CoEx-GRPO: Collective-Experience Rollouts for Group-Relative Policy Optimization under Verifiable Rewards

Anonymous ACL submission

Abstract

Reinforcement Learning with Verifiable Rewards (RLVR) has become an effective paradigm for improving reasoning in domains where correctness can be automatically checked. Group Relative Policy Optimization (GRPO) is widely used in this setting, but its training signal can weaken over time due to *homogenized exploration*, where sampled responses for the same prompt become increasingly similar. This reduces within-group reward variance, weakens group-relative comparison, and limits further learning. We propose CoEx-GRPO, a multi-policy GRPO framework that expands the within-prompt comparison set through collective experience. CoEx-GRPO jointly trains a main policy and multiple diversity policies, where the diversity policies are encouraged to explore response regions distinct from the main policy and from one another. The main policy performs group-relative comparison over the aggregated response set, while the diversity policies are optimized with correctness and policy-repulsion signals.

Experiments on five mathematical reasoning benchmarks with DeepSeek-R1-Distill-Qwen-1.5B and Phi-4-mini-instruct show that CoEx-GRPO achieves the best average performance among compared methods under the same rollout budget. Further analyses show that it maintains greater rollout diversity, higher reward dispersion, and higher policy entropy throughout training, suggesting that structurally expanding the comparison set effectively mitigates training-signal degradation in GRPO-based RLVR. Code is available at the anonymous repository.[1]

1 Introduction

Recent reinforcement learning (RL)-based fine-tuning has emerged as a primary approach for improving reasoning capabilities across a wide range of tasks, including mathematics, logic, and code. In particular, for tasks whose answers can be automatically verified, the Reinforcement Learning with Verifiable Rewards (RLVR) setting has rapidly gained attention, as it enables learning solely from verifiable rewards without requiring either a separate value (critic) model or a preference model (Lambert et al., 2025). Within this line of work, Group Relative Policy Optimization (GRPO) has been widely adopted as a core technique for training reasoning models due to its critic-free formulation and stable learning dynamics (Shao et al., 2024), and has also been used as a practical RL framework for large-scale reasoning model training (Guo et al., 2025; Yu et al., 2025; Mroueh, 2025).

GRPO constructs a response group by sampling multiple responses for the same prompt and uses the relative advantage induced by their scalar rewards as the learning signal. Accordingly, GRPO is most effective when candidate responses within a group remain sufficiently distinguishable, so that reward variance is preserved.

However, recent work suggests that GRPO-based RLVR can suffer from a self-reinforcing failure mode involving *homogenized exploration* and *entropy collapse* (Wu et al., 2025). As the policy becomes increasingly deterministic, sampled responses for the same prompt become more homogeneous, reducing within-group reward variation and weakening group-relative learning signals. Under sparse verification signals such as binary correctness rewards, response groups often become either all-correct or all-incorrect, making informative mixed-quality comparisons increasingly rare. This, in turn, further limits exploration and can exacerbate entropy collapse, ultimately slowing learning (Zhang and Zuo, 2025; Chen et al., 2025a; Wang et al., 2025b; Li et al., 2025a,b; Wu et al., 2025; Wang et al., 2025a). These ob-

[1] https://anonymous.4open.science/r/2026Mar_ARR_CoEx-AE97.README.md

servations suggest the need for a mechanism that maintains a sufficiently rich candidate response set throughout training.

To address this issue, we propose **CoEx-GRPO**, a multi-policy GRPO framework that jointly trains a main policy, which serves as the final model, together with multiple diversity policies. The diversity policies are encouraged to explore response regions distinct from the main policy and from one another, thereby producing more diverse candidate responses. The main policy then performs group-relative comparison over the aggregated rollouts as *collective experience*, structurally expanding the within-prompt comparison set without modifying its GRPO objective. We evaluate CoEx-GRPO on multiple mathematical reasoning benchmarks and analyze its behavior in terms of rollout diversity, reward variance, entropy dynamics, and reward dynamics.

Our contributions are twofold. First, we propose CoEx-GRPO, a multi-policy GRPO framework in which diversity policies are trained with correctness and policy-repulsion signals to provide collective experience for the main policy. Second, we show across multiple mathematical reasoning benchmarks that CoEx-GRPO improves performance while maintaining more diverse rollouts and more informative training signals.

2 Method

This work addresses the tendency of rollout responses to become less diverse as GRPO-based RLVR training progresses. As this diversity diminishes, the group-relative comparison signal (group-relative advantage) weakens, which can in turn hinder exploration and slow learning progress. To mitigate this issue, we propose **CoEx-GRPO**. CoEx-GRPO is a multi-policy GRPO framework based on collective experience, which jointly trains a main policy used for final inference together with diversity policies that are encouraged to continuously supply diverse candidate responses during training. The central goal of our method is to expand the effective candidate set over which the main policy performs relative comparison, so that GRPO can maintain sufficiently discriminative comparison signals even in the later stages of training.

Figure 1 illustrates the overall training process of CoEx-GRPO. The framework consists of (i) a main policy and (ii) K diversity policies, each implemented as an independent LoRA adapter on top of a shared base model. Each adapter is updated independently while sharing the same underlying pretrained model. At each training step, given a prompt x, the main policy samples N_m responses, and each diversity policy samples N_d responses, together forming the global candidate set $\mathcal{Y}(x)$. All responses receive a correctness reward R_c, while responses generated by the diversity policies additionally receive a diversity reward R_d, which encourages different policies to occupy distinct regions of the response distribution. This design encourages each diversity policy to continuously specialize in a different region of the response space.

The key idea of CoEx-GRPO is *asymmetric optimization*. The main policy performs relative comparison over the entire candidate batch $\mathcal{Y}(x)$ as a single group. That is, responses generated by the diversity policies are also incorporated into training as collective experience. In contrast, each diversity policy is updated within its own candidate set $\mathcal{Y}_{d,k}(x)$ by combining correctness and diversity signals. At inference time, only the main policy is used, and the diversity policies are used only during training as auxiliary modules for expanding the candidate pool during training. Below, we formalize this procedure step by step.

2.1 Multi-Policy Rollouts and the Global Candidate Set

Our framework consists of one main policy π_{θ_m} and K diversity policies $\{\pi_{\theta_{d,k}}\}_{k=1}^{K}$. All policies share the same pretrained backbone but use independent LoRA adapters. For a prompt $x \in \mathcal{D}$ and an autoregressive policy π_θ, the generation probability of a response $y = (y_1, \ldots, y_{T_y})$ is defined as

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

Main-policy rollouts. The main policy π_{θ_m} samples N_m responses to form the set

$$\mathcal{Y}_m(x) = \{y^{m,i}\}_{i=1}^{N_m}, \quad y^{m,i} \sim \pi_{\theta_m}(\cdot \mid x) \tag{2}$$

Diversity-policy rollouts. The k-th diversity policy $\pi_{\theta_{d,k}}$ samples N_d responses to form the set

$$\mathcal{Y}_{d,k}(x) = \{y^{d,k,j}\}_{j=1}^{N_d}, \quad y^{d,k,j} \sim \pi_{\theta_{d,k}}(\cdot \mid x) \tag{3}$$

