ARMOR: Stabilizing On-Policy LLM RL with Off-Policy Anchor Samples

Anonymous ACL submission

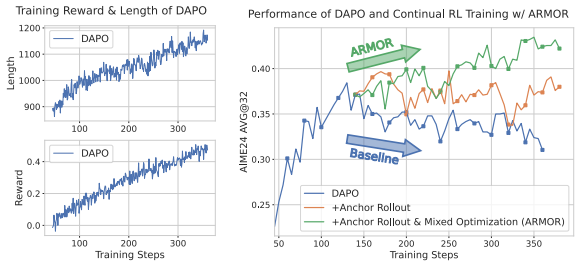
Abstract

Reinforcement learning (RL) has significantly enhanced the reasoning capabilities of large language models (LLMs), yet the training process remains notoriously fragile. In this work, we investigate a critical source of this instability: **over-optimization**, where models exploit training heuristics at the expense of generalizable reasoning. While reverse KL regularization is the standard defense against such degradation, our analysis reveals that it is often insufficient in this regime, as it fails to ensure comprehensive coverage of the reference distribution. To address this, we propose **ARMOR (Anchor Rollout and Mixed Optimization for RL)**, a framework that shifts the paradigm from passive penalty to active sample stabilization. ARMOR comprises two key components: (1) **Anchor Rollout**, which leverages off-policy data from the reference policy to preserve established solution patterns; and (2) **Mixed Optimization**, which reformulates the policy objective to enable controlled exploration without relying on auxiliary losses. Extensive experiments on reasoning benchmarks validate that ARMOR effectively mitigates validation collapse, enabling sustained performance improvements over extended training horizons.

1 Introduction

Reinforcement learning has become a central algorithmic driver of recent advances in large language models, substantially enhancing their ability to solve complex tasks and enabling a new class of reasoning-focused models such as OpenAI o1 (Jaech et al., 2024), DeepSeek R1 (Guo et al., 2025), and Qwen3 (Yang et al., 2025). Yet, scaling RL for reasoning over long horizons remains fragile: **stable optimization can still yield unstable generalization**.

While the community has extensively addressed system-level instabilities, such as the training-inference mismatch (Zheng et al., 2025b; Zhao et al., 2025a; Ma et al., 2025; Zheng et al., 2025a)—an architectural inconsistency that disrupts optimization (Yao et al., 2025; Liu et al., 2025b)—we focus on a more fundamental algorithmic failure: ***over-optimization*** (Gao et al., 2023). As illustrated in Fig. 1, we observe a distinctive *reward–validation gap*: training reward (answer correctness) improves steadily, yet validation performance decouples and degrades. In this regime, stable reward curves can even be a warning sign rather than a reassurance. This phenomenon, echoed in recent works (Mao et al., 2025; Zhang et al., 2025a), indicates a breakdown in generalization rather than an optimization collapse.



(a) Training behaviors (b) Validation score (AIME24)

Figure 1: Illustration of the over-optimization issue and the efficacy of ARMOR (on Qwen2.5-Math-7B). (a) Training reward and response length steadily rise, indicating stable optimization. (b) However, validation performance reveals a critical generalization failure: the baseline performance degrades after an initial ascent. By integrating anchor rollout to stabilize the sample distribution and mixed optimization to enhance exploration, ARMOR prevents degradation and sustains continuous performance gains.

Crucially, this failure differs from the classical form of *reward hacking* that arises from a discrepancy between a proxy reward and the true evaluation metric (Weng, 2024). In reasoning tasks with *verifiable rewards* (Lambert et al., 2024), the reward signal remains consistent across training and validation. Nevertheless, the model still tends to

over-optimize solution patterns that yield high rewards on seen data but fail to generalize, necessitating algorithmic intervention beyond simple reward correction (Ding et al., 2025).

Unfortunately, we find the standard algorithmic defense, reverse KL regularization (Ouyang et al., 2022; Gao et al., 2023), is often insufficient in this regime due to two intrinsic limitations (Fig. 2c): **(1) Mode-seeking nature (Instability):** Reverse KL only penalizes the policy for generating samples *unlikely* under the reference. Consequently, the policy can still collapse onto a narrow subset of "shortcut" patterns without incurring a high penalty, losing the diversity of the reference distribution (Wang et al., 2024a; GX-Chen et al., 2025). **(2) Uniform penalty (Stagnation):** By indiscriminately suppressing deviation from the reference, the KL term dampens the exploration required to discover superior reasoning paths that lie beyond the initial distribution.

To address these limitations, we propose **ARMOR (Anchor Rollout and Mixed Optimization for RL)**, a framework that shifts focus from passive loss penalty to active sample stabilization. Our approach comprises two key components, each targeting a specific failure mode of standard KL:

(1) **Anchor Rollout (Addressing Instability)**: Instead of relying on a passive KL loss to retain distribution modes, we *actively* inject off-policy samples from the reference policy during rollout. This acts as an explicit "anchor" forcing the model to recall and preserve established generalizable solution patterns.

(2) **Mixed optimization (Addressing Stagnation)**: We remove the auxiliary penalty term and instead optimize a mixed policy $\alpha \cdot \pi_\theta + (1-\alpha) \cdot \pi_{\text{ref}}$, to principally align the optimization target and data distribution. Crucially, this mixture constructs an adaptive trust region (Schulman et al., 2017) that permits controlled exploration without the suppressive effect of a uniform penalty.

As visualized in Fig. 1b, ARMOR effectively bridges the generalization gap, maintaining superior validation performance compared to the degraded baseline. We validate these benefits through extensive experiments across different base models (*e.g.,* Qwen2.5-Math-7B, Qwen3-8B-Base) and RL algorithms (*e.g.,* DAPO (Yu et al., 2025), QAE (Wu et al., 2025b)). The results on various reasoning benchmarks confirm that our framework not only secures prolonged training stability but also unlocks continuous performance gains.

2 Preliminaries

Group Relative Policy Optimization (GRPO, Shao et al., 2024). GRPO removes the separate critic model in Proximal Policy Optimization (PPO, Schulman et al., 2017) to enhance training efficiency. Given a QA pair (x, a) from the dataset $\mathcal{D}$, it generates a response group $G = \{y_i\}_{i=1}^{|G|}$ using the current policy $\pi_{\theta_{\text{old}}}$, computes corresponding rewards $\{R_i\}_{i=1}^{|G|}$, and estimates the advantage via:

$$\hat{A}_{i,t} = \frac{R_i - \text{mean}(\{R_i\}_{i=1}^{|G|})}{\text{std}(\{R_i\}_{i=1}^{|G|})}. \tag{1}$$

The policy π_θ is then optimized by maximizing $\mathcal{J}_{\text{GRPO}}(\theta)$, defined as:

$$\mathcal{J} = \mathbb{E}_{\substack{(x,a)\sim\mathcal{D} \\ \{y_i\}_{i=1}^{|G|}\sim\pi_{\theta_{\text{old}}}(\cdot|x)}} \Bigg[\frac{1}{|G|}\sum_{i=1}^{|G|}\frac{1}{|y_i|}\sum_{t=1}^{|y_i|} \min\Big(r_{i,t}(\theta)\hat{A}_{i,t}, \text{clip}\big(r_{i,t}(\theta), 1-\epsilon, 1+\epsilon\big)\hat{A}_{i,t}\Big) - \beta\mathbb{D}_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}})\Bigg],$$

where $r_{i,t}(\theta) = \frac{\pi_\theta(y_{i,t}|x, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t}|x, y_{i,<t})}$ denotes the importance sampling ratio with $1 \pm \epsilon$ being the clipping range, and the KL penalty $\mathbb{D}_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}})$ regularizes the policy towards the reference policy π_{ref}.

