Turning Failures into Value: Negative Experience Replay for RLVR via Confidence Gating and Boundary Failure Sampling

Jialiang Guo[1*], Fucheng Xiong[1*], Xu He[1], Haodong Zhao[1†], Xingyang Li[1†], Ke Zeng[1], Xunliang Cai[1],
[1]Meituan,
Correspondence: {guojialiang04, xiongfucheng, zhaohaodong04, lixingyang04}@meituan.com

Abstract

Reinforcement Learning with Verifiable Rewards (RLVR) has become the standard paradigm for enhancing reasoning capabilities in Large Language Models, yet on-policy algorithms like GRPO suffer from sample inefficiency. Current experience replay methods for RLVR typically replay correct trajectories to consolidate learned reasoning patterns and accelerate convergence, but overlook the vast failure space. This work investigates how to effectively replay failure trajectories. We find that the high heterogeneity of failures renders random replay ineffective, and that high-value negatives should be both gradient-efficient and structurally proximal to correct solutions. To this end, we propose NexGRPO, which employs mid-confidence gating to filter invalid noise and saturated errors, and utilizes boundary failure sampling to retrieve boundary errors semantically similar to correct solutions for targeted refinement. Extensive experiments on mathematical and general reasoning benchmarks demonstrate that NexGRPO outperforms strong baselines and achieves improved out-of-distribution generalization.

1 Introduction

Reinforcement Learning with Verifiable Rewards (RLVR) has emerged as the dominant post-training paradigm for reasoning tasks like mathematics and coding (Achiam et al., 2023; Lightman et al., 2023; Yang et al., 2024; Guo et al., 2025). Within this framework, Group Relative Policy Optimization (GRPO) (Shao et al., 2024; Xiong et al., 2025; Yue et al., 2025) stands out as an efficient, critic-free baseline. However, GRPO's on-policy nature leads to significant sample inefficiency: expensive reasoning trajectories—often involving lengthy chains of thought—are discarded immediately after a single gradient update. (Jaech et al., 2024; Kimi Team, 2025) This disposable data utilization incurs computational costs and prevents the model from taking advantage of historical interactions.

To mitigate this inefficiency, Experience Replay (ER) (Lin, 1992; Mnih et al., 2015; Schaul et al., 2015) has been adapted from classic reinforcement learning to LLM training, aiming to enhance sample utilization by resampling historical data. Recent integrations of ER into RLVR predominantly focus on reusing verified successful trajectories. Zhang et al. (2025) adopts a two-stage strategy that collects correct trajectories for mixed training, while Zhan et al. (2025) further incorporates entropy- or difficulty-based filtering to prioritize high-quality positive samples. The rationale is to prevent the forgetting of mastered reasoning patterns and accelerate convergence by reusing correct solutions. However, such strategies concentrate solely on replaying correct trajectories, neglecting valuable failure trajectories.

Complex reasoning often produces many incorrect attempts per correct solution. Although negative updates on errors can improve accuracy and suppress hallucinations while preserving diversity Wang and Li (2023); Zhu et al. (2025), leveraging failures via replay is challenging: the failure space is vast and highly heterogeneous, so random replay is dominated by low-quality trajectories and yields high-variance, low-relevance updates. RLEP similarly finds that naively adding wrong answers to the buffer brings no gains and can hurt stability (Zhang et al., 2025; Xu et al., 2025), making effective signal extraction from noisy failures the main obstacle.

Furthermore, existing strategies prioritizing low-entropy or high-confidence trajectories ensure stability in early training, persistent reuse of such samples in later stages exacerbates policy self-reinforcement, constraining the model to a narrow set of reasoning patterns. As noted by Cui et al. (2025), a singular pursuit of minimal en-

[*] Co-first authors.
[†] Corresponding authors.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 36316–36334

tropy can induce entropy collapse, suppressing the exploration of high-entropy but correct long-tail solutions. Above all, the critical missing link is an interpretable and realizable value criterion that transforms massive, unusable failure sets into stable training signals.

