i,j,t}(\theta)$ versus $\rho^2_{i,j,t}(\theta)$. Results for this variant are reported in Appendix K.

- **FBOS-RL w/ Modified IS Ratio Transformation**: The cross-prompt ratio is retained, and only f is replaced by g:

$$g(\rho) = \begin{cases} \dfrac{\rho}{\rho + 0.1}, & 0 \le \rho \le \sqrt{0.1} - 0.1, \\ \rho + 1.1 - 2\sqrt{0.1}, & \sqrt{0.1} - 0.1 < \rho \le 1, \\ \dfrac{\rho - 1.1 + \sqrt{0.1}}{\rho - 1 + \sqrt{0.1}} + 1.1 - \sqrt{0.1}, & \rho > 1. \end{cases} \tag{E.10}$$

  Its second-round EPA objective is

$$\mathcal{L}^{\mathrm{FAP}}_{\mathrm{EPA}}(\theta)\big|_{g} = -\mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{n}\sum_{j=1}^{k}\frac{1}{|\mathrm{ans}^2_{i,j}|}\sum_{t=1}^{|\mathrm{ans}^2_{i,j}|} g\big(\rho^2_{i,j,t}(\theta)\big)\,\hat{A}^{\mathrm{EPA}}(\mathrm{ans}^2_{i,j})\right]. \tag{E.11}$$

  Comparing Equations E.5 and E.11, the sole change is $f \mapsto g$; neither applies PPO clipping to second-round EPA tokens. Appendix L reports the results, and Appendix M derives how the transformation enters the FAP gradient.

For GRPO and FBOS-RL, the original-prompt order and total sampling budget follow Section 4.1: 72 new responses per original prompt and 576 per iteration, including both FBOS-RL sampling stages. These experiments are repeated three times; solid curves show pointwise means and shaded regions show pointwise min–max ranges. In addition, on the TravelPlanner dataset, we also include the following baselines:

- **Greedy Search** To evaluate the performance of traditional search algorithms in TravelPlanner, we adopt the greedy search strategy as one of the baselines. Greedy Search focuses on cost minimization as its core objective. Among transportation options, it selects the one with the lowest cost; for dining, it chooses restaurants with the lowest average expenditure; for accommodation, it selects the cheapest option; and for sightseeing, it arranges attractions by randomly selecting them each day. For a 5-day or 7-day travel plan, select the top 1 to 2 cities as destinations from the returned city search results.
- **Sole-Planning Mode** We focus on the sole-planning mode of the TravelPlan task. In this mode, the model is provided in advance with sufficient and necessary reference information required for reasoning and planning. This setting is used to evaluate the model's ability to perform complex reasoning and planning directly based on the given information. The baselines under the sole-planning mode include the following models and strategies:

- *Models:* GPT-3.5-Turbo, GPT-4-Turbo, GPT-4o (version: 2024-11-20), Mixtral-8×7B-MoE, Gemini Pro, Qwen3-8B-Instruct, DeepSeek-R1.
- *Strategies*: Our baselines include the following strategies: Direct, CoT, ReAct, Reflexion, and prompt reflect. Specifically, "Direct" refers to prompting the model to directly generate the final travel plan; "CoT" refers to prompting the model to reason step by step before producing the final travel plan; "ReAct" refers to prompting the model to solve the task by alternating between Thought, Action, and Observation steps; "Reflexion" refers to prompting the model to perform self-reflection before generating the final travel plan; "Prompt reflect" refers to prompting the model to reflect on its previous reasoning before producing the final travel plan.

F Comparison with OPSD under Identical Feedback

Motivation and setup. Because FBOS-RL uses feedback during training, comparison with a method that also uses feedback helps assess whether access to feedback alone explains its gains. We therefore compare FBOS-RL with a feedback-conditioned adaptation of OPSD (Zhao et al., 2026) on TravelPlanner using Llama-3.1-8B-Instruct. Both methods receive exactly the same feedback information. In OPSD, this feedback enters the frozen teacher's FAP, while the student generates responses from the original prompt q. The teacher is initialized from the same base model as the student; only the student is updated. The objective is specified in Equations E.1–E.2.

Both methods use 8 original prompts and 8 initial responses per prompt, corresponding to 64 feedback instances per iteration. The upper axes in Figures 7 and 8 therefore show $64t$ feedback instances after t training iterations. This comparison matches feedback information and its count. Evaluation uses the same 1,000 held-out TravelPlanner instances and metrics as in Section 4.1.

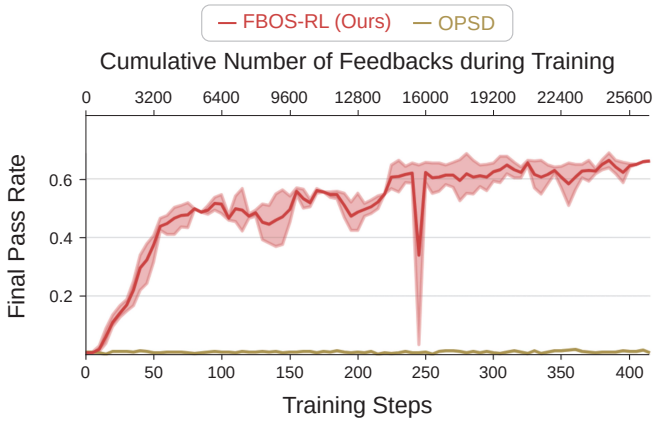


Figure 7: Validation Final Pass Rate of FBOS-RL and OPSD on TravelPlanner with Llama-3.1-8B-Instruct under identical feedback. The bottom axis shows training iterations; the top axis shows cumulative feedback instances.

Validation performance. Figure 7 shows a substantial separation: FBOS-RL reaches a Final Pass Rate above 0.6 in later training, whereas OPSD remains close to zero over the displayed interval. Figure 8 shows higher late-training pass rates for FBOS-RL across all four constraint metrics. OPSD satisfies some individual commonsense constraints, but its commonsense macro and hard-constraint pass rates remain low. Thus, under this feedback-conditioned OPSD setup, access to the same feedback does not reproduce the planning improvements obtained by FBOS-RL.

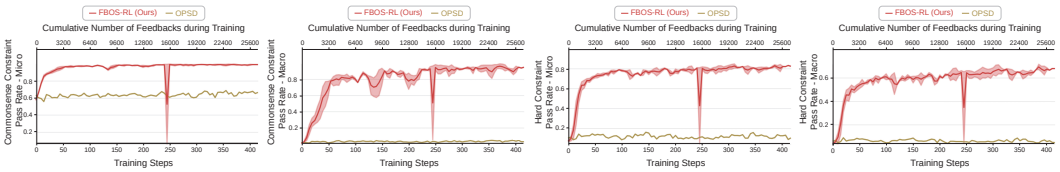


Figure 8: Constraint-level validation performance for the same comparison. From left to right: Commonsense Constraint Pass Rate (Micro and Macro), and Hard Constraint Pass Rate (Micro and Macro). The two horizontal axes follow Figure 7.

OPSD training diagnostics. To examine the low validation performance, we report the student policy entropy and distillation loss in Figure 9. Following an early decline, entropy remains around 0.8–1.0, rather than approaching zero, while the distillation loss falls sharply and then fluctuates within a relatively stable range. Here, loss denotes the implemented teacher-top-64 forward-KL surrogate, with negative truncated token losses floored at zero before averaging over valid response tokens; it is not a task reward or a full-vocabulary KL divergence. These curves show neither sustained loss divergence nor vanishing average token entropy over the displayed training interval.

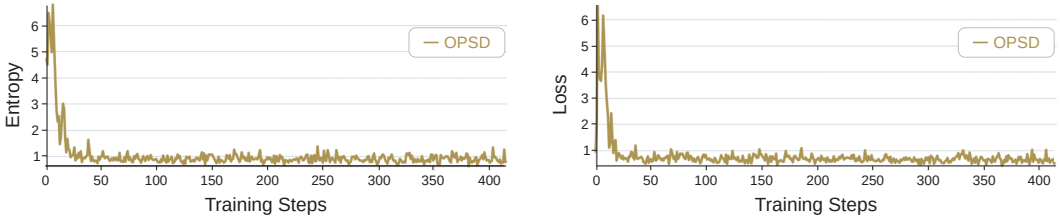


Figure 9: OPSD training diagnostics. Left: student policy entropy. Right: the distillation objective in Equation E.2. Both are plotted against training iterations.

Figure 10 further reports the teacher and student probability mass on the teacher-selected top-64 tokens. Using the notation of Equation E.1, the per-position quantities are

$$m^{\mathrm{T}}_{i,t}(q) = \sum_{v \in \mathcal{V}^{64}_{i,t}(q)} \pi_{\theta_0}(v \mid \tilde{q}_i, \mathrm{ans}_{i,<t}),$$
$$m^{\mathrm{S}}_{i,t}(q) = \sum_{v \in \mathcal{V}^{64}_{i,t}(q)} \pi_{\theta}(v \mid q, \mathrm{ans}_{i,<t}). \tag{F.1}$$

The logged metrics average these masses over valid response positions. Both sums use probabilities normalized over the full vocabulary and the same teacher-selected token set; the student metric does not use the student's own top-64 tokens. After the initial transient, both masses remain above 0.95, indicating high average probability coverage of the retained token set. High coverage does not establish agreement between teacher and student probabilities within that set, or correctness of the generated plans. Together, these diagnostics support stable behavior of the monitored quantities despite low validation performance; they do not by themselves rule out every form of training failure.

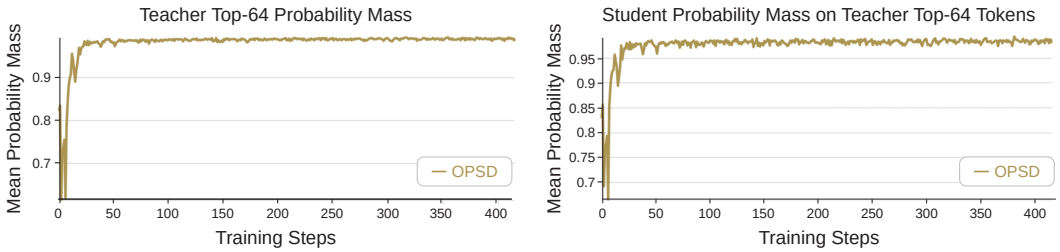


Figure 10: Probability mass on the teacher's top-64 tokens during OPSD training. Left: teacher mass $m^{\mathrm{T}}_{i,t}(q)$. Right: student mass $m^{\mathrm{S}}_{i,t}(q)$ on the same teacher-selected set. Curves report means over valid response positions, using full-vocabulary probabilities without renormalization over the retained tokens.

G Full Results by Difficulty Level and Constraint Type

To understand more precisely *where* the gains of FBOS-RL over vanilla GRPO come from, we further break down the validation performance along two axes: by sample *difficulty* (which probes how the gap behaves as the planning problem becomes harder) and by *constraint type* (which probes whether the gains are driven by commonsense reasoning, by satisfaction of explicit hard constraints, or by both).

Breakdown by difficulty. The TravelPlanner dataset partitions samples by difficulty into "easy", "medium", and "hard". We track the final pass rate on each difficulty subset of the validation set throughout training. Figure 11 reports the results for Qwen3-8B, Llama-3.1-8B-Instruct, and Qwen3-14B in the top, middle, and bottom rows, respectively, with easy, medium, and hard subsets arranged from left to right. Across the three backbones, FBOS-RL generally achieves higher final pass rates at all difficulty levels, although local crossings occur. For Qwen3-8B, the late-training gap is larger on hard instances than on easy instances. For Qwen3-14B, easy instances approach saturation for both methods, while the medium and hard subsets retain a clearer separation. These results highlight the gains of FBOS-RL on challenging, multi-constraint planning instances.

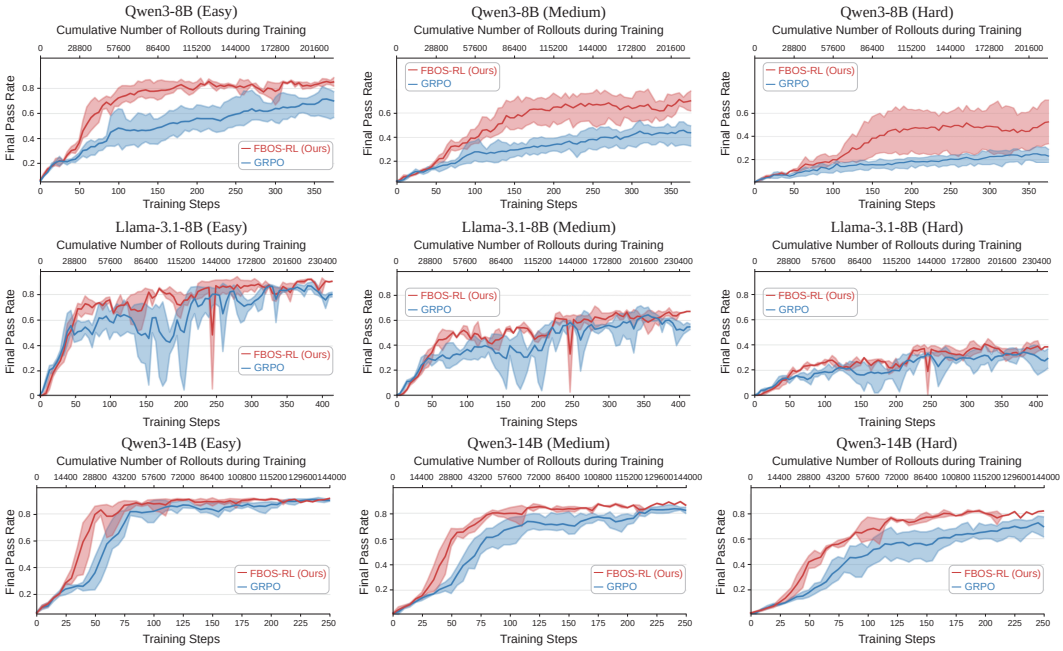


Figure 11: Final pass rate on the TravelPlanner validation set during training, broken down by difficulty level. Rows from top to bottom: Qwen3-8B, Llama-3.1-8B-Instruct, and Qwen3-14B. Columns from left to right: easy, medium, and hard. Each panel compares FBOS-RL with vanilla GRPO. The bottom and top x-axes show training steps and cumulative rollouts, respectively.