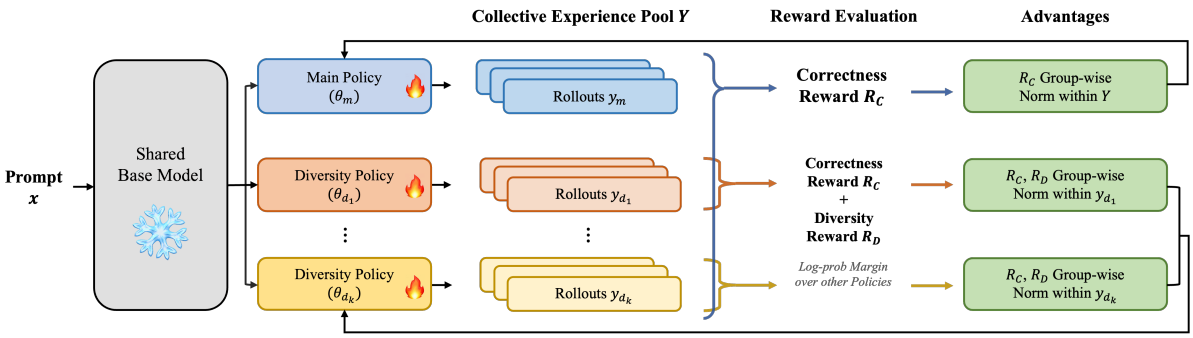

Figure 1: Architecture of CoEx-GRPO

Global candidate set. The global candidate set $\mathcal{Y}(x)$ denotes the rollout batch obtained by aggregating responses generated by the main policy and all diversity policies. Formally,

$$\mathcal{Y}(x) = \mathcal{Y}_m(x) \uplus \biguplus_{k=1}^{K} \mathcal{Y}_{d,k}(x) \tag{4}$$

where all sampled responses are retained, including duplicates. Accordingly, the size of the global candidate set is $|\mathcal{Y}(x)| = N_m + KN_d$.

2.2 Reward Function

CoEx-GRPO assigns a correctness reward to all responses and a policy-repulsion reward to responses generated by diversity policies.

Correctness reward. All responses $y \in \mathcal{Y}(x)$ in the global candidate set receive a correctness reward. For mathematical reasoning tasks, we define the correctness reward $R_c(x, y)$, which combines answer correctness with a lightweight format reward; details provided in Appendix B.1.

Diversity reward. For a response $y \in \mathcal{Y}_{d,k}(x)$ generated by a diversity policy, we additionally assign a policy repulsion reward, separate from the correctness reward. The purpose of this reward is to encourage each diversity policy to occupy a region of the response distribution that is distinct from those of the main policy and the other diversity policies.

To this end, we use the average completion-token log-probability assigned by a policy π_θ to a response y. Given a prompt x and a response y, the average completion-token log-probability under policy π_θ is defined as

$$\ell(x, y; \pi_\theta) = \frac{1}{|y|} \sum_{t=1}^{|y|} \log \pi_\theta(y_t \mid x, y_{<t}). \tag{5}$$

For a response $y \in \mathcal{Y}_{d,k}(x)$ generated by the k-th diversity policy, we define the diversity reward as the gap between the average log-probability under its generating policy and the highest average log-probability assigned to the same response by either the main policy or any of the other diversity policies:

$$R_d(x,y) = \ell(x, y; \pi_{\theta^{old}_{d,k}}) - \max\Big(\ell(x, y; \pi_{\theta^{old}_m}), \max_{\substack{i=1\ldots K \\ i \neq k}} \ell(x, y; \pi_{\theta^{old}_{d,i}})\Big). \tag{6}$$

Thus, $R_d(x, y)$ becomes larger when a response y is assigned high probability by the diversity policy that generated it, but relatively low probability by the main policy and the other diversity policies. This reward therefore encourages each diversity policy to explore regions of the response distribution that do not overlap substantially with those of the other policies. Importantly, a large $R_d(x, y)$ does not necessarily imply a different *final answer*; rather, it indicates a higher likelihood of generating different solution paths, formulations, or reasoning patterns, even when reaching the same answer.

2.3 Group-Relative Advantage Computation

Because the main policy and the diversity policies use different comparison sets, their group-relative advantages are also computed asymmetrically. Let $R(x, \mathcal{Y})$ denote the set of rewards over a response set $\mathcal{Y}$, and let $\mu(R(x, \mathcal{Y}))$ and $\sigma(R(x, \mathcal{Y}))$ denote its mean and standard deviation, respectively. For numerical stability, we use $\epsilon_{\text{std}} > 0$.

Advantage for the main policy. The main policy normalizes the correctness reward over the entire global candidate set, treating all responses $y \in \mathcal{Y}(x)$ generated by all policies as a single

group:

$$A_m(x, y) = \frac{R_c(x, y) - \mu\big(R_c(x, \mathcal{Y}(x))\big)}{\sigma\big(R_c(x, \mathcal{Y}(x))\big) + \epsilon_{\text{std}}} \quad (7)$$

By including responses generated by the diversity policies in the comparison set, the main policy forms its relative comparison signal over a broader candidate pool than would be available from its own rollouts alone.

Advantage for the diversity policies. Each diversity policy normalizes the correctness and diversity signals separately within its own response set $y \in \mathcal{Y}_{d,k}(x)$:

$$A^{\text{corr}}_{d,k}(x, y) = \frac{R_c(x, y) - \mu\big(R_c(x, \mathcal{Y}_{d,k}(x))\big)}{\sigma\big(R_c(x, \mathcal{Y}_{d,k}(x))\big) + \epsilon_{\text{std}}} \quad (8)$$

$$A^{\text{div}}_{d,k}(x, y) = \frac{R_d(x, y) - \mu\big(R_d(x, \mathcal{Y}_{d,k}(x))\big)}{\sigma\big(R_d(x, \mathcal{Y}_{d,k}(x))\big) + \epsilon_{\text{std}}} \quad (9)$$

The final advantage for the diversity policy is defined as a weighted sum of these two signals:

$$A_{d,k}(x, y) = \lambda_c\, A^{\text{corr}}_{d,k}(x, y) + \lambda_d\, A^{\text{div}}_{d,k}(x, y) \quad (10)$$

where $\lambda_c,\ \lambda_d \geq 0$ are hyperparameters that control the relative importance of the two signals.

2.4 Multi-Policy Optimization

Each policy is optimized independently using a PPO-style clipped objective, with KL regularization to improve update stability.

Optimization of the main policy. We define the token-level probability ratio and its clipped version as follows:

$$r_{m,t}(x, y) = \frac{\pi_{\theta_m}(y_t \mid x, y_{<t})}{\pi_{\theta_m^{\text{old}}}(y_t \mid x, y_{<t})}, \quad (11)$$

$$\bar{r}_{m,t}(x, y) = \text{clip}(r_{m,t}(x, y),\ 1 - \epsilon,\ 1 + \epsilon) \quad (12)$$

The loss function of the main policy is then defined as

$$\mathcal{L}_m(\theta_m) := -\mathbb{E}_{x\sim\mathcal{D}}\Bigg[\frac{1}{|\mathcal{Y}(x)|}\sum_{y\in\mathcal{Y}(x)}\frac{1}{T_y}\sum_{t=1}^{T_y} \min\Big(r_{m,t}(x, y)\, A_m(x, y),\ \bar{r}_{m,t}(x, y)\, A_m(x, y)\Big)\Bigg] + \beta\, \mathcal{L}_{\text{KL}}(\theta_m) \quad (13)$$

where $\mathcal{L}_{\text{KL}}(\theta_m)$ denotes the KL-divergence penalty with respect to a reference policy, and β is its weighting coefficient. The main policy evaluates its own log-likelihood over the entire set $\mathcal{Y}(x)$, and applies the clipped objective over an expanded candidate batch that incorporates responses from the diversity policies as collective experience. In this way, the effective range of the relative comparison signal is broadened.