Dynamic Sampling Policy Optimization (DAPO, Yu et al., 2025). DAPO is a prominent critic-free RL algorithm that further refines GRPO. It employs token-level loss aggregation and introduces the clip-higher mechanism with two clip range of $\epsilon_{\text{low}}, \epsilon_{\text{high}}$, while removing the KL penalty:

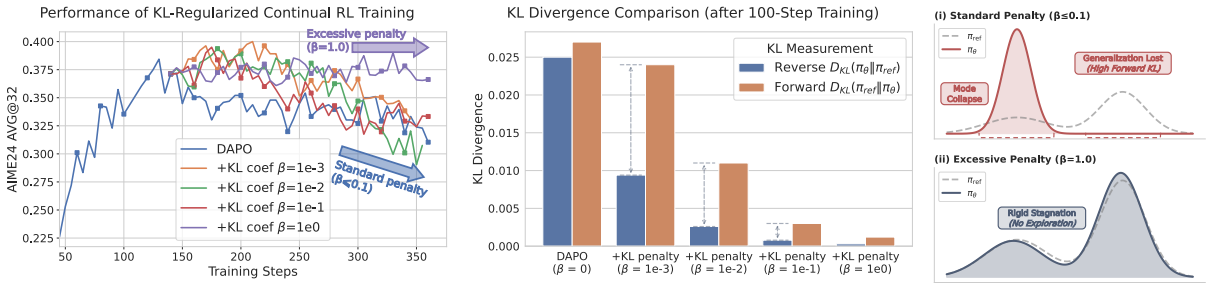
$$\mathcal{J}_{\text{DAPO}}(\theta) = \mathbb{E}_{\substack{(x,a)\sim\mathcal{D} \\ \{y_i\}_{i=1}^{|G|}\sim\pi_{\theta_{\text{old}}}(\cdot|x)}} \Bigg[\frac{1}{\sum_{i=1}^{|G|}|y_i|}\sum_{i=1}^{|G|}\sum_{t=1}^{|y_i|} \min\Big(r_{i,t}(\theta)\hat{A}_{i,t}, \text{clip}\big(r_{i,t}(\theta), 1-\epsilon_{\text{low}}, 1+\epsilon_{\text{high}}\big)\hat{A}_{i,t}\Big)\Bigg], \tag{2}$$

Moreover, DAPO employs a dynamic sampling strategy to ensure the response group is not all correct or all wrong:

$$0 < |\{y_i \mid \text{is_equivalent}(a, y_i)\}| < |G|$$

Given its established efficacy, we adopt DAPO as the primary baseline for our empirical analysis.

KL Regularization. Reverse KL regularization serves as the standard defense against over-optimization in RL training for LLMs (Shah et al.,



(a) KL regularized validation performance (b) Reverse and forward KL measurements (c) An illustrative example

Figure 2: (a) **Performance trends under KL regularization.** Standard penalties ($\beta \leq 0.1$: 1e-3, 1e-2, and 1e-1) merely delay validation collapse of the baseline, while the extreme setting ($\beta = 1.0$) ensures stability only by inducing stagnation, highlighting the stability-plasticity dilemma. (b) **Asymmetry in KL divergence** (measured after 100 steps of continual training). While increasing β drastically reduces the optimized reverse KL $\mathbb{D}_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}})$, it is much less effective at constraining the forward KL $\mathbb{D}_{\text{KL}}(\pi_{\text{ref}} \| \pi_\theta)$. (c) **An illustrative example**. Standard penalties lead to mode collapse (indicated by high Forward KL) due to the mode-seeking nature of the objective, whereas excessive regularization suppresses the exploration required for learning.

2025). Unlike early RLHF methods that apply KL as a reward penalty (Ouyang et al., 2022), reasoning tasks often adopt loss-level regularization to avoid penalizing response length (Shao et al., 2024). We investigate two primary estimators for this term: (1) The variance-reduced unbiased estimator (k3): $\mathbb{D}_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}) = \frac{\pi_{\text{ref}}(\cdot)}{\pi_\theta(\cdot)} - \log \frac{\pi_{\text{ref}}(\cdot)}{\pi_\theta(\cdot)} - 1$, which is used in GRPO; and (2) the squared estimator (k2): $\mathbb{D}_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}) = \frac{1}{2}[\log \pi_{\text{ref}}(\cdot) - \log \pi_\theta(\cdot)]^2$, which is shown to provide unbiased gradients (Tang and Munos, 2025). To ensure rigor, we empirically evaluate both estimators to determine if standard KL can bridge the generalization gap, with k3 adopted for our main results and k2 comparisons deferred to the Appendix.

3 Empirical Analysis of KL Regularization

Experiment Setup. To investigate the efficacy of KL regularization, we adopt a **continual training** setting focused on the phase where over-optimization emerges. We first train Qwen2.5-Math-7B base model (Yang et al., 2024) using DAPO to identify the peak checkpoint π_θ^* immediately prior to validation degradation. We then resume training from π_θ^* under varying KL coefficients, aiming to turn around the degradation trend observed in the unregularized baseline. In this phase, we set the reference policy to the current best model ($\pi_{\text{ref}} \leftarrow \pi_\theta^*$) rather than the initial base model. This setup aligns with practical long-horizon RL training scenarios (Guo et al., 2025), ensuring that the regularization term discourages deviation from the *best-known* policy rather than enforcing regression to the weaker initial state[1].

Evaluation Protocol. We evaluate reasoning performance on the challenging AIME24 benchmark and report avg@k, the average accuracy with $k = 32$ responses sampled per question. Beyond performance metric, we analyze distributional shifts by monitoring both the forward KL divergence $\mathbb{D}_{\text{KL}}(\pi_{\text{ref}} \| \pi_\theta)$ and the reverse KL divergence $\mathbb{D}_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}})$. Tracking both directions provides a holistic view of the policy drift: while reverse KL serves as the training penalty, forward KL offers deeper insights into whether the model maintains adequate coverage of the reference distribution.

Empirical Results. As illustrated in Fig. 2a, the unregularized baseline (DAPO) suffers from immediate degradation in validation performance as training proceeds. Introducing KL penalties reveals a fundamental trade-off:

1) Standard Regime ($\beta \leq 0.1$): It is worth noting that standard RL practices typically employ minimal penalties[2]. Even when increasing β to 0.1, the model eventually succumbs to **instability** after a brief period of improvement.

2) Excessive Regime ($\beta = 1.0$): We further increase β to 1.0 as a stress test to probe the limits of regularization. Although this extreme setting can suppress instability, it leads to **stagnation**. This indicates that such stability is achieved only by rigidly tethering the policy, which precludes the exploration necessary for performance gains.

These observations hold consistently across both

[1]We also explored regularizing against the base model, but this led to rapid performance deterioration.

[2]DeepSeek-R1 utilizes β=1e-3 (Guo et al., 2025).

k2 and k3 estimators (*cf.* Appendix C), indicating that reverse KL is insufficient for stable *and* improving RL training.

Analysis: Why KL Fails. To understand the mechanics of this failure, we examine the divergence metrics in Fig. 2b. A clear **asymmetry** emerges: while increasing β drastically compresses the optimized reverse KL $\mathbb{D}_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}})$, the forward KL $\mathbb{D}_{\text{KL}}(\pi_{\text{ref}} \| \pi_\theta)$ *decreases at a much slower rate*, maintaining a disproportionately large gap. This failure stems from two distinct mechanisms:

1) Mode-Seeking Nature (Standard Regime). Under standard penalties, the optimization is dominated by the **mode-seeking nature** of reverse KL (Wang et al., 2024a; GX-Chen et al., 2025). Fundamentally, reverse KL only penalizes the policy for generating samples that are *unlikely* under the reference distribution, but *not* for missing valid regions of the reference support. Consequently, the model can collapse onto a narrow subset of "shortcut" modes without incurring a high penalty, provided these modes exist within the reference distribution (as illustrated in Fig. 2c). This selective collapse discards other generalizable reasoning paths, resulting in the observed performance degradation and a much higher forward KL (which penalizes dropped modes).

2) Uniform Penalty (Excessive Regime). Under extreme penalties (*e.g.,* $\beta = 1.0$), the regularization acts as a **uniform penalty** on all policy deviations. While this forces $\mathbb{D}_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}) \to 0$ and thus locally aligns forward and reverse KL (Tang and Munos, 2025), it indiscriminately suppresses both harmful degradation and beneficial exploration. This causes the stagnation observed in our experiments, rendering the ineffective training.

Summary

Standard KL regularization faces a dilemma: loose constraints lead to **instability** due to its *mode-seeking nature*, while tight constraints result in **stagnation** due to its *uniform penalty*. This analysis confirms that a passive penalty alone is structurally insufficient.