Breakdown by constraint type. By definition (see Appendix D), the Final Pass Rate is computed at the plan level and only credits a plan when *all* commonsense and *all* hard constraints are simultaneously satisfied, which limits its resolution in two respects: (i) it operates at the plan level rather than the individual-constraint level, so it cannot reflect improvements measured at the per-constraint granularity captured by the *Micro* statistics; and (ii) it aggregates commonsense and hard constraints into a single "all satisfied" indicator, so it cannot reveal how the two constraint families behave separately. To obtain a finer-grained picture along both axes, we additionally track the Commonsense Constraint Pass Rate (both Micro and Macro) and the Hard Constraint Pass Rate (both Micro and Macro) on the validation set throughout training; detailed mathematical definitions of these metrics are provided in Appendix D. Figure 12 reports all four metrics for Qwen3-8B, Llama-3.1-8B-Instruct, and Qwen3-14B in the top, middle, and bottom rows, respectively. Table 1 shows improvements over GRPO on all four metrics for Llama-3.1-8B-Instruct and Qwen3-14B. For Qwen3-8B, FBOS-RL improves Commonsense Macro, Hard Micro, and Hard Macro, while Commonsense Micro is lower

than GRPO (93.57 vs. 96.43). Thus, the higher Final Pass Rate on Qwen3-8B coexists with a trade-off in individual commonsense-constraint satisfaction, while the gains in both Macro metrics indicate that more plans satisfy all constraints within each constraint family.

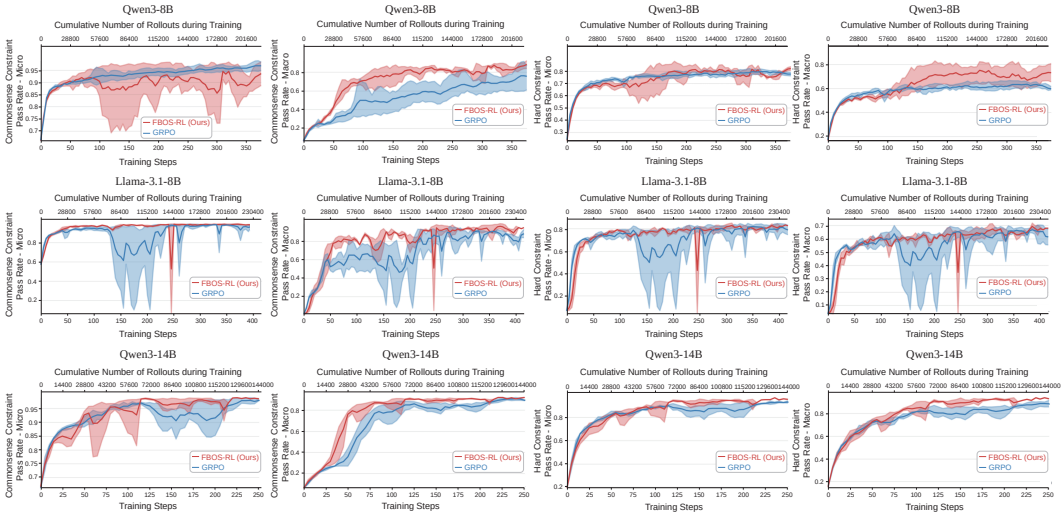


Figure 12: Commonsense and hard constraint pass rates on the TravelPlanner validation set during training. Rows from top to bottom: Qwen3-8B, Llama-3.1-8B-Instruct, and Qwen3-14B. Columns from left to right: Commonsense Constraint Pass Rate (Micro), Commonsense Constraint Pass Rate (Macro), Hard Constraint Pass Rate (Micro), and Hard Constraint Pass Rate (Macro). Each panel compares FBOS-RL with vanilla GRPO. The bottom and top x-axes show training steps and cumulative rollouts, respectively.

H STRONGER EXPLORATION AND BETTER TRAINING STABILITY

Beyond the validation accuracy itself, we further measure the actor policy entropy and the gradient norm during training.

Stronger exploration. GRPO can be prone to the entropy collapse phenomenon (Cui et al., 2025b; Yu et al., 2025; Wang et al., 2025a), in which the actor's output distribution rapidly contracts onto a narrow set of high-probability actions during training, causing the policy to become overly deterministic and to lose the capacity to discover new, potentially higher-quality rollouts. Figure 13 compares four settings: Qwen3-8B, Llama-3.1-8B-Instruct, and Qwen3-14B on TravelPlanner, and Qwen3.5-27B on MiniF2F-Lean4. FBOS-RL generally maintains higher actor entropy than vanilla GRPO, with a particularly clear separation on Qwen3-8B and Qwen3-14B over most of training. These observations are consistent with FBOS-RL preserving broader output distributions during training.

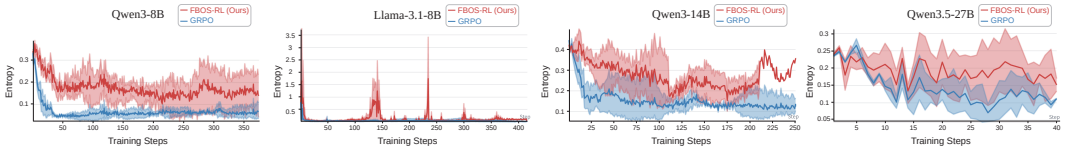


Figure 13: Actor entropy during training: our method (FBOS-RL) vs. vanilla GRPO. From left to right: Qwen3-8B, Llama-3.1-8B-Instruct, and Qwen3-14B on TravelPlanner, followed by Qwen3.5-27B on MiniF2F-Lean4. FBOS-RL generally maintains higher entropy than vanilla GRPO across the four settings.

Better training stability. We further examine the gradient norm during training. As shown in Figure 14, across the four settings, including Qwen3-8B on TravelPlanner, FBOS-RL exhibits a lower gradient norm than vanilla GRPO for most training steps. This observation is consistent with more stable optimization dynamics under the same training settings.

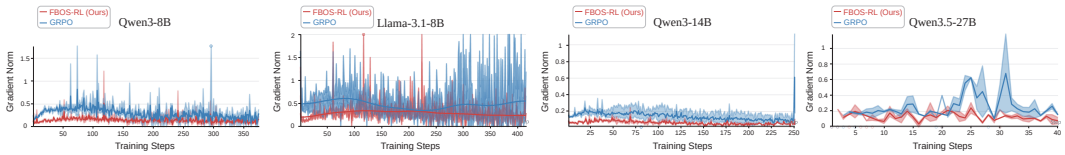


Figure 14: Gradient norm during training: our method (FBOS-RL) vs. vanilla GRPO. From left to right: Qwen3-8B, Llama-3.1-8B-Instruct, and Qwen3-14B on TravelPlanner, followed by Qwen3.5-27B on MiniF2F-Lean4. Across the four settings, FBOS-RL exhibits a lower gradient norm than vanilla GRPO for most training steps.

I Out-of-Distribution Generalization

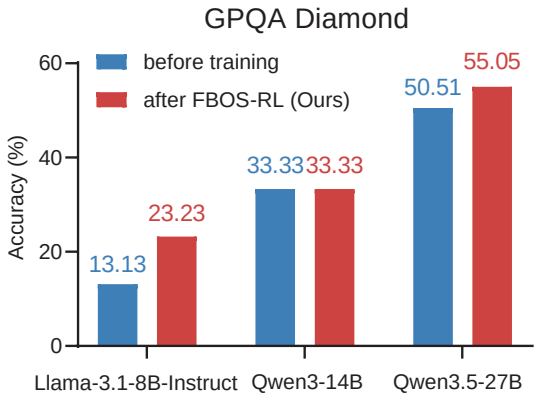


Figure 15: Out-of-distribution generalization to the GPQA-Diamond dataset: comparison of three models (Llama-3.1-8B-Instruct, Qwen3-14B, and Qwen3.5-27B) before training and after training with our FBOS-RL method. All models use greedy decoding during inference. The Llama-3.1-8B-Instruct and Qwen3-14B models are trained with FBOS-RL on the TravelPlanner dataset, while the Qwen3.5-27B model is trained with FBOS-RL on the MiniF2F-Lean4 dataset. None of the models are further trained on GPQA-Diamond, so this evaluation directly measures the out-of-distribution generalization gains brought by FBOS-RL.

We further evaluate the GPQA-Diamond performance of each model before training and after FBOS-RL training, with results shown in Figure 15. All models use greedy decoding during inference. The Llama-3.1-8B-Instruct and Qwen3-14B models are trained only on TravelPlanner with FBOS-RL, while the Qwen3.5-27B model is trained only on MiniF2F-Lean4 with FBOS-RL; none of these models are trained on GPQA-Diamond. Compared with the untrained Llama-3.1-8B-Instruct model, the Llama-3.1-8B-Instruct model trained with FBOS-RL on TravelPlanner improves by 10.1 percentage points on GPQA-Diamond, corresponding to a relative improvement of 76.92%. For Qwen3-14B, training with FBOS-RL on TravelPlanner does not lead to any performance degradation on GPQA-Diamond. For Qwen3.5-27B, training with FBOS-RL on MiniF2F-Lean4 improves GPQA-Diamond accuracy by 4.54 percentage points over the untrained Qwen3.5-27B model, corresponding to a relative improvement of 8.99%.

J FBOS-RL w/o Cross-Prompt IS Ratio

The definition and training objective of FBOS-RL w/o Cross-Prompt IS Ratio are provided in Appendix E.

Informal probability-distribution perspective. Figure 16 gives an informal and nonrigorous explanation of the observed difference in rollout efficiency and validation performance over the reported training horizon. Because this baseline jointly changes the ratio conditioning and surrogate transform, the following discussion describes a possible mechanism and does not identify the independent causal effect of the ratio. In FBOS-RL, the cross-prompt importance sampling ratio directly evaluates high-reward rollouts sampled from $\pi_\theta(\cdot \mid \tilde{q}_i)$ under the original-prompt policy $\pi_\theta(\cdot \mid q)$. Positive EPA updates can therefore transfer probability mass toward these discoveries in $\pi_\theta(\cdot \mid q)$. Under the joint baseline change, the EPA term instead updates $\pi_\theta(\cdot \mid \tilde{q}_i)$ through a same-FAP ratio and a standard clipped surrogate. The two conditioned distributions are consequently trained more separately, and high-reward samples discovered under $\pi_\theta(\cdot \mid \tilde{q}_i)$ no longer provide the same direct transfer signal to $\pi_\theta(\cdot \mid q)$. This missing transfer path offers a possible mechanism-level interpretation of the baseline's slower movement toward high-reward regions and lower observed late-training validation performance.

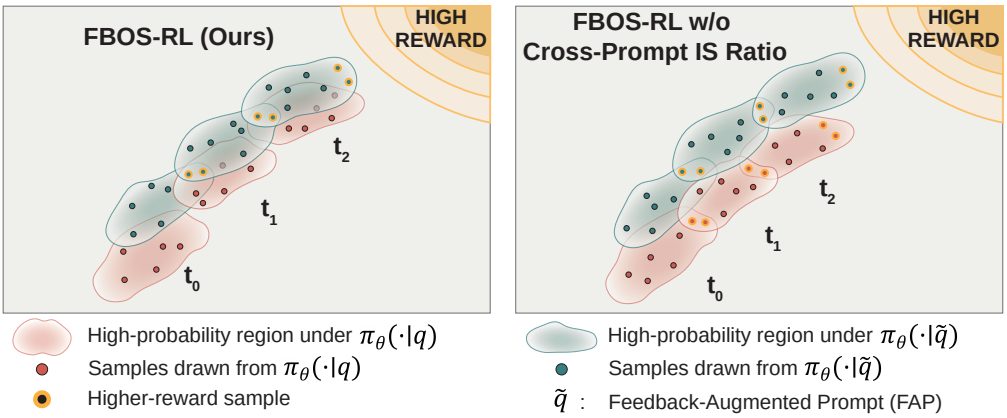


Figure 16: An informal probability-mass view of FBOS-RL and FBOS-RL w/o Cross-Prompt IS Ratio. FBOS-RL uses rollouts sampled from $\pi_\theta(\cdot \mid \tilde{q}_i)$ to update $\pi_\theta(\cdot \mid q)$ through its cross-context ratio, whereas the baseline applies a same-FAP ratio and therefore lacks this direct transfer path. The displayed regions and trajectories are qualitative. They do not define a distance, precise optimization dynamic, monotone trajectory, convergence guarantee, or causal proof. The empirical baseline also changes the surrogate transform, so this comparison does not isolate ratio-only causality. We encourage reviewers to watch the corresponding animated video demonstration in the Supplementary Material.

Experimental setting and aggregation. We compare FBOS-RL, FBOS-RL w/o Cross-Prompt IS Ratio, and GRPO using Qwen3-14B on TravelPlanner. Figures 17, 18, and 19 report the 1,000-example validation set without smoothing. In these three validation figures, the lower horizontal axes report training steps and the upper horizontal axes report cumulative rollouts. Figures 20, 21, and 22 report training-set sampling quality with exponential moving average (EMA) smoothing, with horizontal axes reporting training steps.