We study which failure experiences are worth replaying in RLVR. Our premise is that valuable failures are both gradient-efficient and semantically close to positive samples. We use model confidence to filter out low-probability noise and overconfident errors with saturated gradients, keeping mid-confidence failures with strong update signals. We also retrieve Boundary Failures, which follow the same reasoning path as correct solutions but deviate at key points. Correcting them sharpens the decision boundary near the target solution instead of wasting updates on irrelevant regions.

Accordingly, we propose the **NexGRPO** framework (Negative experience Replay Group Relative Policy Optimization), which incorporates two core mechanisms: Mid-Confidence Gating, acting as a filter to admit only failures within the gradient-sensitive interval into the pool; and Boundary Failure Sampling, which employs semantic representations to retrieve negative samples most entangled with the current correct path, constructing mixed training groups with explicit error-correction directionality. Furthermore, we integrate these high-value negative samples into the online training loop via a mixed policy optimization objective.

Our main contributions are summarized as follows:

- We investigate experience replay in RLVR and argue that high-value negative experiences must possess gradient efficiency and semantic structural relevance to positive samples.

- We propose NexGRPO, a framework that employs Mid-Confidence Gating to filter for effective update intensity and Boundary Failure Sampling to target semantic near-misses, ensuring precise boundary refinement.

- Extensive experiments demonstrate that NexGRPO significantly outperforms strong baselines in sample efficiency and out-of-distribution generalization across mathematical and general reasoning benchmarks.

2 Related Work

RLVR for LLMs Reasoning GRPO achieves stable, critic-free policy optimization via group-wise advantage estimation (Shao et al., 2024; DeepSeek-AI et al., 2025). To further enhance its performance, recent works have proposed various refinements. Regarding training stability, Dr.GRPO and DAPO improve importance sampling ratio estimation and clipping mechanisms (Liu et al., 2025; Yu et al., 2025), while GSPO and CISPO optimize advantage objectives to reduce update variance (Zheng et al., 2025; Chen et al., 2025). Regarding long-chain reasoning efficiency, SPO, TEMPO, and GiGPO introduce fine-grained step-level or token-level credit assignment to address long-range dependencies under sparse rewards (Xie et al., 2024; Cao et al., 2023; Feng et al., 2025a). These methods lack trajectory reuse, constraining sample efficiency (Li et al., 2025).

Experience Replay in RLVR To enhance sample efficiency, experience replay mechanisms have been progressively integrated into RLVR. RLEP (Zhang et al., 2025) prioritizes replaying validator-confirmed correct trajectories to consolidate model capabilities. Building on this, Ex-GRPO (Zhan et al., 2025) incorporates heuristic filtering based on entropy or difficulty, integrating high-quality historical experiences into the training loop via a mixed optimization objective. Additionally, RePO (Li et al., 2025) and EFRame (Wang et al., 2025) explore broader offline or asynchronous reuse strategies, maintaining experience buffers to mix historical data with online samples (Bartoldson et al., 2025). These strategies focus primarily on consolidating successful experiences, treating the vast volume of failure trajectories as invalid noise.

Learning from Negative Signals Mechanistic studies suggest that negative updates on error samples are crucial for defining decision boundaries (Andrychowicz et al., 2017; Rafailov et al., 2023). Zhu et al. (2025) decomposes learning signals into positive and negative updates, demonstrating that negative reinforcement alone significantly enhances generation diversity and inference scaling. To leverage these signals, Feng et al. (2025b) introduces a confidence-aware reweighting mechanism to extract effective gradients from error samples. Existing research lacks systematic mechanisms for screening and organizing heterogeneous failure samples within an experience replay framework.

3 Preliminaries

Verifiable Reward Function In the RLVR setting, the reward is determined by a deterministic verifier that extracts the final answer from the model output and compares it against a ground truth (Lyu et al., 2025; Wen et al., 2025). Formally, given a query q and a model output trajectory o (comprising the reasoning chain and the answer), the reward is defined as:

$$r(q, o) = \mathbb{I}(verify(o, y_q))$$

where y_q denotes the ground truth and $\mathbb{I}(\cdot)$is the indicator function.