4 The ARMOR Framework

Building on our analysis, we introduce ARMOR (Anchor Rollout and Mixed Optimization for RL). This framework replaces the passive KL penalty with an active stabilization mechanism comprising two components, each explicitly targeting a failure mode identified in Section 3.

4.1 Anchor Rollout: Active Mode Retention

To address the mode-seeking nature of reverse KL, where the model tends to collapse by forgetting diverse solutions present in the reference, we propose **Anchor Rollout**. Instead of relying on loss penalties to implicitly retain modes, we *actively* inject reference samples into the training batch. Specifically, for each query x, we construct a hybrid response group G during rollout:

$$\begin{aligned} y_i &\sim \pi_{\theta_{\text{old}}}(\cdot|x), \quad 1 \le i \le |G|-1, \\ y_{\text{anc}} &\sim \pi_{\text{ref}}(\cdot|x), \quad \text{s.t. } R(x, y_{\text{anc}}) = 1. \end{aligned} \tag{3}$$

Here, the auxiliary response y_{anc} serves as an "anchor sample" drawn from the reference policy π_{ref} and is guaranteed to be correct[3] ($R = 1$) through rejection sampling (Algo. 1, Line 11). By including this anchor in every response group, we provide an explicit signal preventing the policy π_θ from drifting away from known correct solutions. As shown in Fig. 1b, **this simple intervention effectively stabilizes training** where standard KL fails.

From an implementation perspective, our continual training setting initializes π_{ref} from the best checkpoint of π_θ, ensuring identical model architectures. This design streamlines deployment: As outlined in Algo. 1 (Line 9), we can synchronize the reference parameters θ_{ref} to the inference engine alongside standard updates. This enables hybrid sampling without the computational overhead of switching between distinct generation engines.

4.2 Mixed Optimization: Performance Ceiling

While Anchor Rollout guarantees stability, our objective extends beyond collapse prevention to maximizing the model's asymptotic performance. To achieve this, we first revisit the theoretical implications of our rollout strategy, observing that by injecting anchor samples, Anchor Rollout implicitly constructs **mixture policies** regularized by the reference:

$$\begin{aligned} \pi_\theta^{\text{mix}} &= \alpha \pi_\theta + (1-\alpha)\pi_{\text{ref}}, \\ \pi_{\theta_{\text{old}}}^{\text{mix}} &= \alpha \pi_{\theta_{\text{old}}} + (1-\alpha)\pi_{\text{ref}} \end{aligned} \tag{4}$$

where $\alpha \in (0, 1)$ is the mixing coefficient. Since the data is generated from this mixture, the **principled optimization objective** should align with this structure. Therefore, rather than optimizing π_θ in

[3]This positive-only anchor is critical, as including negative samples drove the model *away* from the reference policy, which will lead to increased instability (Liu et al., 2025a).

Algorithm 1 ARMOR Training Framework

1: **Input:** Initial parameters $\theta^0, \theta_{\text{ref}} \leftarrow \theta^0$, Dataset $\mathcal{D}$, hyperparameters B, μ, α, τ.
2: **for** step $t = 1, \ldots, T$ **do**
3: // Phase 1: Anchor Rollout
4: Initialize batch $\mathcal{B} \leftarrow \emptyset$
5: **while** $|\mathcal{B}| < B$ **do**
6: Sample query batch $\mathcal{X} \sim \mathcal{D}$
7: // Shared engine by switching parameter
8: Load θ^{t-1}, sample on-policy groups $\mathcal{Y}_{\text{on}}$
9: Load θ_{ref}, sample off-policy groups $\mathcal{Y}_{\text{off}}$
10: Compute rewards $\mathcal{R}_{\text{on}}$ and $\mathcal{R}_{\text{off}}$
11: // Rejection sampling
12: **for** $x \in \mathcal{X}$ **do**
13: **if** $\text{Var}(\mathcal{R}_{\text{on}|x}) > 0, \max(\mathcal{R}_{\text{off}|x}) = 1$
14: Let $y_{\text{anc}} = \arg\max_{y \in \mathcal{Y}_{\text{off}|x}} R(x, y)$
15: $G \leftarrow \mathcal{Y}_{\text{on}|x} \cup \{y_{\text{anc}}\}$
16: $\mathcal{B} \leftarrow \mathcal{B} \cup \{(x, G)\}$
17: **end if**
18: **end for**
19: **end while**
20: // Phase 2: Mixed Optimization
21: **for** iteration = $1, \ldots, \mu$ **do**
22: Update θ^t by maximizing $\mathcal{J}(\theta)$, using the mixed IS ratio $r_{i,t}^{\text{mix}}$ (Eq. 5)
23: **end for**
24: // (Optional) Reference Reset
25: **if** $t \mod \tau = 0$
26: $\theta_{\text{ref}} \leftarrow \theta^t$
27: **end if**
28: **end for**

isolation, we reformulate the objective to optimize the mixture policy itself.

We implement this by substituting the standard probability terms in the policy gradient (e.g., DAPO) with their mixed variants, yielding the mixed Importance Sampling (IS) ratio:

$$r_{i,t}^{\text{mix}}(\theta) = \frac{\alpha \pi_\theta(\cdot) + (1-\alpha)\pi_{\text{ref}}(\cdot)}{\alpha \pi_{\theta_{\text{old}}}(\cdot) + (1-\alpha)\pi_{\text{ref}}(\cdot)}. \quad (5)$$

This formulation explicitly accounts for the reference policy's contribution to the data distribution. Crucially, since π_{ref} is fixed within each step, any performance gain achieved on the mixture target π_θ^{mix} strictly translates to improvements in the target policy π_θ:

$$\mathbb{E}_{y \sim \pi_\theta^{\text{mix}}(\cdot|x), y' \sim \pi_{\theta_{\text{old}}}^{\text{mix}}(\cdot|x)}[R(x,y) - R(x,y')] > 0$$
$$\Rightarrow \mathbb{E}_{y \sim \pi_\theta(\cdot|x), y' \sim \pi_{\theta_{\text{old}}}(\cdot|x)}[R(x,y) - R(x,y')] > 0.$$

Furthermore, we follow the iterative updating paradigm (Guo et al., 2025) by periodically resetting π_{ref} to the current policy state (Algo. 1, Line 24), avoiding saturation due to a fixed anchor.

We designate this holistic approach—integrating the stabilization of Anchor Rollout with Mixed Optimization that calibrates the IS ratio—as **ARMOR**. As demonstrated in Fig. 1b, this combination yields a **significantly higher performance ceiling**.

4.3 Explanation: Adaptive Trust Region

To further elucidate how Mixed Optimization enables superior asymptotic performance, we analyze its impact on policy update dynamics through the lens of the trust-region.

PPO variants (*e.g.,* GRPO, DAPO) clip the positive/negative sample's IS ratio into fixed intervals $[0, 1+\epsilon]$ or $[1-\epsilon, \infty)$ to enforce a trust region $\pi_\theta \approx \pi_{\theta_{\text{old}}}$. However, substituting our mixed ratio yields a flexible boundary. Considering the clipping boundaries $r_{i,t}^{\text{mix}}(\theta) = 1 \pm \epsilon$, we have:

$$\frac{\alpha \cdot \pi_\theta + (1-\alpha) \cdot \pi_{\text{ref}}}{\alpha \cdot \pi_{\theta_{\text{old}}} + (1-\alpha) \cdot \pi_{\text{ref}}} = 1 \pm \epsilon.$$

Solving for the current policy π_θ, we derive the **effective trust region boundary** (Schulman et al., 2015, 2017):

$$\pi_\theta = \underbrace{(1 \pm \epsilon) \cdot \pi_{\theta_{\text{old}}}}_{\text{Standard Boundary}} \pm \underbrace{\epsilon(1-\alpha)/\alpha \cdot \pi_{\text{ref}}}_{\text{Expansion Term } (\geq 0)}. \quad (6)$$

This derivation reveals a critical mechanism: In contrast to the standard clip boundary $\pi_\theta = (1 \pm \epsilon) \cdot \pi_{\theta_{\text{old}}}$, ARMOR **dynamically expands the clipping range** proportional to the reference probability π_{ref}. This derivation reveals that ARMOR dynamically modulates the update magnitude based on the reference prior, enabling a dual mechanism for performance breakthroughs:

1) Reinforcing verified correctness: For positive updates $(1+\epsilon)$, the higher upper bound allows larger steps towards correct actions supported by the reference. This is crucial for effectively learning from *off-policy* correct samples, ensuring they are not unfairly clipped due to distribution shift.