Validation results. FBOS-RL generally improves validation performance with fewer generated training rollouts and reaches the highest late-training validation performance. FBOS-RL w/o Cross-Prompt IS Ratio generally lies between FBOS-RL and GRPO, although local crossings occur. Easy examples approach saturation for all methods, while the hard subset provides the clearest separation.

Training-set sampling results. Across most training steps, FBOS-RL generally attains stronger sampling quality during both the initial sampling stage and the entire sampling phase than GRPO, while FBOS-RL w/o Cross-Prompt IS Ratio generally follows an intermediate trajectory with some late-stage approaches or crossings. The initial-sampling curves and the curves covering the entire sampling phase are jointly consistent with the intended cross-context treatment: the EPA learning

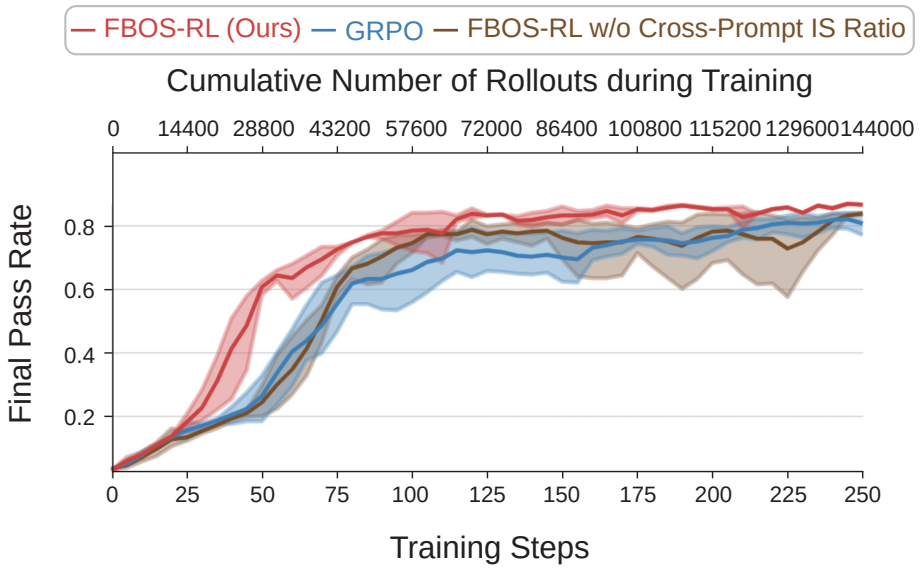


Figure 17: Qwen3-14B overall Final Pass Rate on the 1,000-example TravelPlanner validation set. Each panel compares FBOS-RL, FBOS-RL w/o Cross-Prompt IS Ratio, and GRPO. No smoothing is applied.

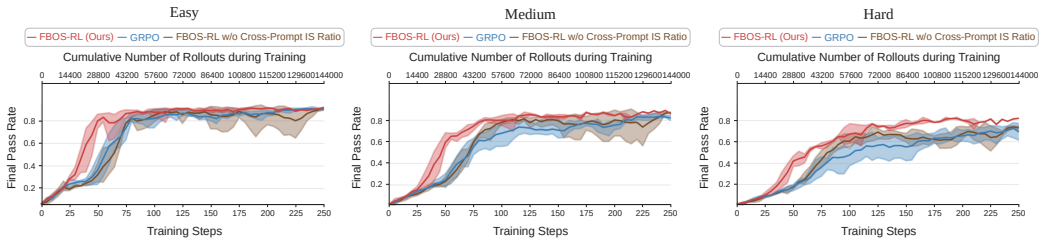


Figure 18: Qwen3-14B Final Pass Rate on the 1,000-example TravelPlanner validation set for easy, medium, and hard examples from left to right. Each panel compares FBOS-RL, FBOS-RL w/o Cross-Prompt IS Ratio, and GRPO. No smoothing is applied.

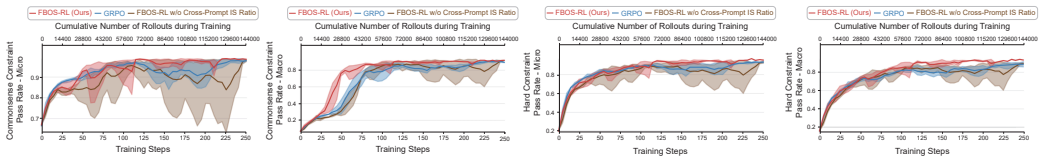


Figure 19: Qwen3-14B constraint pass rates on the 1,000-example TravelPlanner validation set. From left to right: Commonsense Micro, Commonsense Macro, Hard Micro, and Hard Macro. Each panel compares FBOS-RL, FBOS-RL w/o Cross-Prompt IS Ratio, and GRPO. No smoothing is applied.

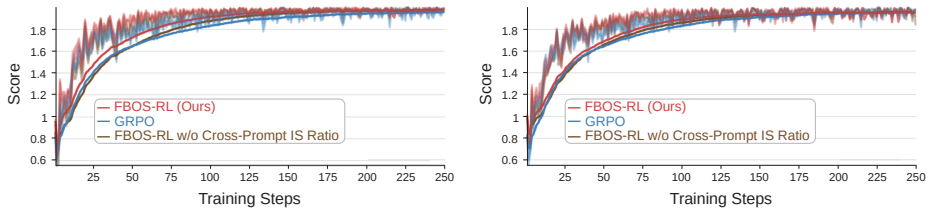


Figure 20: Qwen3-14B training-set sampling quality. The left panel is the mean score of rollouts from the initial sampling stage under $\pi_\theta(\cdot \mid q)$. The right panel is the mean score of rollouts generated during the entire sampling phase, which includes initial sampling from the original prompt under $\pi_\theta(\cdot \mid q)$ and second-round sampling from the FAP under $\pi_\theta(\cdot \mid \tilde{q}_i)$. Each panel compares FBOS-RL, FBOS-RL w/o Cross-Prompt IS Ratio, and GRPO. EMA smoothing is applied.

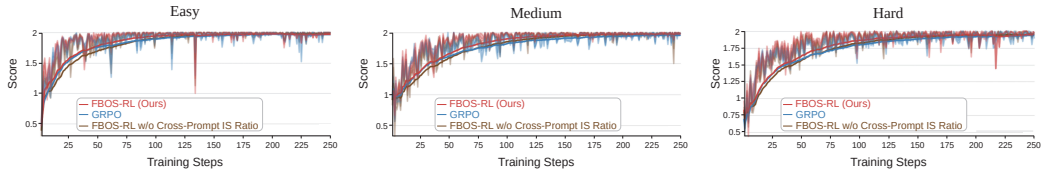


Figure 21: Qwen3-14B training-set sampling quality under $\pi_\theta(\cdot \mid q)$, with easy, medium, and hard examples from left to right. Each panel compares FBOS-RL, FBOS-RL w/o Cross-Prompt IS Ratio, and GRPO. EMA smoothing is applied.

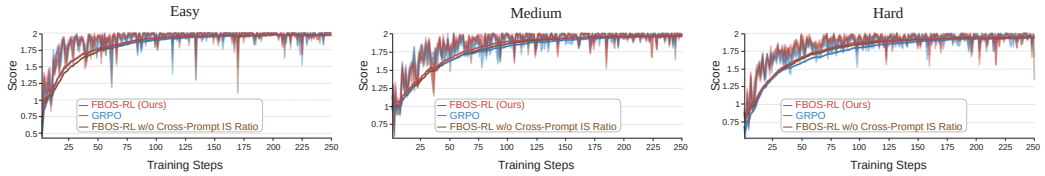


Figure 22: Qwen3-14B mean score of rollouts generated during the entire sampling phase, including initial sampling from the original prompt under $\pi_\theta(\cdot \mid q)$ and second-round sampling from the FAP under $\pi_\theta(\cdot \mid \tilde{q}_i)$, with easy, medium, and hard examples from left to right. Each panel compares FBOS-RL, FBOS-RL w/o Cross-Prompt IS Ratio, and GRPO. EMA smoothing is applied.

signal can use second-round rollouts sampled from $\pi_\theta(\cdot \mid \tilde{q}_i)$ to improve $\pi_\theta(\cdot \mid q)$. This empirical comparison does not isolate ratio-only causality because FBOS-RL w/o Cross-Prompt IS Ratio changes both ratio conditioning and the surrogate transform.

K FBOS-RL w/o IS Ratio Transformation

Experimental setting. We compare FBOS-RL, FBOS-RL w/o IS Ratio Transformation, and GRPO using Qwen3-8B on the 1,000-example TravelPlanner validation set. As defined in Appendix E, this variant retains the cross-prompt ratio but replaces the transformed surrogate with PPO clipping in EPA's FAP branch. The initial EPA branch, mixed-group advantages, ECC objective, and update order remain unchanged. In Figures 23–25, solid curves show the pointwise mean of three independent runs, and shaded regions span their pointwise minimum and maximum. No smoothing is applied. The lower and upper horizontal axes report training steps and cumulative rollouts, respectively.

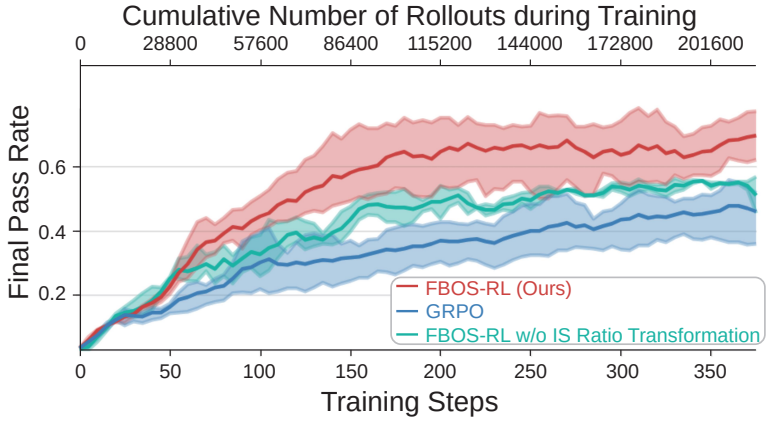


Figure 23: Overall Final Pass Rate of Qwen3-8B on the TravelPlanner validation set, comparing FBOS-RL, FBOS-RL w/o IS Ratio Transformation, and GRPO. Curves and bands show the mean and min–max range across three runs, without smoothing.

Overall performance and task difficulty. Figure 23 shows that, after the initial learning phase, FBOS-RL maintains a higher mean Final Pass Rate than FBOS-RL w/o IS Ratio Transformation

through most of the remaining training horizon. FBOS-RL w/o IS Ratio Transformation generally outperforms GRPO but retains a substantial gap to the full method. The difficulty breakdown in Figure 24 further localizes this difference: on easy examples, FBOS-RL w/o IS Ratio Transformation approaches FBOS-RL later in training, whereas medium and hard examples retain clearer gaps. On hard examples, FBOS-RL w/o IS Ratio Transformation remains much closer to GRPO than to FBOS-RL. The benefit of the full surrogate design is therefore most apparent on tasks where satisfying all constraints remains challenging.

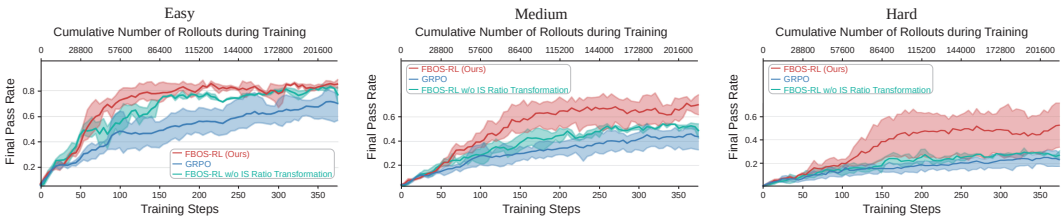


Figure 24: Final Pass Rate of Qwen3-8B by TravelPlanner task difficulty: easy, medium, and hard examples from left to right. Each panel compares FBOS-RL, FBOS-RL w/o IS Ratio Transformation, and GRPO, with the same three-run aggregation and axes as Figure 23.

Constraint-level results. Figure 25 shows the clearest sustained advantage of FBOS-RL in Hard Constraint Pass Rate (Macro), which measures whether all hard constraints in a plan are satisfied. The other constraint metrics have more overlap: FBOS-RL w/o IS Ratio Transformation approaches FBOS-RL in Commonsense Constraint Pass Rate (Macro) later in training, the Hard Constraint Pass Rate (Micro) curves become close, and GRPO is competitive or higher in Commonsense Constraint Pass Rate (Micro) over substantial portions of training.

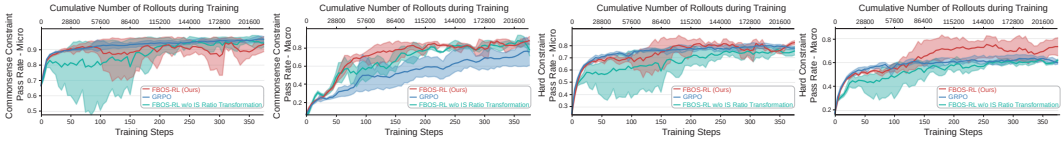


Figure 25: Constraint pass rates of Qwen3-8B on TravelPlanner. From left to right, the panels report Commonsense Micro, Commonsense Macro, Hard Micro, and Hard Macro. Each panel compares FBOS-RL, FBOS-RL w/o IS Ratio Transformation, and GRPO, with the same three-run aggregation and axes as Figure 23.