Group Relative Policy Optimization (GRPO) GRPO obviates the need for a critic model by estimating advantages through group-wise reward normalization. For a query q, a group of trajectories $\{o_i\}_{i=1}^{G}$ is sampled from the reference policy $\pi_{\theta_{\text{old}}}$. We compute the group mean μ_q and standard deviation σ_q of the rewards to derive the relative advantage $\hat{A}_i = \frac{r_i - \mu_q}{\sigma_q + \epsilon}$. The policy is then optimized via a token-level clipped surrogate objective:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(O|q)] \left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{L_i}\sum_{t=1}^{L_i}\min\left(\rho_{i,t}(\theta)\hat{A}_i, \text{clip}(\rho_{i,t}(\theta))\hat{A}_i\right)\right] \quad (1)$$

where $\rho_{i,t}(\theta)$ is the probability ratio between the current and old policies, and $\text{clip}(\rho)$ denotes $\text{clip}(\rho, 1-\epsilon, 1+\epsilon)$. This mechanism reinforces trajectories that outperform the group average while suppressing inferior ones. In this work, when introducing experience replay into GRPO, we retain this core advantage estimation structure, extending and modifying only the sample sources and the policy used for the ratio denominator.

4 Methodology

This section proposes a failure experience replay method tailored for RLVR, aiming to address three core questions: (1) Which negative samples are worth replaying; (2) how to manage and sample these experiences; and (3) how to integrate replayed experiences into policy optimization to enhance sample efficiency while maintaining reasoning diversity.

4.1 Which Negative Samples to Replay

To mitigate the noise caused by failure heterogeneity in RLVR, we propose filtering experiences based on two criteria: **Mid-Confidence Failures** to ensure effective gradient magnitude (Update Magnitude), and **Boundary Failures** to align corrections with successful patterns (Update Direction).

4.1.1 Mid-Confidence Failures

Gradient Characteristic Analysis. We analyze the GRPO gradient to determine which failure samples provide effective optimization signals. For an incorrect trajectory, taking the derivative of Equation (1) with respect to the logit z_v yields:

$$\frac{\partial \mathcal{J}_{\text{GRPO}}}{\partial z_v} \propto -\pi_{o_t}\big(\mathbb{I}(v = o_t) - \pi_v\big). \quad (2)$$

This gradient suppresses the sampled token o_t and redistributes probability mass to other candidates according to π_v. The update strength on the target token is governed by $\pi_v(1 - \pi_v)$: low-confidence failures contribute negligible gradients due to tiny probability mass, while high-confidence failures saturate and yield vanishing gradients. Therefore, only mid-confidence failures provide sufficiently strong and stable gradient signals.

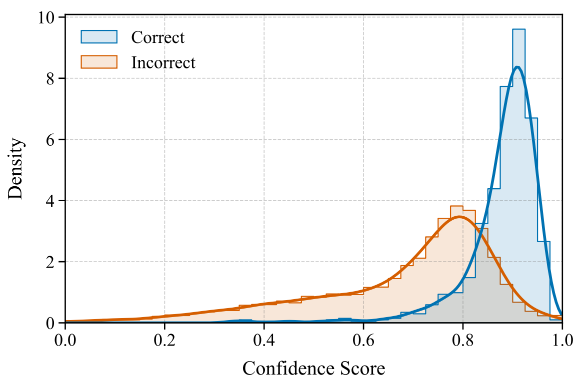


Figure 1: Confidence distributions of correct and incorrect trajectories, where the incorrect distribution exhibits a distinct heavy-tailed characteristic.

Confidence-Based Filtering Guided by the token-level analysis, we define trajectory confidence as the sequence geometric mean probability (Kadavath et al., 2022):

$$\ell(o) = \exp\left(\frac{1}{|o|}\sum_{t=1}^{|o|}\log \pi_\theta(o_t \mid q, o_{<t})\right). \quad (3)$$

We empirically analyze the confidence distribution of Qwen2.5-Math (Yang et al., 2024) on OlympiadBench (He et al., 2024). As shown in Figure 1, failure trajectories exhibit a heavy-tailed distribution. Uniform sampling often over-selects either low-confidence noise or high-confidence saturated errors. We therefore define trajectories with $\ell(o) \in [\ell_{\text{low}}, \ell_{\text{high}}]$ as **Mid-Confidence Failures**, filtering extreme cases and retaining failures with the strongest gradient contribution.