Optimization of the diversity policies. The k-th diversity policy is updated only with respect to its own response set $\mathcal{Y}_{d,k}(x)$. The ratio term $r_{d,k,t}(x, y)$ and its clipped version $\bar{r}_{d,k,t}(x, y)$ are defined analogously to those of the main policy. Accordingly, the loss function for the k-th diversity policy is defined as

$$\mathcal{L}_{d,k}(\theta_{d,k}) := -\mathbb{E}_{x\sim\mathcal{D}}\Bigg[\frac{1}{|\mathcal{Y}_{d,k}(x)|}\sum_{y\in\mathcal{Y}_{d,k}(x)}\frac{1}{T_y}\sum_{t=1}^{T_y} \min\Big(r_{d,k,t}(x, y)\, A_{d,k}(x, y), \bar{r}_{d,k,t}(x, y)\, A_{d,k}(x, y)\Big)\Bigg] + \beta\, \mathcal{L}_{\text{KL}}(\theta_{d,k}) \quad (14)$$

Diversity policies jointly optimize for correctness and diversity, thereby continuously supplying candidates from different regions of the response space while still preserving the ability to generate correct answers.

Whereas the main policy is optimized over the global candidate set $\mathcal{Y}(x)$, each diversity policy is updated only on its own response set $\mathcal{Y}_{d,k}(x)$. This asymmetric structure is the core of CoEx-GRPO. It is designed to (i) encourage each diversity policy to continuously occupy a different region of the response space while minimizing overlap with others, and (ii) enable the main policy to form relative comparison signals over a broader candidate pool, thereby structurally mitigating signal weakening and the risk of mode collapse caused by rollout homogenization in the later stages of training.

3 Experiments

Datasets. Following the experimental protocol of Chen et al. (2025b); Dang and Ngo (2026), we use the OPEN-RS training set released by Dang and Ngo (2026). OPEN-RS is a small RLVR training set designed to cover a mixture of difficulty levels by combining high-difficulty mathematical reasoning problems curated from S1 with DEEPSCALER-style problems (Muennighoff

et al., 2025; Luo et al., 2025; knoveleng, 2025). We evaluate on AIME24, AIME25, MATH-500, GSM8K, and MMLU-MATH-PRO. Details of dataset construction and evaluation protocols are provided in Appendix A.

Baselines. We compare CoEx-GRPO against the following baselines: (i) **Pretrained**, which evaluates the base model without RL post-training; (ii) **GRPO** (Shao et al., 2024), the standard critic-free RLVR method based on group-relative advantages; (iii) **DAPO** (Yu et al., 2025), a GRPO variant that introduces asymmetric clipping and token-level loss normalization to mitigate entropy collapse; and (iv) **DRA-GRPO** (Chen et al., 2025b), a GRPO variant that augments the reward with a diversity signal to reduce semantic redundancy within a group. All methods use the same rollout budget and evaluation pipeline. Full baseline configurations are described in Appendix D.

Experimental Setup. We use DEEPSEEK-R1-DISTILL-QWEN-1.5B (Guo et al., 2025) and PHI-4-MINI-INSTRUCT (Microsoft et al., 2025) as base models. For DRA-GRPO†, we use the checkpoints released by the authors and re-evaluate them under our evaluation pipeline[2]; all other methods are trained and evaluated within the same implementation. For all methods, we fix the total number of rollouts per prompt to $|\mathcal{Y}(x)| = 10$. For CoEx-GRPO, we set $N_m = 4$, $K = 3$, $N_d = 2$, and $(\lambda_c, \lambda_d) = (0.7, 0.3)$. All models are trained for 1,000 steps. To reduce sensitivity to late-stage training instability, we report the score of the best checkpoint selected from four evaluation checkpoints; full details are provided in Appendix C. Since RL post-training is intended to improve reasoning on tasks for which the base model already exhibits non-trivial competence, we exclude benchmarks on which the pretrained model achieves below 10% accuracy. For such cases, we report the corresponding entries as "–" and exclude them from the average. All plots are smoothed using an exponential moving average (EMA) with a fixed smoothing factor of 0.03.

3.1 Main Results

For **DeepSeek-R1-Distill-Qwen-1.5B**, CoEx-GRPO achieves an average score of 41.38%, corresponding to gains of **+4.40** percentage points over the pretrained model and **+3.44** percentage points over GRPO. It attains the best performance among all compared methods on AIME25 with 26.67%, and also achieves the strongest results on MATH-500 (72.60%) and GSM8K (74.45%). Although CoEx-GRPO does not outperform DRA-Dr.GRPO on AIME24, where the latter reaches 23.33%, it yields the best overall average performance across benchmarks.

At the task level, CoEx-GRPO is particularly effective on MATH-500, which requires more complex reasoning. With **Phi-4-mini-instruct**, CoEx-GRPO also remains competitive, achieving 45.20% on MATH-500 and 83.24% on GSM8K, and obtains the best average score of 64.16%. These results suggest that, on challenging reasoning tasks, expanding the candidate set through multiple diversity policies helps capture a broader range of valid solution paths.

Overall, under the same rollout budget, CoEx-GRPO expands the range of candidates observed by the main policy through the collective experience provided by the diversity policies. This helps preserve informative group-relative comparison signals for the main policy even in the later stages of training, where such signals typically weaken, and results in competitive performance across mathematical reasoning tasks of varying difficulty.

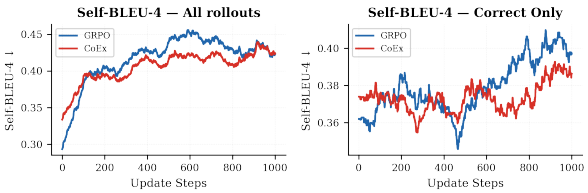

Figure 2: Self-BLEU-4 of the global candidate set $\mathcal{Y}(x)$ over training steps. Lower values indicate less lexical overlap among candidate responses. Left: all rollouts; Right: correct rollouts only.

4 Analysis

4.1 Diversity Analysis

All figures and tables in this section report training dynamics and results measured on DEEPSEEK-R1-DISTILL-QWEN-1.5B.

Diversity. Figure 2 reports the lexical diversity of the global candidate set $\mathcal{Y}(x)$ over training, measured by Self-BLEU-4. Lower values indicate less lexical overlap among candidate responses.

When computed over all rollouts (left), CoEx-GRPO consistently maintains lower Self-BLEU

[2] https://github.com/xiwenc1/DRA-GRPO

Model	Method	AIME24	AIME25	MATH-500	GSM8K	MMLU-Pro	Avg.
DeepSeek-R1-Distill-Qwen-1.5B	Pretrained	10.00	16.67	70.80	71.57	15.84	36.98
	GRPO	13.33	20.00	69.40	73.92	13.03	37.94
	DAPO	3.33	0.00	64.00	71.42	13.47	30.44
	DRA-GRPO†	10.00	10.00	70.40	71.49	**19.99**	36.38
	DRA-Dr.GRPO†	**23.33**	16.67	71.80	71.87	19.69	40.67
	CoEx (Ours)	13.33	**26.67**	**72.60**	**74.45**	19.84	**41.38**
Phi-4-mini-instruct	Pretrained	0.00	0.00	44.00	82.34	63.36	63.23
	GRPO	–	–	44.40	82.26	**64.17**	63.61
	DAPO	–	–	43.80	82.94	63.14	63.29
	CoEx (Ours)	–	–	**45.20**	**83.24**	64.03	**64.16**

Table 1: Main results on five mathematical reasoning benchmarks. Here, "MMLU-Pro" denotes MMLU-MATH-Pro. Scores are reported as percentages (%). Methods marked with † use checkpoints released by Chen et al. (2025b) and are re-evaluated under our evaluation pipeline.