These results support the effectiveness of the FAP surrogate with IS ratio transformation used by FBOS-RL relative to the clipped replacement in this experiment, particularly for overall success and harder tasks. Because this comparison both removes the transformation and restores clipping, it evaluates the joint surrogate change and does not isolate the effect of either operation alone.

L FBOS-RL w/ Modified IS Ratio Transformation

Controlled change. We evaluate whether changing the shape of the IS ratio transformation can improve FBOS-RL. The variant replaces $f(\rho) = \rho/(\rho+0.1)$ with g in Equation E.10, while retaining the cross-prompt ratio, the initial EPA branch, mixed-group advantages, the ECC objective, and the sequential EPA–ECC update order. Both f and g are applied without PPO clipping in EPA's FAP branch. Thus, this comparison isolates the choice of transformation within the same surrogate structure.

Transformation and its gradient weight. Figure 26 compares the two functions and their derivatives. Direct differentiation gives

$$f'(\rho) = \frac{0.1}{(\rho+0.1)^2}, \qquad g'(\rho) = \begin{cases} \dfrac{0.1}{(\rho+0.1)^2}, & 0 < \rho \le \sqrt{0.1} - 0.1, \\ 1, & \sqrt{0.1} - 0.1 < \rho \le 1, \\ \dfrac{0.1}{(\rho - 1 + \sqrt{0.1})^2}, & \rho > 1. \end{cases} \tag{L.1}$$

At $\rho = \sqrt{0.1} - 0.1$, both adjacent pieces of g have value $1 - \sqrt{0.1}$ and derivative 1; at $\rho = 1$, they have value $2.1 - 2\sqrt{0.1}$ and derivative 1. Hence g is continuously differentiable for positive ratios, including both joins. It preserves the low-ratio branch of f, maintains a unit slope over the intermediate interval, and retains a decaying derivative above 1.

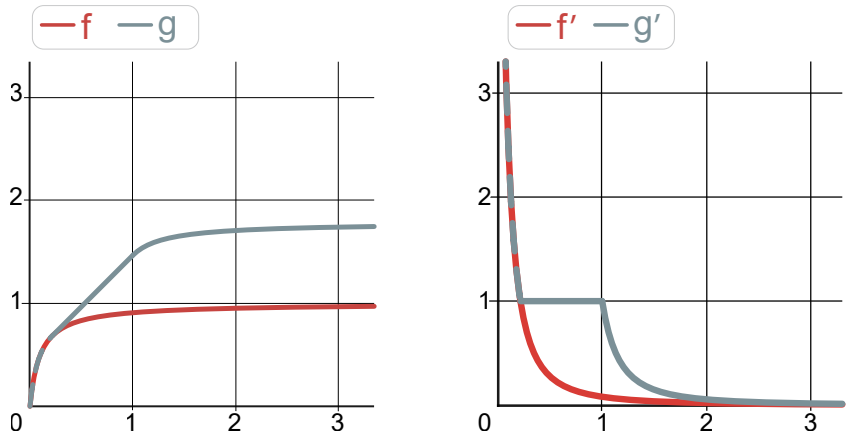


Figure 26: Original transformation f and modified transformation g (left), and their derivatives (right), as functions of the cross-prompt ratio ρ. The modified function preserves the low-ratio segment and has $g'(\rho) = 1$ for $\sqrt{0.1} - 0.1 < \rho \leq 1$.

Theorems 1 and 2 show that, with the parameters and sampled batch fixed, the transformation enters each token's probability-gradient expression through $f'(\rho)$, or equivalently its log-probability-gradient expression through $\rho f'(\rho)$. Replacing f with g therefore changes the relative weights of FAP token contributions across ratio values. In particular, the intermediate-ratio derivative no longer decays as it does under f. This identifies the change to the learning signal; it does not by itself establish an ordering of the aggregate gradient norm or a performance guarantee. The resulting training behavior is evaluated below.

Experimental protocol. We compare the original and modified transformations using Qwen3-8B on the 1,000-example TravelPlanner validation set, following the setup in Section 4.1. The initial and second-round rollout counts remain $n = 8$ and $k = 8$. Following the three-run reporting protocol in Appendix E, solid curves show pointwise means and shaded regions show pointwise minimum–maximum ranges. The lower horizontal axes report training steps, and the upper axes report cumulative rollouts. Our analysis concerns the red original-FBOS-RL curves and the gray modified-transformation curves.

Overall success and difficulty breakdown. Figure 27 shows that the modified transformation attains higher mean Final Pass Rate through most of training after the initial stage. Its curve reaches approximately 0.8 later in training, whereas the original transformation remains around 0.6–0.7 over much of that period. The modified variant also exhibits a narrower late-training min–max band. These are approximate readings of the plotted trajectories, rather than separately computed checkpoint statistics. Figure 28 shows that the advantage is smaller on easy tasks, where the curves approach each other, and more persistent on medium and hard tasks. On the hard subset, the modified curve reaches roughly 0.7 later in training, while the original remains around 0.45–0.55. The overall improvement is therefore accompanied by better success on the more difficult planning instances.

Constraint satisfaction and variability. Figure 29 reports Commonsense Micro, Commonsense Macro, Hard Micro, and Hard Macro pass rates. The modified variant shows more sustained late-training improvements in Commonsense Micro and both hard-constraint metrics. Its Commonsense Micro curve also avoids the pronounced drops and wide bands visible for the original transformation. Commonsense Macro provides a smaller separation, with the original approaching the modified variant toward the end. Accordingly, the benefit is not equally large across all metrics. The narrower observed bands indicate less variation across the three reported runs in several metrics; they are not confidence intervals or a statistical significance test.

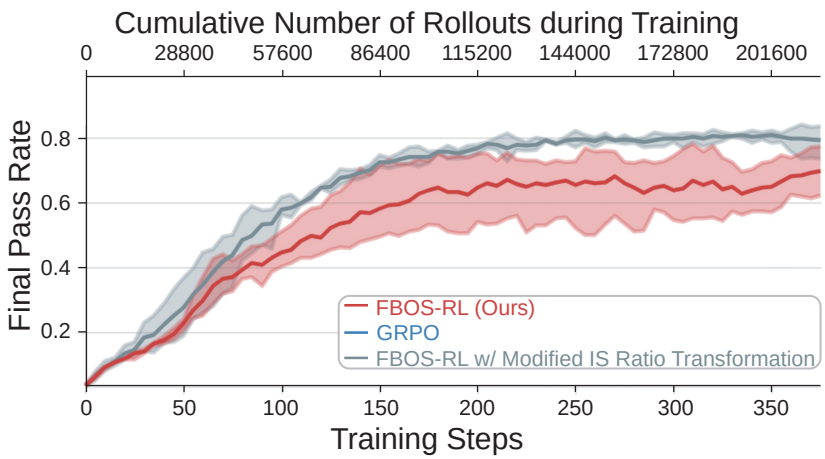


Figure 27: Overall Final Pass Rate on TravelPlanner with Qwen3-8B. Red denotes FBOS-RL using f; gray denotes FBOS-RL w/ Modified IS Ratio Transformation using g. Solid curves and shaded bands denote three-run means and min–max ranges.

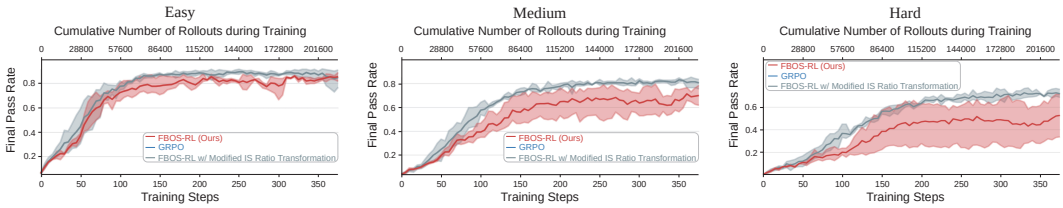


Figure 28: Qwen3-8B Final Pass Rate on easy, medium, and hard TravelPlanner instances, from left to right. The original and modified transformations use the same colors, three-run aggregation, and axes as Figure 27.

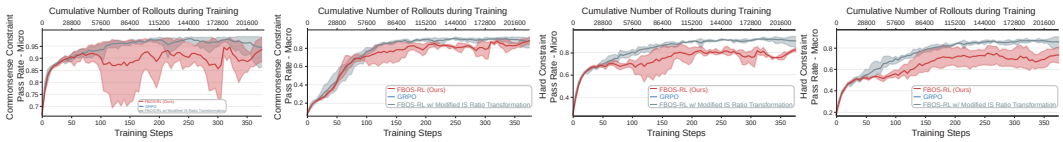


Figure 29: Qwen3-8B constraint pass rates on TravelPlanner. From left to right: Commonsense Micro, Commonsense Macro, Hard Micro, and Hard Macro. Colors, three-run aggregation, and horizontal axes follow Figure 27.

Together, these results show that the choice of transformation matters even when the cross-prompt ratio and the two training objectives are retained. The modified function g improves validation success and several constraint metrics in this setting. The gradient identities explain which token-level weights change, while the curves provide the empirical evidence for the benefit over the reported training horizon.

M Exact Gradients of the FAP Branch in EPA

We derive the gradient of $\mathcal{L}_{\mathrm{EPA}}^{\mathrm{FAP}}$ from a single token to the complete sampled batch. The results apply to the FAP surrogate in Section 3.3; the same statements and proofs hold with f replaced by any differentiable transformation, including g in Equation E.10.

M.1 Fixed Data and the Empirical Objective

Let B be a nonempty finite batch of original prompts, counting repeated prompts separately. Each occurrence of q has its own n initial responses, FAPs $\tilde{q}_i$, and k second-round responses per FAP; we suppress this additional dependence on q in the indices. Let $n, k \geq 1$ and retain $N = n + nk$.

All response lengths are positive and count valid response tokens. The advantage $\hat{A}^{\mathrm{EPA}}(\mathrm{ans}^2_{i,j})$ is computed from the full mixed group $\mathcal{G}_q$, including both initial and second-round responses, as in Section 3.3.

Standing assumptions. During differentiation, fix the sampled prompts, feedback, tokens and prefixes, lengths, rewards, mixed-group advantages, and the old-policy probabilities. Assume that the current token probabilities are positive and differentiable in the finite-dimensional parameter vector θ, the cached old-policy probabilities are positive, and f is differentiable at every ratio under consideration and participates in backpropagation. In particular, for every parameter coordinate θ_d,

$$\frac{\partial \hat{A}^{\mathrm{EPA}}(\mathrm{ans}^2_{i,j})}{\partial \theta_d} = 0, \qquad \frac{\partial \pi_{\theta_{\mathrm{old}}}(\mathrm{ans}^2_{i,j,t} \mid \tilde{q}_i, \mathrm{ans}^2_{i,j,<t})}{\partial \theta_d} = 0, \\ \frac{\partial N}{\partial \theta_d} = 0, \qquad \frac{\partial |\mathrm{ans}^2_{i,j}|}{\partial \theta_d} = 0. \tag{M.1}$$

The frozen advantage still weights the gradient; freezing it only excludes differentiation through sampling, verification, and advantage computation.

For a second-round token, the ratio is

$$\rho^2_{i,j,t}(\theta) = \frac{\pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t})}{\pi_{\theta_{\mathrm{old}}}(\mathrm{ans}^2_{i,j,t} \mid \tilde{q}_i, \mathrm{ans}^2_{i,j,<t})}. \tag{M.2}$$

Thus the differentiated numerator conditions on the original prompt q, while the frozen denominator conditions on the FAP $\tilde{q}_i$ that generated the response. Define the empirical FAP loss by

$$\widehat{\mathcal{L}}^{\mathrm{FAP}}_{\mathrm{EPA},B}(\theta) = -\frac{1}{|B|} \sum_{q \in B} \frac{1}{N} \sum_{i=1}^{n} \sum_{j=1}^{k} \frac{1}{|\mathrm{ans}^2_{i,j}|} \sum_{t=1}^{|\mathrm{ans}^2_{i,j}|} f\big(\rho^2_{i,j,t}(\theta)\big)\, \hat{A}^{\mathrm{EPA}}(\mathrm{ans}^2_{i,j}). \tag{M.3}$$

The hat distinguishes this finite-sample loss from the population expectation. The factor $1/N$ retains the FAP branch's weight within EPA; it is not replaced by $1/(nk)$. Tokens are averaged within each response before responses and prompts are aggregated.