2) Rectifying reference biases: For negative updates $(1-\epsilon)$, the lowered boundary permits stronger penalization of incorrect actions, even if they are highly probable in π_{ref}. This mechanism is key to *surpassing* the reference capability, as it allows the model to decisively correct the reference's inherent biases rather than blindly imitating them.

Table 1: Comparison of RL baselines (DAPO, QAE) against their ARMOR continual-training counterparts across Qwen2.5-Math-7B and Qwen3-8B-Base. ARMOR consistently improves mathematical reasoning (`avg@32`) on AIME and AMC benchmarks while maintaining comparable general capabilities (`avg@8` on GPQA and MMLU-Pro), effectively mitigating the over-optimization observed in baselines.

Model	Method	Step	Math Reasoning Tasks				General Tasks	
			AIME24	AIME25	AMC	Average	GPQA	MMLU-Pro
Qwen2.5-Math-7B	DAPO	140	37.13	15.21	69.39	40.58	38.26	43.93
	+ARMOR	+200	$43.04^{+5.91}$	$18.13^{+2.92}$	$76.13^{+6.74}$	$45.77^{+5.19}$	$42.49^{+4.23}$	$46.03^{+2.1}$
	QAE	230	39.79	15.96	73.53	43.09	40.78	43.30
	+ARMOR	+240	$41.58^{+1.79}$	$16.15^{+0.19}$	$76.62^{+3.09}$	$44.78^{+1.69}$	$39.02^{-1.76}$	$42.15^{-1.15}$
Qwen3-8B-Base	DAPO	70	36.98	28.13	71.72	45.61	49.87	65.55
	+ARMOR	+220	$48.13^{+11.15}$	$34.48^{+6.35}$	$80.35^{+8.63}$	$54.32^{+8.71}$	$56.25^{+6.38}$	$68.90^{+3.35}$

Summary

ARMOR resolves the stability-exploration dilemma by decoupling the constraints: **Anchor Rollout** prevents **instability** by actively retaining correct reference solutions, while **Mixed Optimization** avoids **stagnation** by constructing an adaptive trust region. Empirically, ARMOR secures continuous reasoning improvements without the fragility of standard KL regularization.

5 Experiments

In this section, we provide a comprehensive empirical evaluation of ARMOR. We first present the main results across different base models and RL algorithms to validate the framework's broad effectiveness in mitigating over-optimization. Subsequently, we conduct detailed ablation studies to dissect the individual contributions of the two core components in ARMOR, as well as the impact of various reference policy settings.

5.1 Main Results

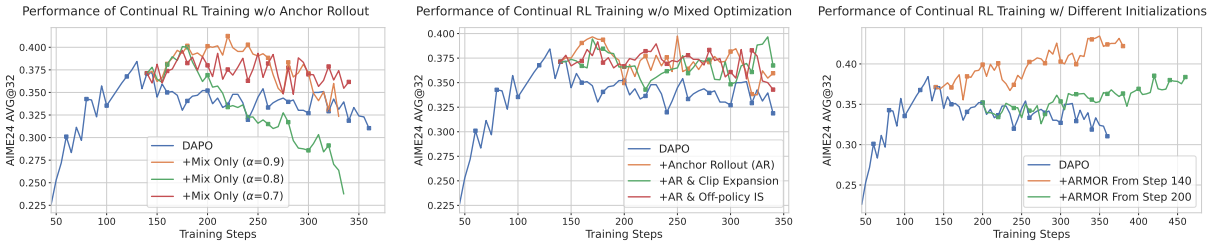
Experimental Setup. To verify that ARMOR effectively addresses the over-optimization problem, we adopt the **continual training** setting described in Section 3. We employ two base models: Qwen2.5-Math-7B (Yang et al., 2024) and Qwen3-8B-Base (Yang et al., 2025), trained on the DAPO-Math-17K dataset. For the underlying RL algorithms, we select DAPO (Yu et al., 2025) and QAE (Wu et al., 2025b). QAE represents a state-of-the-art method with a quantile-based advantage function, which we integrate into our framework by replacing the advantage term in Eq. (1).

The training protocol proceeds in two stages: we first run the baseline algorithm until validation performance degrades due to over-optimization. We then apply ARMOR to continue training from the best checkpoint (identified via the AIME24 validation set) prior to performance collapse. Detailed experimental setting and hyperparameters are provided in Appendix B.

Benchmarks and Metrics. We evaluate performance on both mathematical reasoning and general capabilities. For reasoning, we report both the average accuracy (`avg@k`) and `pass@k` (Chen et al., 2021) on challenging mathematical benchmarks, including AIME24/25 and AMC. We also evaluate the model's general capabilities on science reasoning (GPQA, Rein et al., 2024) and language understanding (MMLU-Pro, Wang et al., 2024b). For math reasoning tasks, we sample 32 responses to derive a robust estimation. For general tasks, we use 8 samples due to the significantly larger query volume. All reported checkpoints are selected based on their AIME24 performance.

Performance Analysis. As summarized in Tab. 1, while standard training suffers from over-optimization (so the select checkpoints undergo fewer training steps), ARMOR successfully stabilizes the training process and secures sustained performance gains beyond the initial peak:

- **Significant Gains with DAPO:** On both Qwen2.5 and Qwen3 variants, applying ARMOR to DAPO yields substantial improvements, boosting average math scores by **+5–8 points**. Notably, this reasoning gain does not come at the expense of general capabilities, which also see clear enhancement by **+2–6 points**.
- **Robustness with QAE:** When applied to QAE, ARMOR further improves mathematical reasoning by **+1.7 points**, validating its robustness across different algorithms. While we observe



(a) Without Anchor Rollout (b) Without Mixed Optimization (c) Robustness to Initial Model Quality

Figure 3: (a) **Necessity of Anchor Rollout**. We compare DAPO against variants using *only* Mixed Optimization (α = 0.9, 0.8, and 0.7). Without the stability foundation provided by Anchor Rollout, these models suffer from eventual performance collapse. (b) **Necessity of Mixed Optimization**. We replace mixed optimization with alternative objectives on top of Anchor Rollout (AR). While AR alone effectively stabilizes training, modifying standard objectives with comparable clip range or off-policy IS fails to yield further improvements. (c) **Ablation of different initializations**. We apply ARMOR to models initialized from an optimal checkpoint (Step 140) versus a collapsed checkpoint (Step 200). Remarkably, ARMOR successfully recovers performance even from a degraded state, although initiating from a higher-quality model enables a higher final performance peak.

Table 2: `Pass@k` evaluation of ARMOR compared to baselines. The checkpoints used for evaluation are consistent with Table 1. ARMOR maintains or improves `pass@16` despite extended training horizons, indicating robust intrinsic reasoning improvements.

	AIME24	AIME25	AMC	Average
7B-DAPO	55.75	34.96	90.82	60.51
+ARMOR	56.52	39.44	90.89	**62.28**
7B-QAE	51.21	30.71	88.52	56.81
+ARMOR	50.33	34.08	86.12	**56.84**
8B-DAPO	70.65	50.88	90.24	70.59
+ARMOR	74.60	51.42	90.66	**72.23**

a slight regression in general capabilities, we attribute this to two factors: the extended training horizon naturally increases the risk of forgetting for general knowledge, and more importantly, QAE's advantage masking mechanism[4] can inadvertently discard our injected anchor samples y_{anc}, thereby weakening the regularization effect (We also observe a larger $\mathbb{D}_{\text{KL}}(\pi_{\text{ref}}\|\pi_\theta)$ than DAPO's in our experiments).

Overall, ARMOR consistently extends the effective training window and pushes the reasoning performance ceiling across all tested settings. As a two-stage recipe, ARMOR also outperforms teacher-guided OPD (Agarwal et al., 2024) and MOPD (Xiao et al., 2026) baselines initialized from the same best zero-RL checkpoint (Appendix C.2).