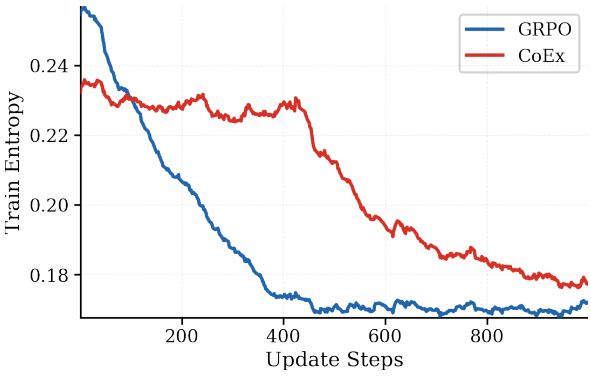

Figure 3: Training entropy of the main policy over training steps.

than GRPO from the early stage of training (after roughly 100 steps), indicating that lexical diversity at the level of the global candidate set is better preserved throughout training. This pattern suggests that the diversity policies continuously contribute responses that remain distinct from those produced by the main policy.

A similar trend is observed when the metric is computed only on correct rollouts (right), where CoEx-GRPO again shows lower Self-BLEU than GRPO. This suggests that diversity is preserved not only among all sampled responses, but also among reasoning paths that arrive at correct answers. These results suggest that CoEx-GRPO preserves diversity not only across all sampled outputs but also within the subset of correct solutions.

Main-Policy Entropy. Figure 3 shows the token-level entropy of the main policy throughout training. Under standard GRPO, entropy decreases steadily as training progresses, whereas the main policy in CoEx-GRPO maintains a relatively higher entropy level even in the later stages of training.

In standard GRPO, rollouts are sampled only from the main policy itself. As training proceeds, diversity among candidate responses tends to shrink, weakening the group-relative comparison signal. As a result, the policy can converge prematurely to a small set of response patterns that already yield relatively high reward, leading to decreasing entropy.

By contrast, in CoEx-GRPO, the diversity policies continuously contribute candidates from distinct regions of the response space, which helps preserve diversity in the global candidate set $\mathcal{Y}(x)$ on which the main policy performs relative comparison. This structurally expanded candidate set provides the main policy with a more diverse stream of learning signals throughout training. Accordingly, the main policy is less prone to early convergence toward a narrow set of response patterns and maintains a higher entropy level over time.

These results suggest that CoEx-GRPO can preserve exploratory capacity through structural expansion of the candidate set, without explicitly regularizing the entropy of the main policy.

Policy Differentiation. Figure 4 shows the centroid cosine distance between the main policy and the set of diversity policies during training. For each policy, we compute the centroid of response embeddings and then measure the cosine distance between the centroid of the main policy and that of the diversity-policy set.

The figure exhibits a gradual upward trend, indicating that the distance between the two centroids increases as training progresses. This sug-

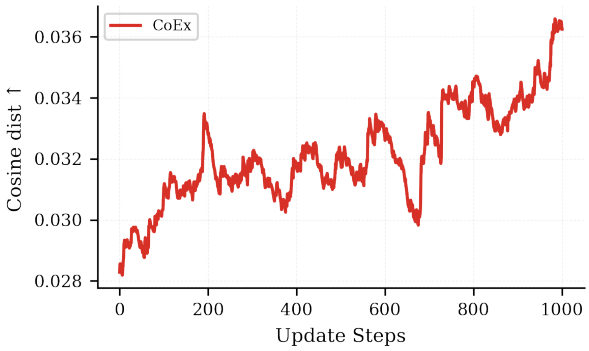

Figure 4: Centroid cosine distance between the main adapter and diversity adapters over training steps.

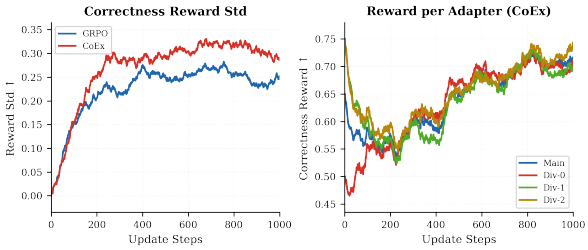

Figure 5: Reward dynamics over training steps. Left: within-group reward standard deviation of GRPO vs. CoEx-GRPO. Right: per-adapter correctness reward trajectories within CoEx-GRPO.

gests that, although the main and diversity policies begin from similar distributions, they gradually differentiate into semantically distinct regions of the response space over the course of training. This observation provides supporting evidence that the diversity policies progressively occupy increasingly distinct regions of the response space from those of the main policy.

4.2 Learning Dynamics

Figure 5 illustrates reward dynamics over the course of training.

Correctness Reward Dispersion. The left panel compares the standard deviation of within-group correctness rewards for GRPO and CoEx-GRPO. Both methods show a rapid increase in reward dispersion during the early stage of training followed by stabilization. However, CoEx-GRPO consistently maintains higher reward dispersion than GRPO throughout training (approximately 0.28–0.30). Because reward dispersion directly reflects the informativeness of the group-relative comparison signal, this result suggests that the expanded global candidate set in CoEx-GRPO preserves informative learning signals for the main policy even in later training stages.

Setting	A25	M500	GSM8K	MMLU-Pro	Avg.
K=1, N_d=2	26.67	71.00	70.51	7.55	43.93
K=2, N_d=2	0.00	55.80	67.02	4.74	31.89
K=3, N_d=2	**26.67**	**72.60**	**74.45**	**19.84**	**48.39**

Table 2: Effect of diversity-policy count K with N_d=2 fixed. A25: AIME25, M500: MATH-500. Scores are reported as percentages (%).

Benchmark	Pretrained	GRPO	DRA-GRPO	CoEx
A24 (p@16)	6.67	10.00	13.33	**23.33**
A25 (p@16)	10.00	20.00	20.00	**26.67**

Table 3: Sampling-based evaluation on AIME24/AIME25, under pass@16 with temperature T=0.6. Scores are reported as percentages (%).

Per-Policy Correctness Reward Trajectories. The right panel shows the correctness reward trajectories of the main policy (Main) and the diversity policies (Div-0, Div-1, Div-2) in CoEx-GRPO. Although Div-0 exhibits temporarily lower reward in the early stage of training, all policies show an upward trend after roughly 200 steps. In the later stage (after approximately 800 steps), the main policy and all diversity policies converge to similar reward levels while continuing to improve. This suggests that the diversity policies are not merely serving as exploratory modules, but are also learning to maintain correctness, thereby supplying useful candidates that can support improvements in the main policy.

4.3 Hyperparameter Sensitivity: Number of Diversity Policies

As K increases from 1 to 2, the average performance drops from 43.93% to 31.89%. This result indicates that simply increasing the number of diversity policies does not necessarily improve performance, and that training may become unstable for certain values of K. With K=3, however, the model achieves the best performance on all comparable benchmarks, reaching an average of 48.39%, which also improves over the K=1 setting.

4.4 Sampling-Based Evaluation

Table 3 reports pass@16 performance, which evaluates model capability under a sufficiently large sampling budget. CoEx-GRPO achieves the

strongest results among all compared methods, reaching 23.33% on AIME24 and 26.67% on AIME25. This contrasts with the single-response greedy-decoding evaluation in Table 1, where CoEx does not outperform DRA-Dr.GRPO on AIME24.