M.2 Token-level gradient

Theorem 1 (Exact gradient of a FAP token loss)**.** *Under the standing assumptions, fix a sampled prompt and indices i, j, t, and define*

$$\ell^{\mathrm{FAP}}_{i,j,t}(\theta) = -f\big(\rho^2_{i,j,t}(\theta)\big)\, \hat{A}^{\mathrm{EPA}}(\mathrm{ans}^2_{i,j}). \tag{M.4}$$

Its parameter gradient has the following two equivalent forms:

$$\nabla_\theta \ell^{\mathrm{FAP}}_{i,j,t}(\theta) = -\hat{A}^{\mathrm{EPA}}(\mathrm{ans}^2_{i,j}) f'\big(\rho^2_{i,j,t}(\theta)\big)\, \frac{\nabla_\theta \pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t})}{\pi_{\theta_{\mathrm{old}}}(\mathrm{ans}^2_{i,j,t} \mid \tilde{q}_i, \mathrm{ans}^2_{i,j,<t})}, \tag{M.5}$$

$$= -\hat{A}^{\mathrm{EPA}}(\mathrm{ans}^2_{i,j}) f'\big(\rho^2_{i,j,t}(\theta)\big)\, \rho^2_{i,j,t}(\theta) \nabla_\theta \log \pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t}). \tag{M.6}$$

Proof. **Step 1: Differentiate the loss with respect to the ratio.** Because the advantage is fixed, Equation M.4 gives

$$\frac{\partial \ell^{\mathrm{FAP}}_{i,j,t}(\theta)}{\partial \rho^2_{i,j,t}(\theta)} = -\hat{A}^{\mathrm{EPA}}(\mathrm{ans}^2_{i,j}) f'\big(\rho^2_{i,j,t}(\theta)\big). \tag{M.7}$$

Step 2: Differentiate the ratio with respect to the current probability. The denominator of Equation M.2 is a fixed positive constant, so

$$\frac{\partial \rho^2_{i,j,t}(\theta)}{\partial \pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t})} = \frac{1}{\pi_{\theta_{\mathrm{old}}}(\mathrm{ans}^2_{i,j,t} \mid \tilde{q}_i, \mathrm{ans}^2_{i,j,<t})}. \tag{M.8}$$

Combining the two derivatives by the chain rule yields

$$\begin{aligned}\frac{\partial \ell^{\mathrm{FAP}}_{i,j,t}(\theta)}{\partial \pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t})} &= \frac{\partial \ell^{\mathrm{FAP}}_{i,j,t}(\theta)}{\partial \rho^2_{i,j,t}(\theta)} \frac{\partial \rho^2_{i,j,t}(\theta)}{\partial \pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t})} \\ &= -\frac{\hat{A}^{\mathrm{EPA}}(\mathrm{ans}^2_{i,j}) f'\big(\rho^2_{i,j,t}(\theta)\big)}{\pi_{\theta_{\mathrm{old}}}(\mathrm{ans}^2_{i,j,t} \mid \tilde{q}_i, \mathrm{ans}^2_{i,j,<t})}.\end{aligned} \tag{M.9}$$

Step 3: Differentiate with respect to each parameter coordinate. For every θ_d, another application of the chain rule gives

$$\begin{aligned}\frac{\partial \ell^{\mathrm{FAP}}_{i,j,t}(\theta)}{\partial \theta_d} &= \frac{\partial \ell^{\mathrm{FAP}}_{i,j,t}(\theta)}{\partial \pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t})} \frac{\partial \pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t})}{\partial \theta_d} \\ &= -\hat{A}^{\mathrm{EPA}}(\mathrm{ans}^2_{i,j}) f'\big(\rho^2_{i,j,t}(\theta)\big) \frac{\partial \pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t})/\partial \theta_d}{\pi_{\theta_{\mathrm{old}}}(\mathrm{ans}^2_{i,j,t} \mid \tilde{q}_i, \mathrm{ans}^2_{i,j,<t})}.\end{aligned} \tag{M.10}$$

Collecting these partial derivatives into a vector proves Equation M.5.

Step 4: Convert to the log-probability gradient. Positivity of the current probability gives

$$\nabla_\theta \log \pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t}) = \frac{\nabla_\theta \pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t})}{\pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t})}. \tag{M.11}$$

Multiplying by the current probability and dividing by the old-policy probability therefore gives

$$\begin{aligned}\frac{\nabla_\theta \pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t})}{\pi_{\theta_{\mathrm{old}}}(\mathrm{ans}^2_{i,j,t} \mid \tilde{q}_i, \mathrm{ans}^2_{i,j,<t})} &= \frac{\pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t})}{\pi_{\theta_{\mathrm{old}}}(\mathrm{ans}^2_{i,j,t} \mid \tilde{q}_i, \mathrm{ans}^2_{i,j,<t})} \nabla_\theta \log \pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t}) \\ &= \rho^2_{i,j,t}(\theta) \nabla_\theta \log \pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t}).\end{aligned} \tag{M.12}$$

Substitution into Equation M.5 proves Equation M.6. □

M.3 The complete finite-batch FAP gradient

Theorem 2 (Exact gradient of the empirical FAP branch)**.** *Under the standing assumptions, the gradient of Equation M.3 is*

$$\begin{aligned}\nabla_\theta \widehat{\mathcal{L}}^{\mathrm{FAP}}_{\mathrm{EPA},B}(\theta) = -\frac{1}{|B|} \sum_{q \in B} \frac{1}{N} \sum_{i=1}^{n} \sum_{j=1}^{k} \frac{1}{|\mathrm{ans}^2_{i,j}|} \sum_{t=1}^{|\mathrm{ans}^2_{i,j}|} \\ \hat{A}^{\mathrm{EPA}}(\mathrm{ans}^2_{i,j}) f'\big(\rho^2_{i,j,t}(\theta)\big) \frac{\nabla_\theta \pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t})}{\pi_{\theta_{\mathrm{old}}}(\mathrm{ans}^2_{i,j,t} \mid \tilde{q}_i, \mathrm{ans}^2_{i,j,<t})}.\end{aligned} \tag{M.13}$$

Equivalently,

$$\begin{aligned}\nabla_\theta \widehat{\mathcal{L}}^{\mathrm{FAP}}_{\mathrm{EPA},B}(\theta) = -\frac{1}{|B|} \sum_{q \in B} \frac{1}{N} \sum_{i=1}^{n} \sum_{j=1}^{k} \frac{1}{|\mathrm{ans}^2_{i,j}|} \sum_{t=1}^{|\mathrm{ans}^2_{i,j}|} \\ \hat{A}^{\mathrm{EPA}}(\mathrm{ans}^2_{i,j}) f'\big(\rho^2_{i,j,t}(\theta)\big) \rho^2_{i,j,t}(\theta) \nabla_\theta \log \pi_\theta(\mathrm{ans}^2_{i,j,t} \mid q, \mathrm{ans}^2_{i,j,<t}).\end{aligned} \tag{M.14}$$

Each transformation derivative is evaluated at its own token ratio inside the complete sum.

Proof. **Step 1: Restore averaging over the tokens of one response.** The response and its length are fixed, so differentiation is linear over the finite token sum:

$$\begin{aligned}\nabla_\theta \left[\frac{1}{|\mathrm{ans}^2_{i,j}|} \sum_{t=1}^{|\mathrm{ans}^2_{i,j}|} \ell^{\mathrm{FAP}}_{i,j,t}(\theta) \right] &= \frac{1}{|\mathrm{ans}^2_{i,j}|} \nabla_\theta \left[\sum_{t=1}^{|\mathrm{ans}^2_{i,j}|} \ell^{\mathrm{FAP}}_{i,j,t}(\theta) \right] \\ &= \frac{1}{|\mathrm{ans}^2_{i,j}|} \sum_{t=1}^{|\mathrm{ans}^2_{i,j}|} \nabla_\theta \ell^{\mathrm{FAP}}_{i,j,t}(\theta).\end{aligned} \tag{M.15}$$

Step 2: Restore all responses and FAPs for one prompt. Since n, k, N are fixed, apply the same finite-sum rule over j and i:

$$\begin{aligned}\nabla_\theta \left[\frac{1}{N} \sum_{i=1}^{n} \sum_{j=1}^{k} \frac{1}{|\mathrm{ans}_{i,j}^2|} \sum_{t=1}^{|\mathrm{ans}_{i,j}^2|} \ell_{i,j,t}^{\mathrm{FAP}}(\theta) \right] \\ = \frac{1}{N} \sum_{i=1}^{n} \sum_{j=1}^{k} \frac{1}{|\mathrm{ans}_{i,j}^2|} \sum_{t=1}^{|\mathrm{ans}_{i,j}^2|} \nabla_\theta \ell_{i,j,t}^{\mathrm{FAP}}(\theta).\end{aligned} \tag{M.16}$$

This retains both the answer-length normalization and the original factor $1/N$.

Step 3: Restore averaging over the sampled prompt batch. Using Equation M.4, including its minus sign, rewrite the empirical loss as

$$\widehat{\mathcal{L}}_{\mathrm{EPA},B}^{\mathrm{FAP}}(\theta) = \frac{1}{|B|} \sum_{q \in B} \left[\frac{1}{N} \sum_{i=1}^{n} \sum_{j=1}^{k} \frac{1}{|\mathrm{ans}_{i,j}^2|} \sum_{t=1}^{|\mathrm{ans}_{i,j}^2|} \ell_{i,j,t}^{\mathrm{FAP}}(\theta) \right]. \tag{M.17}$$

The batch is finite and fixed. Differentiating and substituting Equation M.16 gives

$$\nabla_\theta \widehat{\mathcal{L}}_{\mathrm{EPA},B}^{\mathrm{FAP}}(\theta) = \frac{1}{|B|} \sum_{q \in B} \left[\frac{1}{N} \sum_{i=1}^{n} \sum_{j=1}^{k} \frac{1}{|\mathrm{ans}_{i,j}^2|} \sum_{t=1}^{|\mathrm{ans}_{i,j}^2|} \nabla_\theta \ell_{i,j,t}^{\mathrm{FAP}}(\theta) \right]. \tag{M.18}$$

Step 4: Substitute each token's exact gradient. Substituting Equation M.5 into every summand yields Equation M.13. Substituting Equation M.6 instead yields Equation M.14. No derivative–expectation interchange or parameter-update approximation is used. □

M.4 Relation to the population objective

The empirical gradient above is exact for the realized batch. Relating it to the expectation in Section 3.3 requires additional assumptions about the sampling law and integrability.

Corollary 1 (Population FAP gradient under a frozen sampling law)**.** *Let the law of prompts, responses, and feedback be fixed by $\mathcal{D}$, θ_{old}, and the verifier, independently of the current θ. Suppose the per-prompt FAP loss is integrable at θ_0, is continuously differentiable almost surely on a neighborhood of θ_0, and its gradient norm throughout that neighborhood is bounded by an integrable random variable. Then, at θ_0,*

$$\begin{aligned}\nabla_\theta \mathcal{L}_{\mathrm{EPA}}^{\mathrm{FAP}}(\theta) = -\mathbb{E}\Bigg[\frac{1}{N} \sum_{i=1}^{n} \sum_{j=1}^{k} \frac{1}{|\mathrm{ans}_{i,j}^2|} \sum_{t=1}^{|\mathrm{ans}_{i,j}^2|} \\ \hat{A}^{\mathrm{EPA}}(\mathrm{ans}_{i,j}^2) f'\big(\rho_{i,j,t}^2(\theta)\big)\, \rho_{i,j,t}^2(\theta) \nabla_\theta \log \pi_\theta(\mathrm{ans}_{i,j,t}^2 \mid q, \mathrm{ans}_{i,j,<t}^2) \Bigg].\end{aligned} \tag{M.19}$$

Proof. **Step 1: Identify the random finite-sample loss.** For a sampled prompt and its two rounds of responses, write

$$H(\theta) = \frac{1}{N} \sum_{i=1}^{n} \sum_{j=1}^{k} \frac{1}{|\mathrm{ans}_{i,j}^2|} \sum_{t=1}^{|\mathrm{ans}_{i,j}^2|} \ell_{i,j,t}^{\mathrm{FAP}}(\theta).$$

By definition, $\mathcal{L}_{\mathrm{EPA}}^{\mathrm{FAP}}(\theta) = \mathbb{E}[H(\theta)]$. Its sampling law is independent of the differentiated parameters.

Step 2: Justify differentiation under the expectation. Let e_d denote a coordinate vector and take sufficiently small nonzero s so that the segment from θ_0 to $\theta_0 + se_d$ stays in the neighborhood. By the one-dimensional mean value bound along this segment,

$$\left| \frac{H(\theta_0 + se_d) - H(\theta_0)}{s} \right| \le \sup_{\vartheta} \|\nabla_\theta H(\vartheta)\| \le M, \qquad \mathbb{E}[M] < \infty,$$

where the supremum is over that segment. The difference quotient converges almost surely to $\partial H(\theta_0)/\partial\theta_d$. The integrable bound allows its limit to pass through the expectation, giving

$$\frac{\partial}{\partial\theta_d}\mathbb{E}[H(\theta)]\bigg|_{\theta=\theta_0} = \mathbb{E}\left[\frac{\partial H(\theta_0)}{\partial\theta_d}\right].$$

Applying this argument to every coordinate yields $\nabla_\theta\mathbb{E}[H] = \mathbb{E}[\nabla_\theta H]$ at θ_0.

Step 3: Substitute the finite-sum derivative. Equation M.16 gives the derivative of H, and Equation M.6 expands every token gradient. Substitution gives Equation M.19. □

Scope. Theorems 1 and 2 require no exchange of differentiation and expectation. A realized batch gradient need not numerically equal the population gradient in Corollary 1. These results concern the mathematical FAP surrogate with the response and prompt normalization written above. For the implementation's clamped log-ratio, the displayed ratio derivatives apply where the clamp is inactive; saturation also introduces the clamp's derivative. The results characterize the FAP learning signal and do not assert monotone training progress or improved validation performance.

N Ablation Study of Mutual Reinforcement between EPA and ECC

To demonstrate that training Objective 1 (Exploitation-oriented Policy Alignment, EPA) and Objective 2 (Exploration-oriented Capability Cultivation, ECC) can mutually reinforce each other, forming a positive flywheel (bootstrapping) effect, we conduct in-depth experiments on the Qwen3-14B model.

N.1 Objective 2 (ECC) Boosts Objective 1 (EPA)

We design a baseline that only optimizes Objective 1 (EPA) during training.

Objective 2 (ECC) trains $\pi_\theta(\cdot \mid \tilde{q})$ to sample higher-quality rollouts, where $\tilde{q}$ denotes a Feedback-Augmented Prompt (FAP). These high-quality rollouts are then injected into the GRPO group used by Objective 1 (EPA). In this way, Objective 2 (ECC) effectively raises the quality of the rollouts that Objective 1 (EPA) leverages during the sampling stage, thereby boosting Objective 1 (EPA) (and significantly raising the ceiling of policy alignment). (ECC, which improves the sampling capability of $\pi_\theta(\cdot \mid \tilde{q})$, provides EPA, which exploits the already-sampled high-quality rollouts, with high-quality gradient guidance: it teaches the model how to produce better rollouts based on feedback.)