`Pass@k` Evaluation. Beyond average performance, we investigate intrinsic reasoning capability via the `pass@k` metric (Yue et al., 2025). Tab. 2 reports `pass@16` results on three math benchmarks (Chen et al., 2021). ARMOR achieves comparable or improved scores across all models and baselines, confirming that our method effectively raises the model's absolute problem-solving ceiling rather than simply trading off `pass@k` for accuracy.

5.2 Ablation Study

To understand the source of ARMOR's gains, we dissect the framework to analyze the necessity of Anchor Rollout, Mixed Optimization, and the impact of reference policy.

5.2.1 Necessity of Anchor Rollout

We first test a variant using only Mixed Optimization (Eq. 5) without injecting anchor samples, so the only difference from DAPO here is the replaced IS ratio: $r_{i,t}^{\text{mix}}$. As shown in Fig. 3a, on the one hand, this variant can achieve initial growth across different mixing coefficients α, with the peak performance increased from 37 to 40+, showcasing Mixed Optimization's effect on improving the performance ceiling. On the other hand, without the explicit retention provided by anchor samples, the model eventually degenerates.

This result demonstrates that the exploration capability provided by Mixed Optimization alone is insufficient for stable RL scaling; **Anchor Rollout is essential for providing the stability floor** required to prevent collapse.

5.2.2 Necessity of Mixed Optimization

Next, we remove Mixed Optimization and evaluate Anchor Rollout paired with three alternative

[4]QAE uses reward quantiles as an advantage baseline, naturally producing zero-advantage samples (typically ∼80% of the batch).

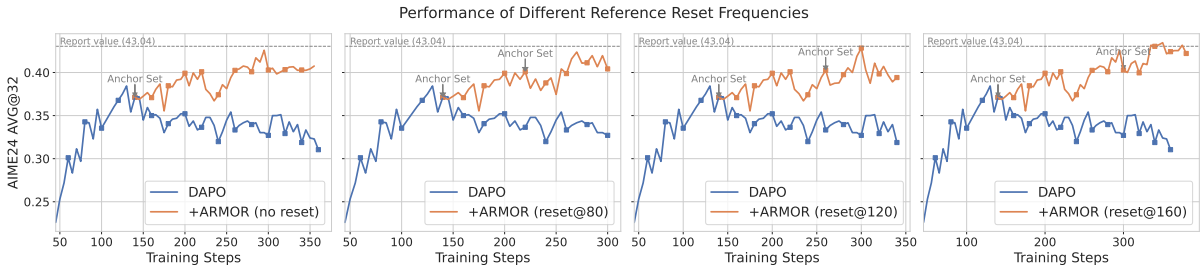


Figure 4: **Impact of Reference Reset Frequency.** We compare ARMOR training with a static anchor (no reset) versus resetting the reference policy every 80, 120, or 160 steps (noted by "Anchor Set" in the figure). Periodic resets allows the anchor to evolve with the policy, preventing stagnation and fostering higher reasoning accuracy.

optimization objectives: *(a) Standard DAPO*: default objective with no modification to the IS ratio; *(b) Static Clip Expansion*: DAPO with a manually expanded clipping boundary $1 \pm \epsilon/\alpha$, designed to match the theoretical effective clip boundary of Mixed Optimization; *(c) Standard Off-Policy IS*: Setting IS ratio based on the sampling distribution:

$$r_{i,t}^{\text{off}} = \begin{cases} \pi_\theta(y_{i,t}|x)/\pi_{\theta_{old}}(y_{i,t}|x) & 1 \le i \le |G|-1, \\ \pi_\theta(y_{i,t}|x)/\pi_{\text{ref}}(y_{i,t}|x) & i = |G|. \end{cases}$$

Note that we include (b) to ensure a comprehensive ablation: As derived in Sec. 4.3, the clip boundary of mixed optimization is *dynamically* broadened to $(1\pm\epsilon)\pi_{\theta_{\text{old}}} \pm \epsilon(1-\alpha)/\alpha \cdot \pi_{\text{ref}}$. If we were to replace π_{ref} with $\pi_{\theta_{\text{old}}}$, this boundary would simplify to $(1 \pm \epsilon/\alpha)\pi_{\theta_{\text{old}}}$. To verify that the gain of Mixed Optimization comes from the specific inclusion of π_{ref} rather than simply a wider clip range, we test this *static* expanded clip boundary of $1 \pm \epsilon/\alpha$ in baseline (b). We also ensure that the hyperparameter α here is identical to the one used in our ARMOR experiments for a fair comparison.

As illustrated in Fig. 3b, while all three alternatives successfully stabilize training (thanks to Anchor Rollout), they hit a significantly lower performance ceiling compared to the full ARMOR framework. Baseline (b)'s failure specifically highlights that simple expansion is insufficient; the gain comes from the *adaptive trust region* that selectively reinforces verified correctness and rectifies reference biases. This demonstrates that ARMOR's advantage is two-fold: **Anchor Rollout ensures stability, while Mixed Optimization unlocks the exploration capability required to reach a higher ceiling.**

5.2.3 Impact of Reference Policy

Finally, we analyze the impact of the reference policy, focusing on reset frequency and anchor quality.

1) Impact of Reset Frequency. We compared training without resets against resetting the anchor after 80, 120, and 160 steps (Line 24 in Algo. 1). As shown in Fig. 4, while all ARMOR variants outperform the baseline, periodic resets consistently achieve a higher performance ceiling than no reset. This confirms that the anchor must evolve alongside the model to avoid becoming a bottleneck.

2) Robustness to Initial Anchor Quality. We tested initializing ARMOR from two distinct points: a *Peak Checkpoint* (step 140, optimal performance) and a *Degenerated Checkpoint* (step 200, after performance drop). Both settings underwent a consistent reset frequency of 160 for a fair comparison. Remarkably, ARMOR yielded clear improvements in both cases (Fig. 3c), demonstrating its ability to recover a failing model. However, starting from the higher-quality anchor allowed the model to reach a markedly higher final ceiling. These results emphasize that while ARMOR is robust, a well-timed application strategy is critical for maximizing ultimate reasoning capabilities.

6 Conclusion

In this work, we tackle the critical challenge of *over-optimization* in scaling RL for reasoning. We identify that standard reverse KL regularization is insufficient due to an intrinsic **stability-exploration dilemma**: its mode-seeking nature leads to mode collapse, while uniform constraints cause stagnation. To resolve this, we introduce **ARMOR**, a framework that synergizes **Anchor Rollout** to actively stabilize the sample distribution and **Mixed Optimization** to construct an adaptive trust region for exploration. Extensive experiments confirm that ARMOR effectively prevents degradation and unlocks sustained performance gains across diverse training settings.

Limitations

While ARMOR offers a robust solution for RL scaling, we acknowledge several limitations that point toward future directions: *(1) Computational Overhead:* Incorporating anchor samples through rejection sampling incurs additional inference cost. While we regard this as a justified investment for stable scaling, future work could mitigate this cost with an offline replay buffer to recycle high-quality anchors. *(2) Additional Hyperparameter:* Currently, the reference reset strategy is determined manually. Future work could explore adaptive scheduling to automate this process. *(3) Theoretical Analysis:* While our trust-region analysis explains how Mixed Optimization safely incorporates more signals, establishing a theoretical guarantee linking this adaptation to superior convergence or optimality remains an open challenge.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, and 1 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, and 1 others. 2025. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*.

Wenhao Deng, Long Wei, Chenglei Yu, and Tailin Wu. 2025. Unlocking reasoning capabilities in llms via reinforcement learning exploration. *arXiv preprint arXiv:2510.03865*.

Yuyang Ding, Chi Zhang, Juntao Li, Haibin Lin, Xin Liu, and Min Zhang. 2025. Fapo: Flawed-aware policy optimization for efficient and reliable reasoning. *arXiv preprint arXiv:2510.22543*.

Leo Gao, John Schulman, and Jacob Hilton. 2023. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pages 10835–10866. PMLR.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, and 1 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645:633–638.