Prior work has reported that RLVR training can improve pass@1 while simultaneously narrowing the output distribution through entropy collapse, which may degrade performance under large-sample evaluation settings (Yue et al., 2025). In our results, GRPO shows only limited gains under sampling-based evaluation relative to greedy decoding, whereas CoEx-GRPO achieves the strongest problem-solving performance under pass@16. This finding suggests that CoEx-GRPO mitigates entropy collapse and internalizes a broader set of reasoning paths, consistent with the higher entropy maintained in Figure 3.

4.5 Computational Overhead

CoEx-GRPO introduces additional computational cost during training because diversity reward computation requires extra forward passes across multiple policies, and because auxiliary diversity policies must be optimized jointly with the main policy. More specifically, this overhead arises in the reward computation stage and the parameter update stage. By contrast, the rollout stage does not require additional response generation, since it reuses responses already produced by the main and diversity policies.

Prior work has shown that, in RL post-training, the rollout stage often accounts for a substantial fraction of total wall-clock time (approximately 66%–90%). In light of this observation, increased computation in the reward and training stages does not necessarily translate proportionally into end-to-end training time. Under the main experimental setting of this paper, the theoretical overhead is estimated to be approximately 30%, although additional implementation-level overhead may arise depending on the experimental environment. A more detailed analysis of the theoretical computation cost and implementation considerations is provided in Appendix E.

5 Conclusion

In this paper, we proposed **CoEx-GRPO**, a multi-policy framework that mitigates the weakening of group-relative comparison signals in GRPO-based RLVR training, where rollout responses tend to become increasingly homogeneous as training progresses. CoEx-GRPO jointly trains a main policy together with multiple diversity policies, and structurally expands the global candidate set by integrating the rollouts of the diversity policies as collective experience. In doing so, it preserves informative group-relative comparison signals in the later stages of training without modifying the GRPO objective of the main policy.

Experimental results on five mathematical reasoning benchmarks show that CoEx-GRPO achieves competitive or best average performance across both DEEPSEEK-R1-DISTILL-QWEN-1.5B and PHI-4-MINI-INSTRUCT. Our analysis further shows that CoEx-GRPO maintains lower Self-BLEU, higher reward dispersion, and higher token-level entropy of the main policy throughout training compared with GRPO. We also observe a gradual increase in semantic distance between the main policy and the diversity policies over the course of training, suggesting that the diversity policies progressively specialize in response regions that are distinct from those of the main policy. Under pass@16 evaluation, CoEx-GRPO also outperforms the compared baselines on both AIME24 and AIME25.

Taken together, these findings suggest that, in contrast to prior GRPO variants that primarily rely on algorithmic modifications within a single-policy framework, structurally expanding the candidate response set through a multi-policy design is a promising direction for mitigating training-signal degradation in RLVR.

Limitations

Training Efficiency and System Overhead. One practical limitation of CoEx-GRPO is its higher training cost relative to standard GRPO. Our method requires additional forward passes to compute the diversity reward and jointly optimizes multiple policy adapters, which increases both computational cost and system complexity during training. Although the rollout stage itself does not require additional response generation, the overall overhead is still estimated to be approximately 30%, as discussed in the main text. However, the actual wall-clock overhead can vary substantially depending on hardware configuration, batching strategy, implementation efficiency, and framework-level optimization. Accordingly,

in this work we primarily report theoretically estimated overhead rather than a direct quantitative comparison of wall-clock training time, and treat the latter as an informal observation.

Single-Domain Evaluation. Our experiments focus exclusively on mathematical reasoning tasks. Accordingly, the effectiveness of CoEx-GRPO in other RLVR domains, such as code generation or logical reasoning, remains unverified. Evaluating the generality of the proposed framework across a broader range of domains is an important direction for future work.

References

Peter Chen, Xiaopeng Li, Ziniu Li, Xi Chen, and Tianyi Lin. 2025a. Stepwise guided policy optimization: Coloring your incorrect reasoning in grpo. *Preprint*, arXiv:2505.11595.

Xiwen Chen, Wenhui Zhu, Peijie Qiu, Xuanzhao Dong, Hao Wang, Haiyu Wu, Huayu Li, Aristeidis Sotiras, Yalin Wang, and Abolfazl Razi. 2025b. Dra-grpo: Diversifying reasoning ability via hybrid model- and rule-based reward. *Preprint*, arXiv:2505.09655. GitHub: https://github.com/xiwenc1/DRA-GRPO.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Quy-Anh Dang and Chris Ngo. 2026. Reinforcement learning for reasoning in small LLMs: What works and what doesn't. In *Logical and Symbolic Reasoning in Language Models @ AAAI 2026*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2024. The language model evaluation harness.

Wei Gao, Yuheng Zhao, Dakai An, Tianyuan Wu, Lunxi Cao, Shaopan Xiong, Ju Huang, Weixun Wang, Siran Yang, Wenbo Su, Jiamang Wang, Lin Qu, Bo Zheng, and Wei Wang. 2025. Rollpacker: Mitigating long-tail rollouts for fast, synchronous rl post-training. *Preprint*, arXiv:2509.21009.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *Advances in neural information processing systems*.

Maxwell Jia. 2024. Aime problem set 2024.

knoveleng. 2025. knoveleng/open-rs: Open-rs training data and evaluation scripts. GitHub repository.

Hynek Kydlicek, Alina Lozovskaya, Nathan Habib, and Clémentine Fourrier. 2025. Fixing open llm leaderboard with math-verify.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, and 4 others. 2025. Tulu 3: Pushing frontiers in open language model post-training. *Preprint*, arXiv:2411.15124.

Gang Li, Ming Lin, Tomer Galanti, Zhengzhong Tu, and Tianbao Yang. 2025a. Disco: Reinforcing large reasoning models with discriminative constrained optimization. *Preprint*, arXiv:2505.12366.

Qingbin Li, Rongkun Xue, Jie Wang, Ming Zhou, Zhi Li, Xiaofeng Ji, Yongqi Wang, Miao Liu, Zheming Yang, Minghui Qiu, and Jing Yang. 2025b. Cure: Critical-token-guided re-concatenation for entropy-collapse prevention. *Preprint*, arXiv:2508.11016.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025. Understanding R1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Erran Li Li, Raluca Ada Popa, and Ion Stoica. 2025. Deepscaler-1.5b-preview: Surpassing o1-preview with a 1.5b model by scaling rl. Blog / Project page. Repo: https://github.com/agentica-project/deepscaler (redirects to https://github.com/rllm-org/rllm).

math ai. 2025. Aime problem set 2025.

Microsoft, :, Abdelrahman Abouelenin, Atabak Ashfaq, Adam Atkinson, Hany Awadalla, Nguyen Bach, Jianmin Bao, Alon Benhaim, Martin Cai, Vishrav Chaudhary, Congcong Chen, Dong Chen, Dongdong Chen, Junkun Chen, Weizhu Chen, Yen-Chun Chen, Yi ling Chen, Qi Dai, and 57 others. 2025. Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras. *Preprint*, arXiv:2503.01743.

Youssef Mroueh. 2025. Reinforcement learning with verifiable rewards: Grpo's effective loss, dynamics, and success amplification. *Preprint*, arXiv:2503.06639.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling. *Preprint*, arXiv:2501.19393. GitHub: https://github.com/simplescaling/s1.