Figure 30 reports, at each training step on the training set, the average quality and the best (max) quality of rollouts sampled from $\pi_\theta(\cdot \mid \tilde{q})$.

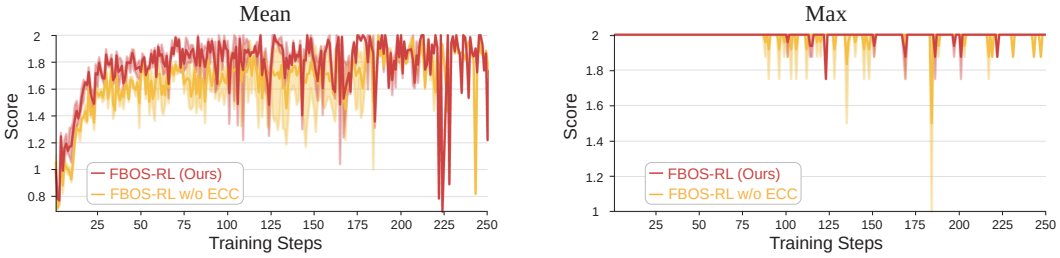


Figure 30: Mean quality (left) and max quality (right) of rollouts sampled from $\pi_\theta(\cdot \mid \tilde{q})$ with Qwen3-14B at each training step on the training set: our method vs. the baseline that only trains Objective 1 (EPA).

We further analyze rollouts sampled from $\pi_\theta(\cdot \mid \tilde{q})$ by difficulty level in Figure 31 and Figure 32. Figure 31 reports the mean quality of rollouts sampled from $\pi_\theta(\cdot \mid \tilde{q})$, broken down by difficulty level. Figure 32 reports the max quality of rollouts sampled from $\pi_\theta(\cdot \mid \tilde{q})$, broken down by difficulty level.

Figures 30, 31, and 32 show that introducing Objective 2 (ECC) leads to higher-quality rollouts sampled from $\pi_\theta(\cdot \mid \tilde{q})$, with both mean and max quality improved across every difficulty level.

Figure 33 reports, at each training step on the training set, the average quality of rollouts generated during the sampling phase, including initial sampling from $\pi_\theta(\cdot \mid q)$ and second-round sampling from $\pi_\theta(\cdot \mid \tilde{q})$.

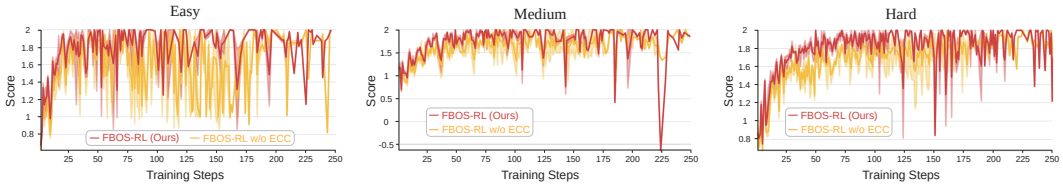


Figure 31: Mean quality of rollouts sampled from $\pi_\theta(\cdot \mid \tilde{q})$ on the training set, broken down by difficulty level: easy (left), medium (middle), and hard (right). Our method vs. the baseline that only trains Objective 1 (EPA).

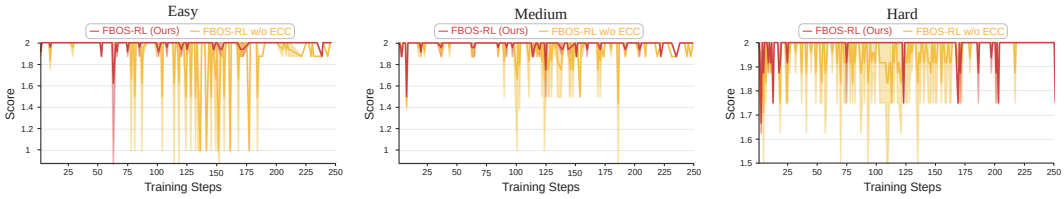


Figure 32: Max quality of rollouts sampled from $\pi_\theta(\cdot \mid \tilde{q})$ on the training set, broken down by difficulty level: easy (left), medium (middle), and hard (right). Our method vs. the baseline that only trains Objective 1 (EPA).

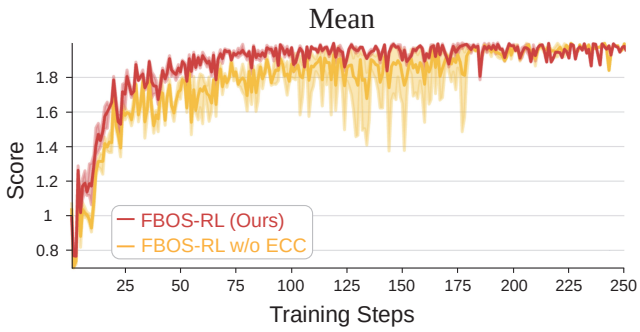


Figure 33: Mean quality of rollouts generated during the entire sampling phase (initial sampling from $\pi_\theta(\cdot \mid q)$ and second-round sampling from $\pi_\theta(\cdot \mid \tilde{q})$ combined) at each training step on the training set: our method vs. the baseline that only trains Objective 1 (EPA).

We further report the mean and max quality of sampling-phase rollouts for each difficulty level in Figure 34 and Figure 35. Figure 34 reports the mean quality of sampling-phase rollouts at each difficulty level. Figure 35 reports the max quality of sampling-phase rollouts at each difficulty level.

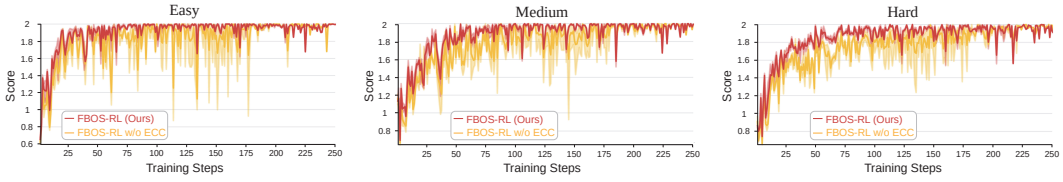


Figure 34: Mean quality of rollouts generated during the sampling phase on the training set, broken down by difficulty level: easy (left), medium (middle), and hard (right). Our method vs. the baseline that only trains Objective 1 (EPA).

Figures 33, 34, and 35 show that introducing Objective 2 (ECC) significantly improves the quality of rollouts discovered by the model during the sampling phase.

Figure 4 shows that our method significantly outperforms the baseline on the validation set. Figure 36 reports, during training, the final pass rate on the validation set for each difficulty level ("easy", "medium", "hard"). As difficulty increases, the lead of our method grows larger. Figure 37 reports four metrics on the validation set during training, arranged from left to right: Commonsense Constraint Pass Rate (Micro), Commonsense Constraint Pass Rate (Macro), Hard Constraint Pass Rate (Micro),

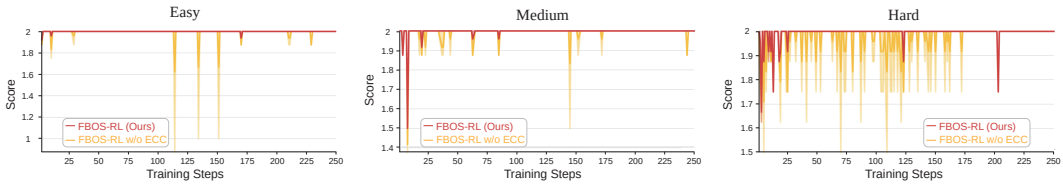


Figure 35: Max quality of rollouts generated during the sampling phase on the training set, broken down by difficulty level: easy (left), medium (middle), and hard (right). Our method vs. the baseline that only trains Objective 1 (EPA).

and Hard Constraint Pass Rate (Macro). Our method significantly outperforms the baseline on all four metrics.

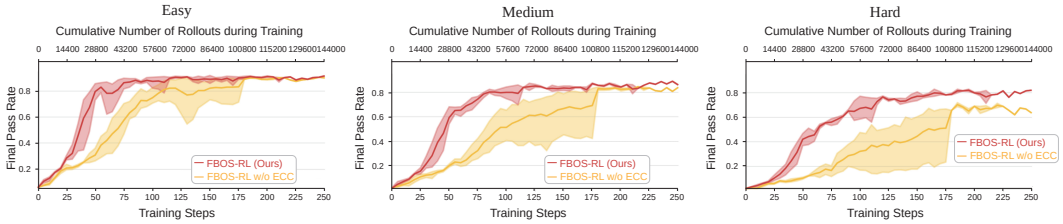


Figure 36: Final pass rate on the TravelPlanner validation set across different difficulty levels: easy (left), medium (middle), and hard (right). Our method vs. the baseline that only trains Objective 1 (EPA).

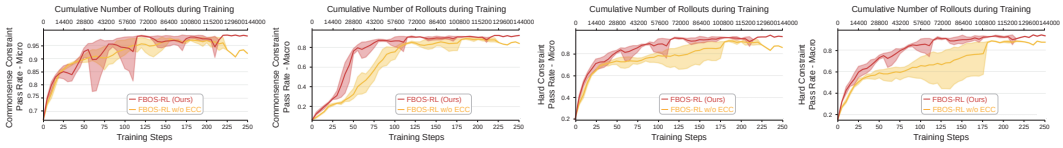


Figure 37: Commonsense and hard constraint pass rates on the TravelPlanner validation set during training. From left to right: Commonsense Constraint Pass Rate (Micro), Commonsense Constraint Pass Rate (Macro), Hard Constraint Pass Rate (Micro), and Hard Constraint Pass Rate (Macro). Each panel compares our method with the baseline that only trains Objective 1 (EPA). The bottom and top x-axes show training steps and cumulative rollouts, respectively.

These results demonstrate that Objective 2 (ECC) can effectively boost Objective 1 (EPA).

N.2 Objective 1 (EPA) Boosts Objective 2 (ECC)

We design a baseline that only optimizes Objective 2 (ECC) during training.

Figure 38 reports, at each training step on the training set, the average quality of rollouts generated by the model during the sampling phase, including initial sampling from $\pi_\theta(\cdot \mid q)$ and second-round sampling from $\pi_\theta(\cdot \mid \tilde{q})$. Figure 39 further reports the mean quality of sampling-phase rollouts at each difficulty level.

Figures 38 and 39 show that introducing Objective 1 (EPA) significantly improves the quality of rollouts discovered by the model during the sampling phase.

Figure 5 shows that our method significantly outperforms the baseline on the validation set. Figure 40 reports, during training, the final pass rate on the validation set for each difficulty level ("easy", "medium", "hard"). Figure 41 reports four metrics on the validation set during training, arranged from left to right: Commonsense Constraint Pass Rate (Micro), Commonsense Constraint Pass Rate (Macro), Hard Constraint Pass Rate (Micro), and Hard Constraint Pass Rate (Macro). Our method significantly outperforms the baseline on all four metrics.

These results demonstrate that Objective 1 (EPA) can effectively boost Objective 2 (ECC).

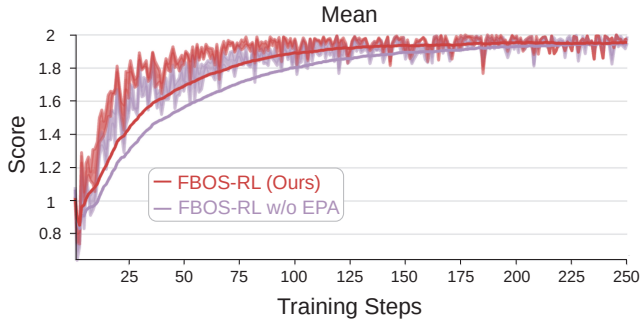


Figure 38: Mean quality of rollouts generated during the entire sampling phase at each training step on the training set: our method vs. the baseline that only trains Objective 2 (ECC).

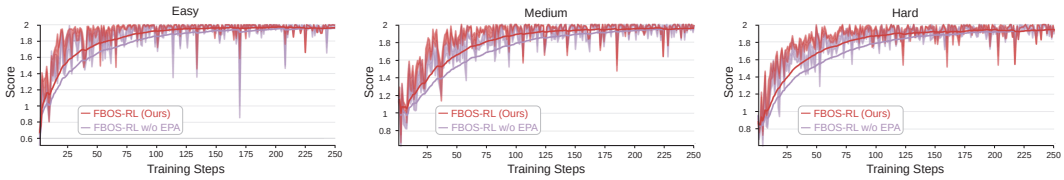


Figure 39: Mean quality of rollouts generated during the sampling phase on the training set, broken down by difficulty level: easy (left), medium (middle), and hard (right). Our method vs. the baseline that only trains Objective 2 (ECC).

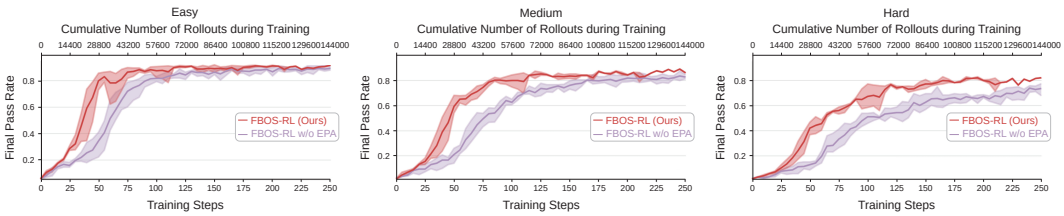


Figure 40: Final pass rate on the TravelPlanner validation set across different difficulty levels: easy (left), medium (middle), and hard (right). Our method vs. the baseline that only trains Objective 2 (ECC).