Anthony GX-Chen, Jatin Prakash, Jeff Guo, Rob Fergus, and Rajesh Ranganath. 2025. Kl-regularized reinforcement learning is designed to mode collapse. *arXiv preprint arXiv:2510.20817*.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. 2025. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Zishang Jiang, Jinyi Han, Tingyun Li, Xinyi Wang, Sihang Jiang, Jiaqing Liang, Zhaoqian Dai, Shuguang Ma, Fei Yu, and Yanghua Xiao. 2025. Selective expert guidance for effective and diverse exploration in reinforcement learning of llms. *arXiv preprint arXiv:2510.04140*.

Adam Tauman Kalai, Ofir Nachum, Santosh S Vempala, and Edwin Zhang. 2025. Why language models hallucinate. *arXiv preprint arXiv:2509.04664*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *SOSP*, pages 611–626.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, and 1 others. 2024. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*.

Long Li, Zhijian Zhou, Jiaran Hao, Jason Klein Liu, Yanting Miao, Wei Pang, Xiaoyu Tan, Wei Chu, Zhe Wang, Shirui Pan, and 1 others. 2025a. The choice of divergence: A neglected key to mitigating diversity collapse in reinforcement learning with verifiable reward. *arXiv preprint arXiv:2509.07430*.

Siheng Li, Zhanhui Zhou, Wai Lam, Chao Yang, and Chaochao Lu. 2025b. Repo: Replay-enhanced policy optimization. *arXiv preprint arXiv:2506.09340*.

Jing Liang, Hongyao Tang, Yi Ma, Jinyi Liu, Yan Zheng, Shuyue Hu, Lei Bai, and Jianye Hao. 2025. Squeeze the soaked sponge: Efficient off-policy reinforcement finetuning for large language model. *arXiv preprint arXiv:2507.06892*.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, and 1 others. 2025a. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*.

Jiacai Liu, Yingru Li, Yuqian Fu, Jiawei Wang, Qian Liu, and Zhuo Jiang. 2025b. When speed kills stability: Demystifying RL collapse from the training-inference mismatch.

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

Xingtai Lv, Yuxin Zuo, Youbang Sun, Hongyi Liu, Yuntian Wei, Zhekai Chen, Lixuan He, Xuekai Zhu, Kaiyan Zhang, Bingning Wang, and 1 others. 2025. Towards a unified view of large language model post-training. *arXiv preprint arXiv:2509.04419*.

Wenhan Ma, Hailin Zhang, Liang Zhao, Yifan Song, Yudong Wang, Zhifang Sui, and Fuli Luo. 2025. Stabilizing moe reinforcement learning by aligning training and inference routers. *Preprint*, arXiv:2510.11370.

Hanyi Mao, Quanjia Xiao, Lei Pang, and Haixiao Liu. 2025. Clip your sequences fairly: Enforcing length fairness for sequence-level rl. *arXiv preprint arXiv:2509.09177*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

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

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR.

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

Vedant Shah, Johan Obando-Ceron, Vineet Jain, Brian Bartoldson, Bhavya Kailkhura, Sarthak Mittal, Glen Berseth, Pablo Samuel Castro, Yoshua Bengio, Nikolay Malkin, and 1 others. 2025. A comedy of estimators: On kl regularization in rl training of llms. *arXiv preprint arXiv:2512.21852*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:3008–3021.

Yunhao Tang and Rémi Munos. 2025. On a few pitfalls in kl divergence gradient estimation for rl. *arXiv preprint arXiv:2506.09477*.

Chaoqi Wang, Yibo Jiang, Chenghao Yang, Han Liu, and Yuxin Chen. 2024a. Beyond reverse KL: Generalizing direct preference optimization with diverse divergence constraints. In *The Twelfth International Conference on Learning Representations*.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. 2025. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for LLM reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, and 1 others. 2024b. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290.

Lilian Weng. 2024. Reward hacking in reinforcement learning. *lilianweng.github.io*.

Junkang Wu, Kexin Huang, Xue Wang, Jinyang Gao, Bolin Ding, Jiancan Wu, Xiangnan He, and Xiang Wang. 2025a. RePO: Understanding preference learning through reLU-based optimization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Junkang Wu, Kexin Huang, Jiancan Wu, An Zhang, Xiang Wang, and Xiangnan He. 2025b. Quantile advantage estimation for entropy-safe reasoning. *Preprint*, arXiv:2509.22611.

Bangjun Xiao, Bingquan Xia, Bo Yang, Bofei Gao, Bowen Shen, Chen Zhang, Chenhong He, Chiheng Lou, Fuli Luo, Gang Wang, and 1 others. 2026. Mimo-v2-flash technical report. *arXiv preprint arXiv:2601.02780*.

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. 2025. Learning to reason under off-policy guidance. *arXiv preprint arXiv:2504.14945*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *Preprint*, arXiv:2409.12122.

Feng Yao, Liyuan Liu, Dinghuai Zhang, Chengyu Dong, Jingbo Shang, and Jianfeng Gao. 2025. Your efficient rl framework secretly brings you off-policy rl training.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, and 1 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

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 *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Runzhe Zhan, Yafu Li, Zhi Wang, Xiaoye Qu, Dongrui Liu, Jing Shao, Derek F Wong, and Yu Cheng. 2025. Exgrpo: Learning to reason from experience. *arXiv preprint arXiv:2510.02245*.

Han Zhang, Ruibin Zheng, Zexuan Yi, Zhuo Zhang, Hanyang Peng, Hui Wang, Zike Yuan, Cai Ke, Shiwei Chen, Jiacheng Yang, and 1 others. 2025a. Gepo: Group expectation policy optimization for stable heterogeneous reinforcement learning. *arXiv preprint arXiv:2508.17850*.

Hongzhi Zhang, Jia Fu, Jingyuan Zhang, Kai Fu, Qi Wang, Fuzheng Zhang, and Guorui Zhou. 2025b. Rlep: Reinforcement learning with experience replay for llm reasoning. *arXiv preprint arXiv:2507.07451*.

Xin Zhao, Yongkang Liu, Kuan Xu, Jia Guo, Zihao Wang, Yan Sun, Xinyu Kong, Qianggang Cao, Liang Jiang, Zujie Wen, Zhiqiang Zhang, and Jun Zhou. 2025a. Small leak can sink a great ship–boost rl training on moe with icepop!

Yuzhong Zhao, Yue Liu, Junpeng Liu, Jingye Chen, Xun Wu, Yaru Hao, Tengchao Lv, Shaohan Huang, Lei Cui, Qixiang Ye, and 1 others. 2025b. Geometric-mean policy optimization. *arXiv preprint arXiv:2507.20673*.

Chujie Zheng, Kai Dang, Bowen Yu, Mingze Li, Huiqiang Jiang, Junrong Lin, Yuqiong Liu, Hao Lin, Chencan Wu, Feng Hu, An Yang, Jingren Zhou, and Junyang Lin. 2025a. Stabilizing reinforcement learning with llms: Formulation and practices. *Preprint*, arXiv:2512.01374.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, and 1 others. 2025b. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*.

A Related Work

Reinforcement learning for LLM. While early foundational RL works in LLM post-training centered on *Reinforcement Learning from Human Feedback* (RLHF) to align models with human preferences (Stiennon et al., 2020; Ouyang et al., 2022), the RL paradigm has recently shifted towards *Reinforcement Learning with Verifiable Rewards* (RLVR) for verifiable tasks such as mathematics and coding (Lambert et al., 2024). Under this regime, OpenAI o1 (Jaech et al., 2024) is a seminal reasoning-oriented LLM built by RL, while DeepSeek-R1 (Guo et al., 2025) introduces a detailed recipe using GRPO (Shao et al., 2024). These milestones have catalyzed a surge of research dedicated to reproducing reasoning behaviors (Hu et al., 2025; Yu et al., 2025), investigating underlying mechanisms such as entropy dynamics (Yue et al., 2025; Cui et al., 2025; Wang et al., 2025), and refining learning algorithms (Liu et al., 2025c; Wu et al., 2025b; Zhao et al., 2025b). In this work, we adopt DAPO (Yu et al., 2025) as our primary backbone.