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

Chen Wang, Lai Wei, Yanzhi Zhang, Chenyang Shao, Zedong Dan, Weiran Huang, Yuzhi Zhang, and Yue Wang. 2025a. Eframe: Deeper reasoning via exploration-filter-replay reinforcement learning framework. *Preprint*, arXiv:2506.22200.

Haozhe Wang, Chao Qu, Zuming Huang, Wei Chu, Fangzhen Lin, and Wenhu Chen. 2025b. Vl-rethinker: Rethinking the role of reinforcement learning in visual language models. *Preprint*, arXiv:2504.08837.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Preprint*, arXiv:2406.01574.

Junkang Wu, Kexin Huang, Jiancan Wu, An Zhang, Xiang Wang, and Xiangnan He. 2025. Quantile advantage estimation for entropy-safe reasoning. *Preprint*, arXiv:2509.22611. OpenReview forum (optional): https://openreview.net/forum?id=WDP5b3mtFV.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, and 16 others. 2025. An open-source llm reinforcement learning system at scale. *Preprint*, arXiv:2503.14476. Project page: https://dapo-sia.github.io/.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. 2025. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In *Advances in Neural Information Processing Systems*. Oral.

Jixiao Zhang and Chunsheng Zuo. 2025. Grpo-lead: A difficulty-aware reinforcement learning approach for concise mathematical reasoning in language models. *Preprint*, arXiv:2504.09696.

Haizhong Zheng, Yang Zhou, Brian R. Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. 2025. Act only when it pays: Efficient reinforcement learning for LLM reasoning via selective rollouts. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

A Datasets and Preprocessing

Training set: OPEN-RS. Following the setup of (Dang and Ngo, 2026; Chen et al., 2025b), we use the OPEN-RS training data released by Dang and Ngo (2026). OPEN-RS consists of 7,000 mathematical reasoning problems in total, and is a small RLVR training set constructed to mix difficulty levels by combining 3,000 high-difficulty problems curated from OPEN-S1 with 4,000 problems curated from OPEN-DEEPSCALER (1,000 easy + 3,000 hard) (Dang and Ngo, 2026; Muennighoff et al., 2025; Luo et al., 2025; knoveleng, 2025). Each sample includes `problem` (question) and `solution` (explanation including the answer) fields, and the dataset and evaluation tools are provided in the `knoveleng/open-rs` repository[3]. For RLVR training, we use `problem` as the prompt, and for correctness rewards we extract the answer from `solution` and use it as the gold answer.

Dataset Statistics. Table 4 summarizes the key statistics of the OPEN-RS training set and evaluation benchmarks used in this work.

Curation and filtering. OPEN-RS is curated by extracting mathematical problems from source datasets and refining their quality. Dang and Ngo (2026) first (i) prioritizes math-style problems by retaining only solutions that include LaTeX `\boxed{}` in S1, (ii) removes trivial problems using a small reasoning model, and then (iii) removes noisy and multi-part problems using a larger model to construct OPEN-S1. For the DEEPSCALER family, additional filtering is applied to remove easy problems, yielding OPEN-DEEPSCALER (Dang and Ngo, 2026). Finally, OPEN-RS is formed by sampling from OPEN-S1 and OPEN-DEEPSCALER to construct a 7,000-problem set (knoveleng, 2025).

Evaluation benchmarks. For AIME24/AIME25 and MATH-500, we use the task configurations provided by `lm-evaluation-harness` (Gao et al., 2024), which implement symbolic equivalence checking via the `math_verify` library. For GSM8K, we use the standard `lm-evaluation-harness` task

[3] `https://github.com/knoveleng/open-rs`

Split / Benchmark	# Problems	Avg. Prompt (tok)	Avg. Solution (tok)	Source
OPEN-S1 subset	3,000	63	429	Muennighoff et al. (2025)
OPEN-DEEPSCALER (easy)	1,000	50	59	Luo et al. (2025)
OPEN-DEEPSCALER (hard)	3,000	63	429	Luo et al. (2025)
OPEN-RS (total)	7,000	61	376	Dang and Ngo (2026)
AIME24	30	103	–	Jia (2024)
AIME25	30	175	–	math ai (2025)
MATH-500	500	72	–	Hendrycks et al. (2021)
GSM8K	1,319	62	–	Cobbe et al. (2021)
MMLU-MATH-PRO	1,351	51	–	Wang et al. (2024)

Table 4: Statistics of training data and evaluation benchmarks. Token lengths are measured using the DEEPSEEK-R1-DISTILL-QWEN-1.5B tokenizer. OPEN-S1 subset and OPEN-DEEPSCALER (hard) are both Hard-level problems and share identical token statistics (avg. prompt: 63, avg. solution: 429). "–": not applicable (evaluation benchmarks).

with flexible numeric matching. For MMLU-MATH-PRO, we use the multiple-choice subset corresponding to mathematical reasoning problems (Wang et al., 2024). All benchmarks are evaluated under a temperature of 0.0 (greedy decoding) unless otherwise stated. The same system prompt (Appendix C.2) is applied to all methods and all benchmarks to ensure a consistent evaluation condition.

Evaluation protocol and answer extraction. We follow the evaluation pipeline provided by `lm-evaluation-harness`, which performs answer extraction and flexible answer matching rather than strict string exact match. Specifically, the evaluator (i) extracts the final answer from model outputs by locating the last `\boxed{...}` expression, or falling back to the last line or math block if `\boxed` is absent; (ii) normalizes the extracted answer string by removing extraneous whitespace and punctuation and by canonicalizing equivalent mathematical representations (e.g., fractions, rationals); and (iii) applies symbolic equivalence checking via the `math_verify` library, which compares parsed LaTeX expressions beyond surface-level string matching (Kydlicek et al., 2025). Integer-answer benchmarks (AIME24, AIME25) additionally normalize parsed answers to integer strings prior to comparison.

Handling low-accuracy scores. For low-sample benchmarks such as AIME24 and AIME25 (30 problems each), scores at below 10% (i.e., $\leq$3 correct problems) are susceptible to high variance and stochastic noise, making them unreliable as a basis for meaningful comparison. Accordingly, we report such scores as "–" in all result tables rather than treating them as informative performance estimates.

B Reward Function in Mathematical Reasoning

In the mathematical reasoning RLVR setting, we construct training signals by combining (i) a correctness reward that verifies whether the final answer is correct and (ii) a format reward that stabilizes the output format. In addition, for diversity policies, we further use (iii) a diversity reward that reduces overlap with responses from other policies. (Note that R_c in the main text corresponds to the combined reward R_{c} defined in this section.)

B.1 Correctness Reward

Given a prompt x and a model response y, we first define an answer indicator $R_{\mathrm{ans}}(x, y)$ that checks whether the final answer extracted from the response matches the reference answer:

$$R_{\mathrm{ans}}(x, y) \;=\; \mathbb{I}[\mathrm{Verify}(\mathrm{Parse}(y),\, \mathrm{Parse}(y^{\star}))]\,, \tag{15}$$

where $y^{\star}$ is the answer string provided by the dataset, $\mathrm{Parse}(\cdot)$ is a LaTeX-based answer extraction function, and $\mathrm{Verify}(\cdot,\cdot)$ is a verification function that checks equivalence between normalized answers.

Answer extraction and normalization. Our parser applies normalization to be robust to notational differences such as equations, units, and parentheses. For the reference answer string $y^{\star}$, we extract a single answer using a first-match strategy. For the model output y, we prioritize extracting the answer by detecting the `\boxed{...}` pattern. If `\boxed` appears multiple times, we use the content of the last `\boxed` as the final answer.