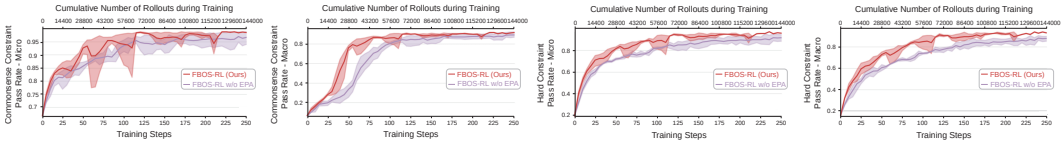


Figure 41: Commonsense and hard constraint pass rates on the TravelPlanner validation set during training. From left to right: Commonsense Constraint Pass Rate (Micro), Commonsense Constraint Pass Rate (Macro), Hard Constraint Pass Rate (Micro), and Hard Constraint Pass Rate (Macro). Each panel compares our method with the baseline that only trains Objective 2 (ECC). The bottom and top x-axes show training steps and cumulative rollouts, respectively.

O Controlling for the Number of Parameter Updates

FBOS-RL performs an EPA update followed by an ECC update in each sampling-and-optimization iteration, whereas standard GRPO performs one update. We control for this difference using GRPO w/ Extra Update.

EPA uses all 72 responses per original prompt, or 576 responses for the batch of 8 prompts. ECC then reuses the 64 second-round responses per original prompt, or 512 responses per batch. Thus, the two updates use 1,088 response instances in total but only 576 newly generated rollouts. *GRPO w/ Extra Update* first updates on all 72 responses per original prompt and then randomly selects 64 of those responses without replacement for a second update. It therefore matches the two update

stages and the 576/512 response counts without generating additional rollouts. Input lengths and computation can still differ, as discussed in Appendix R.1.

We conduct this controlled experiment on Qwen3-14B and Llama-3.1-8B-Instruct using the same experimental protocol. The two settings differ only in the underlying base model. In each setting, we compare FBOS-RL with vanilla GRPO and GRPO w/ Extra Update.

Figure 42 reports the final pass rate on the TravelPlanner validation set, with Qwen3-14B on the left and Llama-3.1-8B-Instruct on the right. Figure 43 breaks down the final pass rate by difficulty, with Qwen3-14B in the top row and Llama-3.1-8B-Instruct in the bottom row, and easy, medium, and hard subsets arranged from left to right. Figure 44 reports the four constraint pass rate metrics using the same model ordering by row; from left to right, the columns show Commonsense Micro, Commonsense Macro, Hard Micro, and Hard Macro.

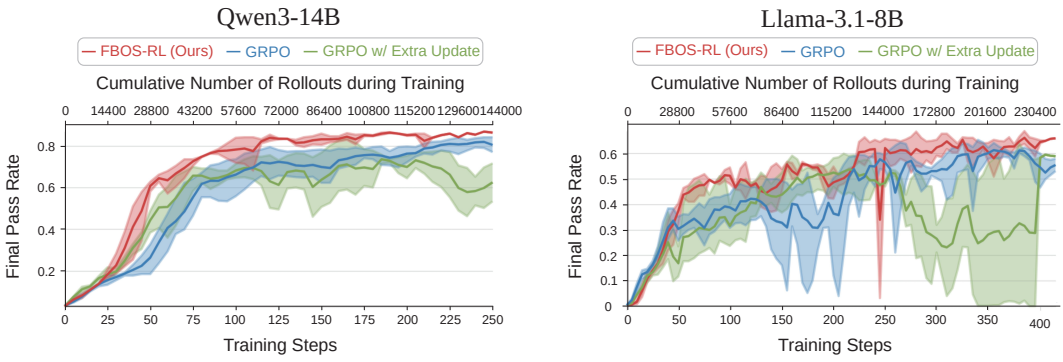


Figure 42: Final pass rate on the TravelPlanner validation set during training. Left: Qwen3-14B. Right: Llama-3.1-8B-Instruct. Each panel compares FBOS-RL, vanilla GRPO, and GRPO w/ Extra Update. The bottom and top x-axes show training steps and cumulative rollouts, respectively.

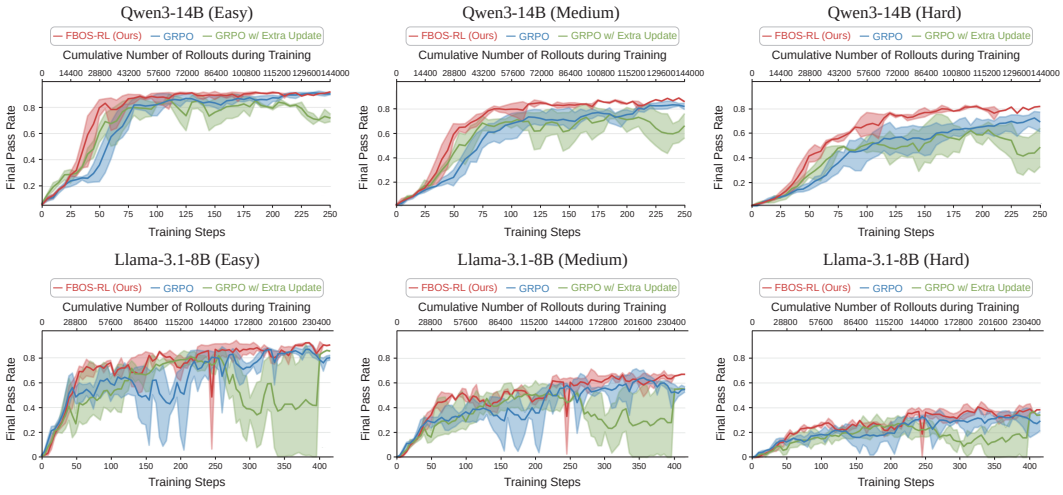


Figure 43: Final pass rate on the TravelPlanner validation set during training, broken down by difficulty level. Top row: Qwen3-14B. Bottom row: Llama-3.1-8B-Instruct. Columns from left to right: easy, medium, and hard. Each panel compares FBOS-RL, vanilla GRPO, and GRPO w/ Extra Update. The bottom and top x-axes show training steps and cumulative rollouts, respectively.

Figures 42, 43, and 44 show that FBOS-RL outperforms the GRPO w/ Extra Update baseline on both Qwen3-14B and Llama-3.1-8B-Instruct, with gains in overall final pass rate, difficulty-specific final pass rates, and constraint pass rates. These results indicate that the gains of FBOS-RL cannot be explained solely by the additional parameter update introduced by ECC.

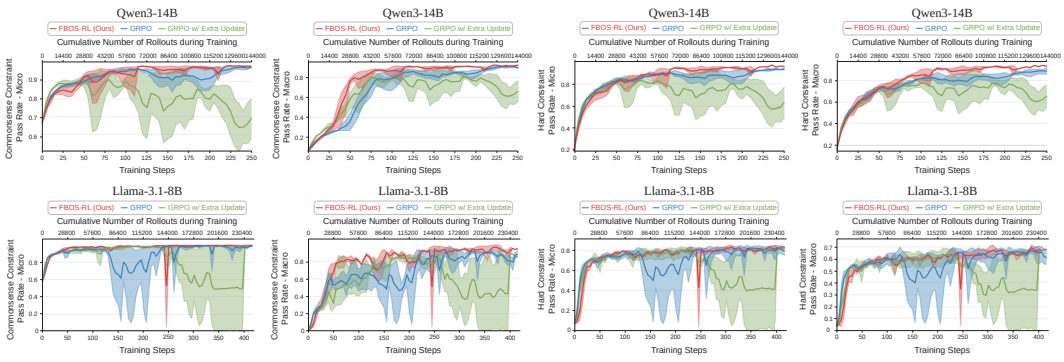


Figure 44: Commonsense and hard constraint pass rates on the TravelPlanner validation set during training. Top row: Qwen3-14B. Bottom row: Llama-3.1-8B-Instruct. Columns from left to right: Commonsense Constraint Pass Rate (Micro), Commonsense Constraint Pass Rate (Macro), Hard Constraint Pass Rate (Micro), and Hard Constraint Pass Rate (Macro). Each panel compares FBOS-RL, vanilla GRPO, and GRPO w/ Extra Update. The bottom and top x-axes show training steps and cumulative rollouts, respectively.

P TRAINING DYNAMICS OF THE EPA OBJECTIVE

To empirically demonstrate that the *Exploitation-oriented Policy Alignment* (EPA) objective introduced in Section 3.3 can be trained in a stable and effective manner, we report its training-set score and the corresponding standard deviation (std) as a function of training steps on two policy models, Llama-3.1-8B-Instruct (Figures 45 and 46) and Qwen3-14B (Figures 47 and 48). Across both models, the training-set score of EPA exhibits a clear and stable upward trend while its std steadily decreases, which together confirm that EPA can be properly optimized. Moreover, this positive trend consistently holds on training subsets of different difficulty levels (*easy*, *medium*, *hard*), indicating that EPA is able to make robust progress regardless of sample difficulty.

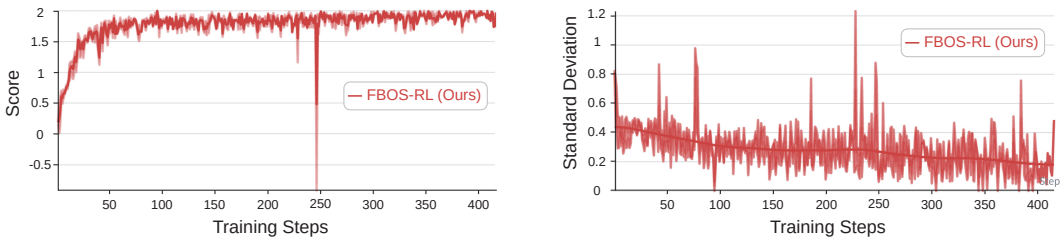


Figure 45: Training dynamics of the EPA objective on Llama-3.1-8B-Instruct. (a) Training-set score of EPA steadily increases along training steps. (b) The corresponding std steadily decreases, indicating that EPA can be optimized in a stable manner.

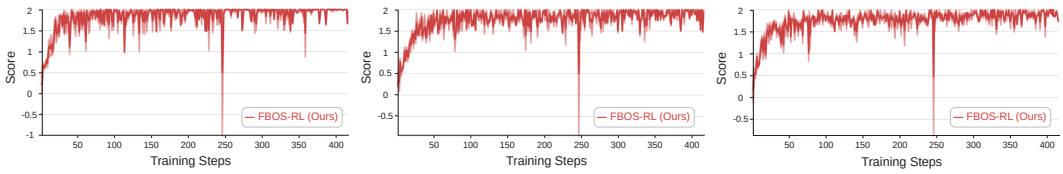


Figure 46: Training-set score of the EPA objective on Llama-3.1-8B-Instruct, broken down by training-sample difficulty (*easy*, *medium*, *hard*, from left to right). The score consistently rises along training steps across all three difficulty levels.

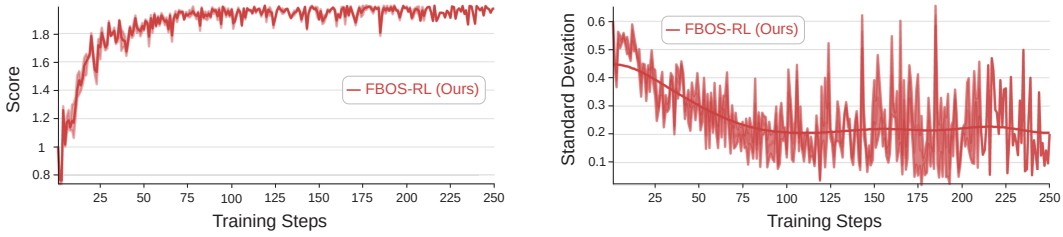


Figure 47: Training dynamics of the EPA objective on Qwen3-14B. (a) Training-set score of EPA steadily increases along training steps. (b) The corresponding std steadily decreases, indicating that EPA can be optimized in a stable manner.

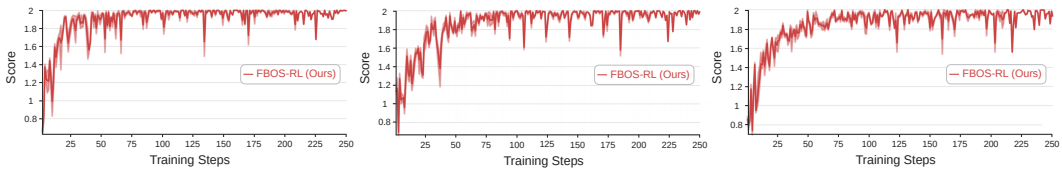


Figure 48: Training-set score of the EPA objective on Qwen3-14B, broken down by training-sample difficulty (*easy*, *medium*, *hard*, from left to right). The score consistently rises along training steps across all three difficulty levels.

Q TRAINING DYNAMICS OF THE ECC OBJECTIVE

We similarly validate the trainability of the *Exploration-oriented Capability Cultivation* (ECC) objective introduced in Section 3.3. We report the training-set score and the corresponding standard deviation (std) of ECC along training steps on Llama-3.1-8B-Instruct (Figures 49 and 50) and Qwen3-14B (Figures 51 and 52). As with EPA, we observe that the ECC training-set score steadily rises and its std steadily decreases on both models, and the same trend holds across all three difficulty levels (*easy*, *medium*, *hard*). These results demonstrate that ECC can also be properly trained, confirming that the model indeed acquires an improved ability to sample higher-quality rollouts from $\pi_\theta(\cdot \mid \tilde{q}_i)$.