Stability Issues in LLM RL. Despite these advancements, scaling RL training faces significant stability challenges: *(1) System-level Instability.* RL frameworks typically decouple rollout (*e.g.,* vllm (Kwon et al., 2023)) and training engines (*e.g.,* Megatron (Shoeybi et al., 2019)) to maximize throughput, introducing a *training-inference mismatch* (Yao et al., 2025; Liu et al., 2025b). This architectural inconsistency, which is exacerbated in Mix-of-Expert (MoE) models by diverging router behaviors (Zheng et al., 2025b), can destabilize optimization. Although extensive existing works address this problem (Zhao et al., 2025a; Ma et al., 2025; Zheng et al., 2025a), stable optimization can still yield unstable generalization, and we focus on this fundamental challenge: *(2) Algorithmic Over-optimization.* This issue arises when stable improvement in training rewards paradoxically leads to degraded generalization (Gao et al., 2023; Wu et al., 2025a). Unlike classic *Reward Hacking* driven by proxy reward flaws (Weng, 2024), this phenomenon persists in RLVR even when reward signals are consistent with evaluation. While some studies attribute this to "false positive" reasoning (Kalai et al., 2025; Ding et al., 2025), we argue that reward shaping might not be the sole cure. Given that imperfect outcome rewards remain highly effective (Ding et al., 2025) and evaluation metrics mirror training objectives, the degradation implies a fundamental **algorithmic tendency toward over-optimization**. Thus, we advocate for algorithmic interventions to enforce stability, serving as a necessary foundation that complements reward modeling.

Learning with off-policy guidance. Using off-policy data is a common strategy in RL, typically serving two purposes: efficiency and distillation. For example, methods like Experience Replay (Li et al., 2025b; Liang et al., 2025; Zhan et al., 2025) and unified SFT-RL frameworks (Yan et al., 2025; Lv et al., 2025; Jiang et al., 2025) utilize historical or expert trajectories alongside on-policy generations for improved efficiency or better performance. In addition, Zhang et al. (2025b) explored a two-stage training paradigm, where an optimized RL model guides a base model to bootstrap capabilities. While ARMOR also utilizes off-policy samples y_{anc} from π_{ref} and employs a two-stage training paradigm, it differs fundamentally in its target: We utilize the reference not to bootstrap a new model, but to serve as a dynamic anchor that sustains the model's own scaling and explicitly counteracts the tendency toward over-optimization.

KL Regularization and Divergence Choices. KL regularization is widely used to constrain policy updates and mitigate reward over-optimization in RLHF/RLVR (Ouyang et al., 2022; Gao et al., 2023; Shah et al., 2025). However, reverse KL is mode-seeking and can still allow the policy to collapse onto a narrow subset of high-reward patterns (Wang et al., 2024a; GX-Chen et al., 2025), motivating recent studies of alternative divergences, including forward-KL-style exploration in RAPO (Deng et al., 2025) and f-divergence-based GRPO variants (Li et al., 2025a). Different from existing RLVR works that mainly use alternative divergences to preserve diversity or enhance exploration (pass@k), our work targets late-stage **over-optimization**, where training reward keeps increasing while validation accuracy collapses, and addresses it through Anchor Rollout and Mixed Optimization.

B Implementation Details

We implement our method based on the open-source DAPO recipe. Below we detail the specific

Table 3: Supplementary comparison with teacher-guided two-stage training. OPD and MOPD use the same DAPO-trained checkpoint as a teacher policy, while ARMOR continues training from this checkpoint to address over-optimization. All checkpoints are selected by AIME24 performance, and math results report avg@32.

Model	Method	Math Reasoning Tasks				General Tasks	
		AIME24	AIME25	AMC	Average	GPQA	MMLU-Pro
Qwen2.5-Math-7B	DAPO (teacher)	37.13	15.21	69.39	40.58	38.26	43.93
	OPD	38.12	15.21	68.59	40.64	39.39	43.34
	MOPD	38.85	15.93	68.56	41.11	38.76	43.81
	ARMOR	**43.04**	**18.13**	**76.13**	**45.77**	**42.49**	**46.03**
Qwen3-8B-Base	DAPO (teacher)	36.98	28.13	71.72	45.61	49.87	65.55
	OPD	38.23	27.81	71.16	45.73	52.27	65.91
	MOPD	37.80	29.38	72.03	46.40	50.51	65.99
	ARMOR	**48.13**	**34.48**	**80.35**	**54.32**	**56.25**	**68.90**

configurations for reproducibility (we will also release the code upon acceptance).

Training Hyperparameters. We use a learning rate of 1e-6 with a 10-step warmup. For the DAPO objective, we adopt the default dual-clip ratios of $\epsilon_{\text{low}} = 0.2$ and $\epsilon_{\text{high}} = 0.28$. The training process is configured with a global batch size of $B = 512$ queries and a response group size of $|G| = 16$. Optimization is performed with a mini-batch size of 32 prompts, resulting in $\mu = 16$ gradient updates per RL step. For QAE, we set the baseline quantile to 45%.

ARMOR Configurations. For **Anchor Rollout**, we set a sampling budget of $|\mathcal{Y}_{\text{ref}|x}| = 4$ per query for the reference policy to conduct rejection sampling, ensuring strict correctness ($R = 1$). Consequently, given the original group size of $|G| = 16$, the total inference cost is estimated at approximately $20/16 = 125\%$ of the on-policy baseline. The mixing coefficient for **Mixed Optimization** is set to $\alpha = 0.9375$ for Qwen2.5-Math-7B and $\alpha = 0.875$ for Qwen3-8B-Base.

Model-Specific Settings. To accommodate different context window capabilities, the overlong penalty thresholds (and max generation lengths) are set to 4k (8k) for Qwen2.5-Math-7B and 16k (20k) for Qwen3-8B-Base, leveraging the latter's long-context pretraining (Yang et al., 2025). Regarding the **reference reset strategy**, we apply distinct schedules to produce the results in Tab. 1, for which we provide a detailed analysis in the following section:

- **Qwen2.5-Math-7B:** A fixed reset frequency of $\tau = 160$ for both DAPO and QAE.
- **Qwen3-8B-Base:** A two-stage strategy, starting with $\tau = 1$ for the first 180 steps, followed by a fixed reference policy.

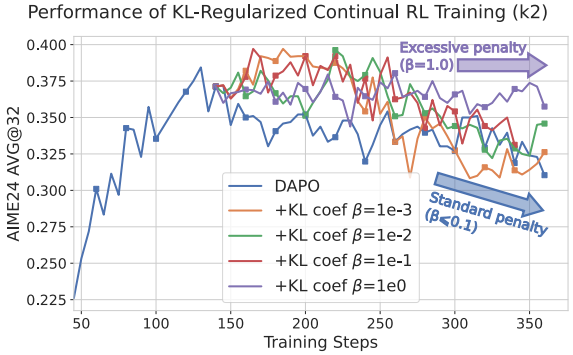


Figure 5: **KL-regularized training performance (using k2 estimation).** Consistent with k3 results (Fig. 2a), standard penalties ($\beta \leq 0.1$: 1e-3, 1e-2, 1e-1) merely delay validation collapse. The extreme setting ($\beta = 1.0$) enforces stability but induces stagnation, validating the stability-plasticity dilemma in KL-regularization.