If \boxed is absent, we extract an answer candidate from the last line (or the last math block) of the output. Normalization includes basic LaTeX cleanup, handling of formulas/equations, unit processing, and \boxed processing, with the goal of treating notationally different but equivalent answers as identical.

B.2 Format Reward

Using correctness alone can lead to unstable output formats. We therefore additionally reward whether the output follows a specified template. Our system prompt requires the use of <think> and <answer> tags, and the format reward $R_{\text{fmt}}(x, y)$ checks minimal compliance:

$$R_{\text{fmt}}(x, y) = \alpha \cdot \mathbb{I}\big[\texttt{</think>}\text{ appears exactly once}\big], \tag{16}$$

where α is the weight of the format reward, and we use $\alpha = 0.5$ in our implementation. This design is a minimal constraint intended to simplify format parsing and improve the stability of the training/evaluation pipeline, and aims to induce more consistent final-answer extraction.

B.3 Combined Reward for RLVR

We define the final RLVR reward as the sum of the correctness and format signals:

$$R_{\text{c}}(x, y) = R_{\text{ans}}(x, y) + R_{\text{fmt}}(x, y). \tag{17}$$

Thus, $R_{\text{c}}(x, y) \in [0, 1.5]$, and this value is used as the input to the group-relative advantage computation in the main text.

B.4 Diversity Reward for Diversity Policies (Policy Repulsion)

Diversity policies are trained to generate candidates that occupy distinct regions of the output space relative to the main policy and other diversity policies. Rather than relying on surface-level n-gram overlap metrics, we adopt a *policy repulsion* reward that directly measures distributional separation in the model's own probability space.

Let $\pi_\theta^{(k)}$ denote the k-th diversity policy adapter and let $\mathcal{A} \setminus \{k\}$ be the set of all other adapters (including the main policy $\pi_\theta^{(0)}$). For a prompt x and a candidate completion y generated by $\pi_\theta^{(k)}$, we define the mean sequence log-probability under adapter a as:

$$\bar{\ell}_a(y \mid x) = \frac{1}{|y|} \sum_{t=1}^{|y|} \log \pi_\theta^{(a)}(y_t \mid y_{<t}, x). \tag{18}$$

The diversity reward for adapter k is then defined as:

$$R_{\text{div}}^{(k)}(x, y) = \bar{\ell}_k(y \mid x) - \max_{a \in \mathcal{A} \setminus \{k\}} \bar{\ell}_a(y \mid x). \tag{19}$$

Intuitively, $R_{\text{div}}^{(k)}$ measures the *log-probability margin* between the generating adapter and its strongest competitor. A positive value indicates that the completion y is more probable under $\pi_\theta^{(k)}$ than under any other adapter — i.e., it occupies a region that is *distinctive* to adapter k. Maximizing this reward encourages diversity policies to generate candidates that cannot be well-explained by the main policy or sibling adapters, thereby promoting complementary exploration of the solution space.

Relation to output diversity. Unlike lexical-based diversity rewards(e.g., BLEU-n, distinct-n ...) that operate on token surface forms, diversity is grounded in the model's learned probability distribution. Two completions that are lexically similar but arise from different derivation paths can still receive high repulsion reward if the generating adapter assigns them substantially higher probability than competing adapters do. This is particularly suited to mathematical reasoning, where multiple valid solution paths may share surface-level tokens (e.g., "=") while representing genuinely distinct reasoning strategies.

C Experiment Details

C.1 Checkpoint Selection and Evaluation Schedule

To mitigate the effect of training instability near convergence, we evaluate checkpoints at steps 700, 800, 900, and 1,000, and report the best performance among these four checkpoints as the final result.

Training. We train CoEx-GRPO by loading the base model with 4-bit (NF4) quantization and updating only the LoRA adapter parameters. We fix N_m=4, N_d=2, K=3, yielding $|\mathcal{Y}(x)|$=10. To reduce rollout generation cost, we use vLLM's colocate mode. Key hyperparameters are summarized in Appendix C.

Hardware. All training and evaluation are conducted on a single RTX 3090 GPU.

Category / Parameter		Value
Infrastructure	vLLM mode	colocate
	Tensor parallel size	1
	Temperature	0.7
Training	Base model quantization	4-bit (NF4)
	LoRA $(r, \alpha, \text{dropout})$	$(16,\ 32,\ 0.05)$
	Loss type	GRPO
	Learning rate	5×10^{-6}
	LR scheduler	Cosine (min_lr_rate$=0.1$)
	Global / mini batch	10 / 1
	Gradient accumulation	2
	Max steps	1,000
	ϵ, β	$0.2,\ 0.04$
Generation	Max length (prompt / completion)	512 / 3,584
	Total rollouts per prompt	10
Multi-Policy	$(K,\ N_m,\ N_d)$	$(3,\ 4,\ 2)$
	$(\lambda_c,\ \lambda_d)$	$(0.7,\ 0.3)$
Rewards	Correctness reward weight	1.0
	Format reward weight (α)	0.5
	Diversity reward weight	1.0
Hardware	GPU	RTX 3090 $\times$ 1

Table 5: Full hyperparameter configuration for CoEx-GRPO (DEEPSEEK-R1-DISTILL-QWEN-1.5B).

Evaluation protocol. We directly use the test sets and evaluator configuration provided by `lm-evaluation-harness` (Gao et al., 2024). Specifically, for each benchmark (AIME24/AIME25, MATH-500, GSM8K, MMLU-MATH-PRO), we measure accuracy under the same prompt template and the same answer extraction/normalization rules. **DRA-GRPO** uses released checkpoints and is re-evaluated under the same pipeline; all other methods are trained and evaluated within the same implementation. All comparisons share the same benchmark instances, parsing/verification procedure, and evaluation script.

C.2 Prompt used for Training

All methods use the same system prompt. To ensure consistent answer parsing, we require the final answer to be printed within `\boxed{}`, and instruct that the reasoning process and the final answer be wrapped with `<think>` and `<answer>` tags, respectively. The following system prompt is used for all methods.

```
SYS_OPENRS_PROMPT = (
    "A conversation between User and
    Assistant. The user asks a question,
     and the Assistant solves it. "
    "The assistant first thinks about
    the reasoning process in the mind
    and then provides the user with the
    answer, "
    "and put your final answer within \\
    boxed{}. "
    "The reasoning process and answer
    are enclosed within <think> </think>
     and <answer> </answer> tags,
    respectively, "
    "i.e., <think> reasoning process
    here </think> <answer> answer here
    </answer>. "
    "Note that respond by English, NOT
    use other languages."
)
```

D Detailed Baselines

This section summarizes the key configurations and the training/evaluation protocols of the baselines compared in the main text. All methods use the same prompt format as Appendix C.2, the same answer parsing/normalization and verification rules, and the same evaluation script.

Pre-trained. We evaluate the base model as-is, without additional training.

GRPO. Standard GRPO samples $|\mathcal{Y}(x)|=10$ responses per prompt and updates the policy using group-relative advantages computed from the verification-based reward. (See Section 2 in the main text for detailed equations and the training procedure.)