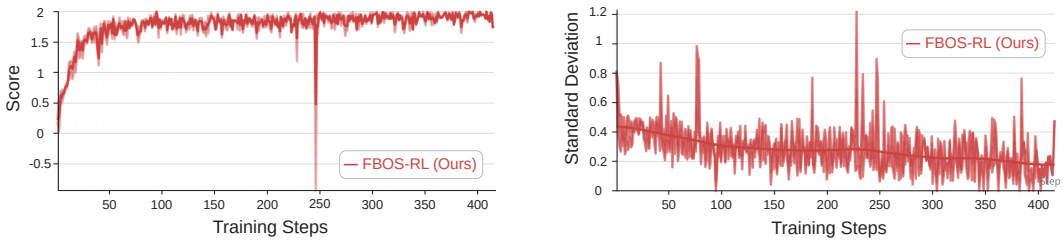


Figure 49: Training dynamics of the ECC objective on Llama-3.1-8B-Instruct. (a) Training-set score of ECC steadily increases along training steps. (b) The corresponding std steadily decreases, indicating that ECC can be optimized in a stable manner.

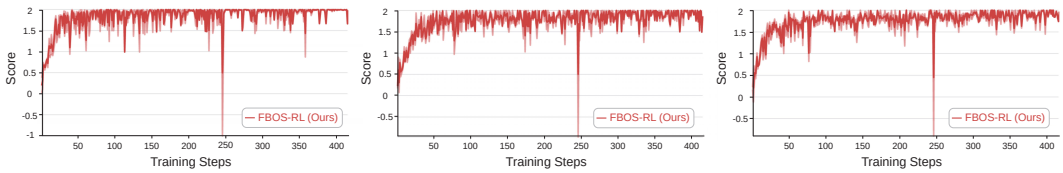


Figure 50: Training-set score of the ECC objective on Llama-3.1-8B-Instruct, broken down by training-sample difficulty (*easy*, *medium*, *hard*, from left to right). The score consistently rises along training steps across all three difficulty levels.

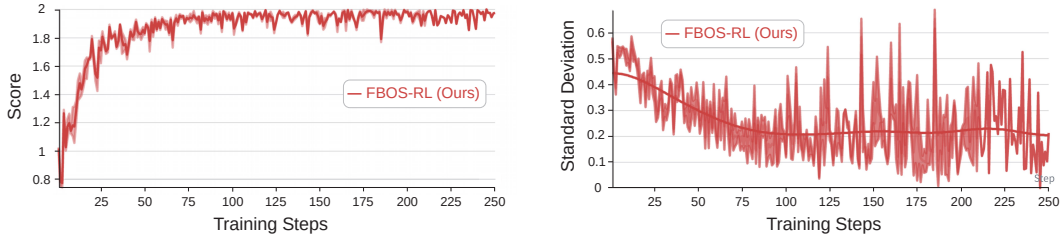


Figure 51: Training dynamics of the ECC objective on Qwen3-14B. (a) Training-set score of ECC steadily increases along training steps. (b) The corresponding std steadily decreases, indicating that ECC can be optimized in a stable manner.

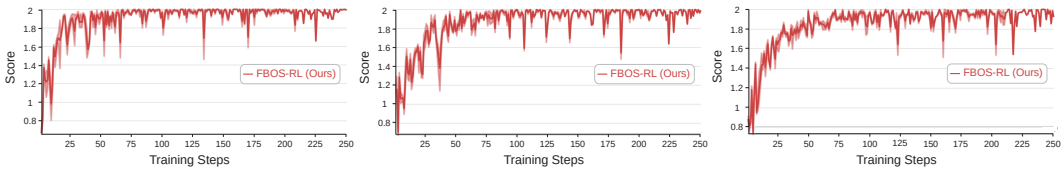


Figure 52: Training-set score of the ECC objective on Qwen3-14B, broken down by training-sample difficulty (*easy*, *medium*, *hard*, from left to right). The score consistently rises along training steps across all three difficulty levels.

R HYPERPARAMETERS

Tables 2 and 3 summarize the FBOS-RL training settings for TravelPlanner and MiniF2F-Lean4. The fixed FBOS-RL rollout sizes are $n = 8$ and $k = 8$, giving $n(1 + k) = 72$ rollouts per prompt. The matched GRPO baseline samples 72 responses per original prompt. For TravelPlanner, Qwen3-8B uses the same shared settings as Qwen3-14B, with the model-specific training durations listed below.

R.1 ROLLOUT

A rollout is one newly generated response. The first response and each second-round response are counted separately; reusing a response in another objective does not create a new rollout. For a batch B of b original prompts, GRPO generates bN_{G} responses and FBOS-RL generates $bn + bnk$. We match these counts by setting $N_{\mathrm{G}} = N = n(1 + k)$. With $b = n = k = 8$, both methods generate 576 responses per iteration. Cumulative training rollouts after t iterations are $C(t) = tbN = 576t$ in this fixed-budget setting. Validation responses are excluded from this training-sample count.

R.2 TRAVELPLANNER

Table 2: TravelPlanner training settings.

Setting	Value
Models and epochs	Qwen3-8B for 23 epochs; Qwen3-14B for 9 epochs; Llama-3.1-8B-Instruct for 21 epochs
Batching and rollouts	Prompt batch 8; PPO minibatch 8 prompts; per-GPU microbatch 1; $n = 8$; $k = 8$; 72 rollouts per prompt
Optimization	AdamW; learning rate 1×10^{-6}; $\beta = (0.9, 0.999)$; weight decay 0.01; constant schedule; zero warmup; gradient clip 1.0; PPO epoch 1; token mean; PPO clip $\varepsilon = 0.2$ and dual-clip $c = 3$ where applicable
Regularization	Entropy coefficient 0; actor KL loss and reward KL penalty disabled
Precision and system	FP16 mixed precision; gradient checkpointing; asynchronous vLLM; tensor parallelism 2 for Qwen3-8B and Qwen3-14B, and 1 for Llama; memory utilization 0.6; 1 node with 8 GPUs
Qwen3 thinking mode	Thinking disabled for Qwen3-8B and Qwen3-14B
Generation and evaluation	Temperature 1; top p 1; greedy validation; validation before training and every 5 steps; data-shuffle seed 42
Lengths	Prompt 17,218 tokens; response 15,550 tokens

For FBOS-RL, PPO clipping applies to the initial-response EPA branch and the ECC objective; the feedback-conditioned responses in EPA use the unclipped transformation $f(\rho) = \rho/(\rho + 0.1)$.

R.3 MINIF2F-LEAN4

Table 3: Qwen3.5-27B training settings for MiniF2F-Lean4.

Setting	Value
Model and data	Qwen3.5-27B with thinking disabled; validation split for training and test split for validation
Batching	Prompt batch 8; PPO minibatch 8; per-GPU microbatch 1
Lengths and rollouts	Prompt 2048; response 7952; fixed $n = 8$; $k = 8$; 72 total rollouts per prompt
Optimization	AdamW; learning rate 1×10^{-6}; $\beta = (0.9, 0.999)$; weight decay 0.01; constant schedule; warmup 0; gradient clip 1.0; PPO epoch 1; token mean; clip 0.2
Regularization and precision	Entropy coefficient 0; actor KL loss and reward KL penalty disabled; FP16; gradient checkpointing
Rollout system	Asynchronous vLLM; tensor parallelism 2; GPU memory utilization 0.6
Decoding	Training temperature 1 and top p 1; greedy validation
Schedule and seed	21 epochs; validation before training and every 5 steps; seed 42
Compute	2 nodes with 8 GPUs per node, 16 GPUs total

S EXACT PROMPT TEMPLATES

The following listings reproduce the four source prompt templates exactly. Visual line wrapping is used only for page layout and does not introduce template characters.

S.1 TRAVELPLANNER INITIAL PROMPT

```
You are a professional travel planner. Based on the Given information and
     Query, please give me a detailed plan in JSON format, similar to the
     format shown in the EXAMPLE below.
Note: Each item MUST include and only include ['days', 'current_city', '
    transportation', 'breakfast', 'attraction', 'lunch', 'dinner', '
    accommodation'].
      All the information in your plan should be derived from the Given
    information!
      You MUST pay special attention to all the constraints mentioned in
    the Given information and the user's Query, as well as implicit
    common-sense constraints that may not be explicitly stated but must
    be satisfied. You MUST ensure that the final travel plan you provide
    meets all of these constraints!

When you travel to two cities in one day, you should note it in the '
    current_city' section as in the example (i.e., from A to B).
If transportation details indicate a journey from one city to another (e.
    g., from A to B), the 'current_city' should be updated to the
    destination city (in this case, B).
If there's information about transportation, ensure that the '
    current_city' aligns with the destination mentioned in the
    transportation details (i.e., the current city should follow the
    format 'from A to B').
Also, ensure that all flight numbers and costs are followed by a colon (i.
    e., 'Flight Number:' and 'Cost:'), consistent with the provided
    example.
Use a ';' to separate different attractions, with each attraction
    formatted as 'Name, City'.
The symbol '-' indicates that information is unnecessary. For example, in
     the provided example, you do not need to plan after returning to the
     departure city.
Additionally, delete any '$' symbols!

The EXAMPLE provided is as follows:
-----EXAMPLE-----
[
    {{
        "days": 1,
        "current_city": "from Dallas to Peoria",
        "transportation": "Flight Number: 4044830, from Dallas to Peoria,
     Departure Time: 13:10, Arrival Time: 15:01",
        "breakfast": "-",
        "attraction": "Peoria Historical Society, Peoria;Peoria Holocaust
     Memorial, Peoria;",
        "lunch": "-",
        "dinner": "Tandoor Ka Zaika, Peoria",
        "accommodation": "Bushwick Music Mansion, Peoria"
    }},
    {{
        "days": 2,
        "current_city": "Peoria",
        "transportation": "-",
        "breakfast": "Tandoor Ka Zaika, Peoria",
        "attraction": "Peoria Riverfront Park, Peoria;The Peoria
    PlayHouse, Peoria;Glen Oak Park, Peoria;",
        "lunch": "Cafe Hashtag LoL, Peoria",
        "dinner": "The Curzon Room - Maidens Hotel, Peoria",
```

```
        "accommodation": "Bushwick Music Mansion, Peoria"
    }},
    {{
        "days": 3,
        "current_city": "from Peoria to Dallas",
        "transportation": "Flight Number: 4045904, from Peoria to Dallas,
     Departure Time: 07:09, Arrival Time: 09:20",
        "breakfast": "-",
        "attraction": "-",
        "lunch": "-",
        "dinner": "-",
        "accommodation": "-"
    }}
]
-----EXAMPLE END-----
The Given information and Query are as follows:
Given information: {text}
Query: {query}

Based on the Given information and Query, please give me a detailed plan
    in JSON format, similar to the format shown in the EXAMPLE.
Note: Please strictly follow the format in the EXAMPLE but do not output
    the plan based on the content provided in the EXAMPLE! Since the
    example is merely for illustration purposes, you only need to follow
    its format!!
      Before outputting the final JSON-formatted Travel Plan, you MUST
    first output your step-by-step thought process, planning, and
    reasoning according to the instructions. Then, output the final JSON-
    formatted Travel Plan after the special tag "[JSON format Travel Plan
    ]". Let's begin:
```

S.2 MINIF2F INITIAL PROMPT

```
You are an expert in Lean 4 theorem proving. Prove the following theorem
    by filling in the proof body after `:= by`.

Problem:
```lean
{prelude}{statement_without_proof}
```

Output your proof inside a single Lean code block. Only write the tactics
     that complete the proof after `:= by`. Do not repeat the theorem
    header. Example output format:

```lean
  <your tactics here>
```

Let's begin:
```

S.3 TRAVELPLANNER FEEDBACK-AUGMENTED PROMPT

```
You are a professional travel planner. Here is the JSON format Travel
    Plan you provided previously: {prev_answer}
The feedback regarding your previous answer is as follows: {comment}
You MUST pay special attention to the feedback above, as it specifies
    exactly why your previous answer failed. Treat this feedback as a set
     of original, mandatory constraints.

CRITICAL RULE:
Your entire response MUST read as a flawless, first-time execution. You
    are strictly forbidden from mentioning that this is a revision, a
    second attempt, or a response to feedback. Present your reasoning
    cleanly, exactly as if you are solving this problem for the very
    first time based on the provided requirements.

Output requirements:
First, output your reasoning and planning process. Finally, output the
    final Travel Plan in JSON format strictly after the "[JSON format
    Travel Plan]" tag.

Let's begin:
```

S.4 MINIF2F FEEDBACK-AUGMENTED PROMPT

```
You are an expert in Lean 4 theorem proving. Here is the Lean 4 proof you
     provided previously:
{prev_answer}

The feedback from the Lean 4 verifier regarding your previous proof is as
     follows:
{comment}

You MUST pay special attention to the feedback above, as it specifies
    exactly why your previous proof failed.

CRITICAL RULE:
Your entire response MUST read as a flawless, first-time execution. You
    are strictly forbidden from mentioning that this is a revision, a
    second attempt, or a response to feedback. Present your reasoning
    cleanly, exactly as if you are solving this problem for the very
    first time.

Output requirements:
First, demonstrate your reasoning process. Then, output the final Lean 4
    proof following the "[Final Lean Proof]" tag. After the tag, write
    ONLY the tactics that complete the proof after `:= by`. Do NOT repeat
    the theorem header.

Let's begin:
```