C Additional Experiment Results

C.1 KL Regularization with k2 Estimation

In Section 3, we analyzed continual training using the standard k3 estimator for reverse KL. To ensure our conclusions are robust to estimator choice, we extend this analysis using the k2 estimator, which is shown to provide unbiased gradient estimation (Tang and Munos, 2025). As illustrated in Fig. 5, the performance dynamics mirror those observed with k3: standard penalties ($\beta \leq 0.1$) fail to prevent eventual collapse, whereas excessive penalties ($\beta = 1.0$) secure stability only at the cost of stagnation. This confirms that the stability-plasticity dilemma is intrinsic to the KL objective itself,

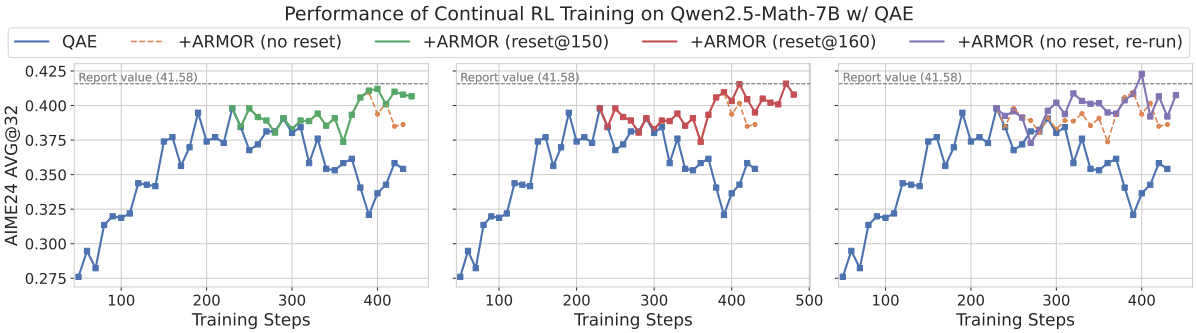


Figure 6: Hyperparameter robustness of ARMOR on QAE. We compare ARMOR variants against the QAE baseline. Across all settings—whether using static anchors, periodic resets after 150 steps, 160 steps, or a distinct re-run—ARMOR consistently stabilizes training and maintains high performance. Notably, all variants approach or surpass the reported peak performance (AIME24 `avg@32=41.58`, indicated by the gray dashed line).

rather than an artifact of the estimation method.

C.2 Comparison with Teacher-Guided Two-Stage Training

Beyond directly continuing RL from the best zero-RL checkpoint, another natural two-stage strategy is to use this checkpoint as a teacher policy for on-policy distillation. We therefore compare ARMOR with OPD (Agarwal et al., 2024) and MOPD (Xiao et al., 2026) under the same DAPO-trained reference policy. As shown in Tab. 3, OPD and MOPD provide stable teacher-guided training but only marginally improve over the reference checkpoint. In contrast, ARMOR achieves substantially larger gains across both base models, indicating that explicitly targeting over-optimization is more effective than simply distilling from the zero-RL model.

C.3 Hyperparameter Robustness

While the main body of our paper focuses on ablations using DAPO + Qwen2.5-Math-7B, we further evaluate the robustness of ARMOR's hyperparameters across different algorithms and base models.

Firstly, we present a detailed comparison using the **QAE + Qwen2.5-Math-7B** setting. Starting from the best QAE checkpoint (Step 230), we assess four variants of ARMOR to analyze the impact of reset frequency and training stochasticity. As illustrated in Fig. 6, we compare:

1. ARMOR (no reset): We freeze the reference policy π_{ref} throughout the training phase.

2. ARMOR (reset@150): The reference policy is updated ($\pi_{\text{ref}} \leftarrow \pi_\theta$) every 150 steps during continual training.

3. ARMOR (reset@160): The reference policy is updated every 160 steps.

4. ARMOR (no reset, re-run): This is a reproducibility check on no-reset. To verify stability against the inherent randomness of RL training (e.g., rollout randomness), we conduct an independent re-run of the static anchor setting.

The results in Fig. 6 demonstrate strong algorithmic robustness. While the baseline QAE suffers from performance degradation, all ARMOR variants successfully sustain the performance and approach or surpass the reported AIME24 value in Tab. 1, demonstrating that the gains are robust.

Next, we present a comparison using the **DAPO + Qwen3-8B-Base** setting.

Unlike the Qwen2.5-Math-7B model, Qwen3 is a strong base model with high intrinsic reasoning potential. We hypothesize that in the continual training setting, such a model benefits from a more aggressive exploration space in the early stages, without being strictly tethered to a static reference. To validate this, we compare the following strategies (Fig. 7):

1. ARMOR (reset@1) We reset the reference policy at every step. This maximizes the adaptivity of the trust region but minimizes the "anchoring" effect.

2. ARMOR (reset@1 til 110) and ARMOR (reset@1 til 180): We adopt a two-stage schedule: updating the reference at every step ($\tau = 1$) for the first T steps (110 or 180) to facilitate rapid adaptation, then freezing the reference policy to secure stability.

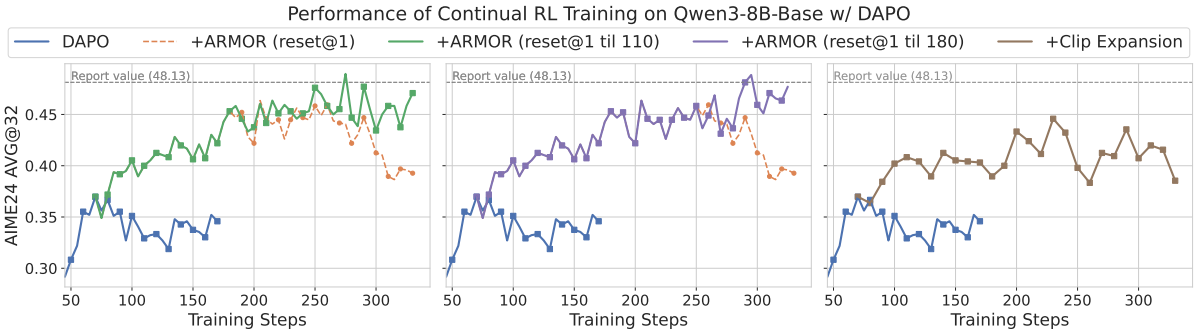


Figure 7: Ablation of Reset Strategy on Qwen3-8B-Base. Due to Qwen3's strong capability, continuous reference updates (reset@1, *i.e.*, $\tau = 1$) enable rapid early gains but suffer from late-stage instability (orange dashed line). Our Hybrid Strategy (reset@1 til 110 and reset@1 til 180) resolves this by freezing the anchor after an initial adaptation phase, both reaching the reported peak performance (AIME24 `avg@32=48.13`). In contrast, the Static Clip Expansion baseline yields lower and more unstable performance, confirming that ARMOR's benefit stems from structure-aware regularization rather than simple clip relaxation.

3. Clip Expansion: To verify that ARMOR's gain is not merely due to a wider clip range, we test a baseline using standard DAPO with a manually expanded clip range matching ARMOR's theoretical boundary (as detailed in Sec. 5.2.2).

Analysis. The results reveal three key insights: *(1) Necessity of Anchors*: Continuous updates ($\tau = 1$) accelerate early adaptation but eventually degrade due to the loss of anchor stabilization, confirming the need for a fixed reference in the late stage. *(2) Effectiveness and Robustness*: The hybrid strategy secures the best of both worlds: rapid early exploration and sustained stability. Notably, both freeze timings (180/250 steps) reach the reported ceiling, demonstrating ARMOR's robustness to the specific schedule. *(3) Superiority over Simple Relaxation*: The instability of Static Clip Expansion confirms that ARMOR's benefit derives from principled, direction-aware regularization rather than a generic relaxation of constraints.

D Potential Risks

This paper is mainly concerned with algorithmic advances in reinforcement learning for large language models. The proposed method does not introduce new application scenarios, datasets, deployment settings, or model capabilities that would create risks beyond those generally associated with LLM post-training. Therefore, we do not foresee specific potential risks unique to this work, while acknowledging that general LLM-related risks such as misuse, hallucination, and biased generation remain relevant.

E Artifacts and Licenses

The artifacts used in this work, including code, datasets, and model checkpoints, are listed below together with their corresponding licenses. We use these artifacts in accordance with their intended research use and license terms.

- DAPO's training recipe, including the released code, training scripts, and training data: Apache 2.0 License.
- AIME datasets: Apache 2.0 License. These datasets are used only for evaluation.
- AMC dataset: Apache 2.0 License. The dataset is used only for evaluation.
- GPQA dataset: Creative Commons Attribution 4.0 International (CC BY 4.0) License.
- MMLU-Pro dataset: MIT License.
- Qwen-2.5-Math model: Apache 2.0 License.
- Qwen-3-8B-Base model: Apache 2.0 License.

We do not introduce new datasets containing private, sensitive, or personally identifiable information. All external artifacts are publicly available, and our use of them is limited to algorithm development and benchmark evaluation.

F The use of LLMs

We utilize LLMs only to polish some of the language of this paper. All content was originally drafted by the authors. The use of LLMs was restricted to refining some pre-existing text, and any suggested modifications were reviewed by the authors to confirm their accuracy and alignment with the original meaning.