DAPO. DAPO modifies GRPO's training objective to address entropy collapse and training in-

Category / Parameter		Value
Infrastructure	vLLM mode	colocate
	Tensor parallel size	1
	vLLM GPU memory utilization	0.6
	Temperature	0.7
Training	Base model quantization	4-bit (NF4)
	LoRA $(r, \alpha, \text{dropout})$	$(16,\ 32,\ 0.05)$
	Loss type	GRPO
	Learning rate	5×10^{-6}
	LR scheduler	Cosine (min_lr_rate$=0.1$)
	Global / mini batch	10 / 1
	Gradient accumulation	2
	Max steps	1,000
	ϵ, β	$0.2,\ 0.04$
Generation	Max length (prompt / completion)	512 / 3,584
	Total rollouts per prompt	10
Multi-Policy	$(K,\ N_m,\ N_d)$	$(3,\ 4,\ 2)$
	$(\lambda_c,\ \lambda_d)$	$(0.7,\ 0.3)$
Rewards	Correctness reward weight	1.0
	Format reward weight (α)	0.5
	Diversity reward weight	1.0
Hardware	GPU	RTX 3090 $\times$ 1

Table 6: Full hyperparameter configuration for CoEx-GRPO (PHI-4-MINI-INSTRUCT).

stability by removing the KL penalty, applying asymmetric clipping to sustain exploration, and adopting token-level loss normalization. In our study, we train it under the same rollout budget of $|\mathcal{Y}(x)|$=10 responses per prompt, following the released implementation and configuration (Yu et al., 2025).

DRA-GRPO. DRA-GRPO incorporates an additional signal into the reward to reduce semantic redundancy within a group, with the goal of alleviating excessive concentration of the rollout distribution. DRA-Dr.GRPO applies the same DRA reward adjustment but substitutes the base optimization loss from GRPO to Dr.GRPO (Liu et al., 2025). In our study, we use the checkpoints released by the authors and re-evaluate under the same evaluator and parsing/normalization rules (Chen et al., 2025b).

E Detailed Computational Overhead

CoEx-GRPO requires additional computation to evaluate the diversity reward, since it must compute the mean log-probability for responses under multiple policies. This introduces extra forward passes during the reward computation stage. Specifically, for the KN_d responses generated by the diversity policies, computing the diversity reward requires one additional forward pass under the main policy and $K-1$ additional forward passes under the other diversity policies, excluding the generating policy itself. Therefore, each such response requires a total of K additional forward passes, and the total forward-pass cost for reward computation increases by a factor of

$$\frac{N_m + KN_d + K^2 N_d}{N_m + KN_d}$$

relative to the standard setting.

The total number of responses used during training also increases. The main policy is trained on $N_m + KN_d$ responses, while the diversity policies are additionally trained on a total of KN_d responses. As a result, the total number of training responses becomes $(N_m + KN_d) + KN_d$, and the training-time computation therefore increases by a factor of

$$\frac{N_m + 2KN_d}{N_m + KN_d}$$

compared with the standard setting. By contrast, the rollout stage incurs no additional overhead, since no extra response generation is required.

According to Gao et al. (2025), a typical RL post-training pipeline can be divided into three stages: rollout, reward, and training. Prior work further reports that the rollout stage accounts for the majority of total wall-clock time in RL post-training (Gao et al., 2025; Zheng et al., 2025),

typically occupying approximately 66%–90% of the total time. In contrast, the reward stage accounts for a relatively smaller fraction, approximately 5%–13%, depending on the experimental setup (Gao et al., 2025). Based on these observations, we conservatively assume a rollout/reward/-training time ratio of 7:1:2 in order to estimate the theoretical overhead of CoEx-GRPO.

Under the main experimental setting used in this paper, with $N_m = 4$, $N_d = 2$, and $K = 3$, the rollout stage incurs no additional overhead, while the computation cost increases by $2.8\times$ in the reward stage and by $1.6\times$ in the training stage. Accordingly, the overall theoretical overhead on end-to-end training time can be estimated as

$$\frac{7 \times 1 + 1 \times 2.8 + 2 \times 1.6}{7 + 1 + 2} = 1.3,$$

which implies an approximately $1.3\times$ increase in total wall-clock time in theory.

That said, CoEx-GRPO jointly operates multiple adapters corresponding to different policies, which introduces additional implementation-level overhead such as adapter switching. The magnitude of such overhead depends heavily on factors including batch size, implementation strategy, code optimization, and framework-level efficiency. For this reason, we do not present a direct quantitative comparison of wall-clock time in this work, and instead report it only as an informal observation. In our own experimental environment and implementation, the total wall-clock time increased by approximately $1.6\times$ to $3.1\times$ (roughly $46\,\mathrm{h}$ to $87\,\mathrm{h}$), depending on the experimental setting.

F Related Work

F.1 RLVR-Based Reasoning Training and GRPO

For reasoning tasks whose answers can be automatically verified (e.g., mathematics and code), the RLVR (Reinforcement Learning with Verifiable Rewards) setting has been widely adopted, as it enables policy optimization solely from verifiable outcome rewards without requiring either preference data or a separate value (critic) model (Lambert et al., 2025). In this context, GRPO forms a group of responses sampled from the same prompt and constructs update signals through within-group relative comparison (group-relative advantage), thereby enabling critic-free optimization (Shao et al., 2024). GRPO has also been adopted as a practical RL framework for large-scale reasoning model training (Guo et al., 2025; Yu et al., 2025), and recent studies have further analyzed the GRPO objective and its training dynamics in order to better understand the factors underlying its empirical success (Mroueh, 2025). In this work, we focus on how to mitigate signal weakening and reduced exploration during training while preserving the efficiency afforded by GRPO's relative-comparison structure.

F.2 Signal Weakening in GRPO and Rollout Diversity

GRPO updates rely on relative comparisons among candidate responses under the same prompt, and are most effective when within-group reward variance remains sufficiently discriminative. However, as training progresses, rollouts tend to become increasingly homogeneous. Under binary verification rewards, response groups may converge to all-correct or all-incorrect outcomes, which reduces reward variance and causes group-relative advantages to weaken or approach zero. Prior work has analyzed this degradation of learning signals from several perspectives, including uniform-reward regimes associated with delayed convergence (Zhang and Zuo, 2025), stalled updates in all-negative groups (Chen et al., 2025a), increasingly severe vanishing advantages (Wang et al., 2025b), and imbalance in update magnitude across questions of different difficulty (Li et al., 2025a). In addition, pathological exploration dynamics have also been reported, including increasingly deterministic policies accompanied by entropy collapse (Li et al., 2025b), oscillation between entropy collapse and entropy explosion (Wu et al., 2025), and weakened superiority signals and reduced exploration in later stages of training (Wang et al., 2025a). Taken together, these findings suggest that reduced rollout diversity directly undermines the comparison signal on which GRPO depends, which in turn can lead to reduced exploration and stalled learning.

Most existing approaches address this issue within a single-policy setting by modifying normalization or objectives, explicitly handling vanishing-signal cases, or injecting diversity-related signals into the reward or weighting scheme (Zhang and Zuo, 2025; Chen et al., 2025a; Li et al., 2025a). DAPO directly targets entropy collapse and training instability by removing the KL penalty, introducing asymmetric clipping to

sustain exploration, and applying token-level loss normalization to stabilize gradient updates (Yu et al., 2025). Similarly, DRA-GRPO incorporates an additional diversity signal to suppress semantic redundancy within a group (Chen et al., 2025b). These methods share the common objective of preserving effective learning signals within a single-policy framework, either by refining the optimization objective or by reweighting rewards.

By contrast, we argue that the fundamental bottleneck lies in the narrowing of the candidate pool itself. We therefore introduce separate diversity policies to structurally expand the comparison set, and integrate their rollouts as collective experience for the main policy. Whereas prior approaches focus on refining *how* learning signals are computed from a given candidate set, CoEx-GRPO directly targets the diversity of the candidate set itself, thereby offering a complementary perspective.