4.1.2 Boundary Failures

Building on mid-confidence filtering for sufficient gradient magnitude, we introduce **Boundary Failures** to reduce directional noise from heterogeneous errors. Boundary Failures are negative trajectories that follow the same reasoning path as successful samples, but deviate at a few critical points. They focus gradient updates on the key divergences that separate correct logic from subtle mistakes.

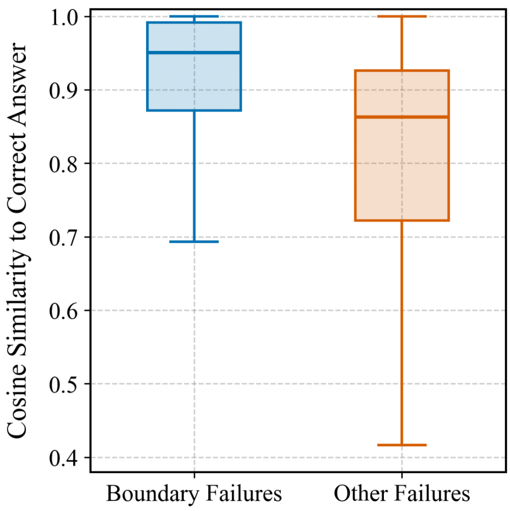


Figure 2: Quantitative analysis of error representations: Box plot of cosine similarity.

To accurately quantify structural proximity, we employ the Model-Internal Representation. Formally, given a trajectory o of length T generated by a model with L layers, we define the representation vector $\phi(o)$ as the mean-pooled hidden states from the last decoder layer:

$$\phi(o) \triangleq \frac{1}{T} \sum_{t=1}^{T} \mathbf{h}_L^{(t)} \in \mathbb{R}^d \quad (4)$$

where d denotes the hidden dimension. We adopt mean pooling because our objective is to retrieve structurally similar failures—trajectories that follow analogous reasoning paths but diverge at critical points. By aggregating hidden states across the entire sequence, mean pooling provides a holistic representation of the reasoning structure, effectively capturing the overall solution approach rather than any single localized decision.

To verify the validity of this representation, we analyzed typical problems involving diverse solution paths. The box plot in Figure 2 shows that the cosine similarity between manually annotated boundary failures and successful samples is higher than that of other error types. Based on this observation, we formulate the selection of boundary failures as a nearest neighbor search problem using cosine distance in the representation space.

4.2 Experience Management

Based on this analysis, we design an experience management mechanism that integrates Mid-Confidence Gating and Boundary Failure Sampling to construct high-value replay data.

Experience Pool Structure We maintain a query-indexed experience replay buffer $\mathcal{E}$ utilizing a dual-queue structure. For each query q, we maintain a mapping $q \mapsto (\mathcal{P}_q, \mathcal{N}_q)$, where $\mathcal{P}_q$ is the positive queue storing historical successful trajectories as positive anchors, and $\mathcal{N}_q$ is the candidate negative queue storing failure trajectories with high training potential. Each trajectory entry τ stored in the pool comprises a triplet:

$$\tau = \langle o, \log \pi_{\text{old}}, \ell(o) \rangle \quad (5)$$

where o denotes the generated trajectory text; $\log \pi_{\text{old}}$ is the sequence of token-level log probabilities from the behavioral policy at the time of generation, used for calculating off-policy importance sampling ratios; and $\ell(o)$ is the geometric mean probability of the sequence, serving as a scalar metric for sample confidence.

Experience Collection As illustrated in Figure 3(a), during training, for each query q, we collect K trajectories $\mathcal{G}_q = \{o_1, \ldots, o_K\}$ generated by the policy $\pi_{\theta_{\text{old}}}$. For each trajectory, we calculate its verifiable reward $r(q, o)$ and sequence confidence $\ell(o)$. Correct trajectories are treated as positive anchors and pushed directly into $\mathcal{P}_q$ to maintain memory of correct patterns for subsequent optimization; for incorrect trajectories, we further inspect whether their confidence falls within the preset effective interval $[\ell_{\text{low}}, \ell_{\text{high}}]$, admitting only qualifying mid-confidence failures into $\mathcal{N}_q$. This

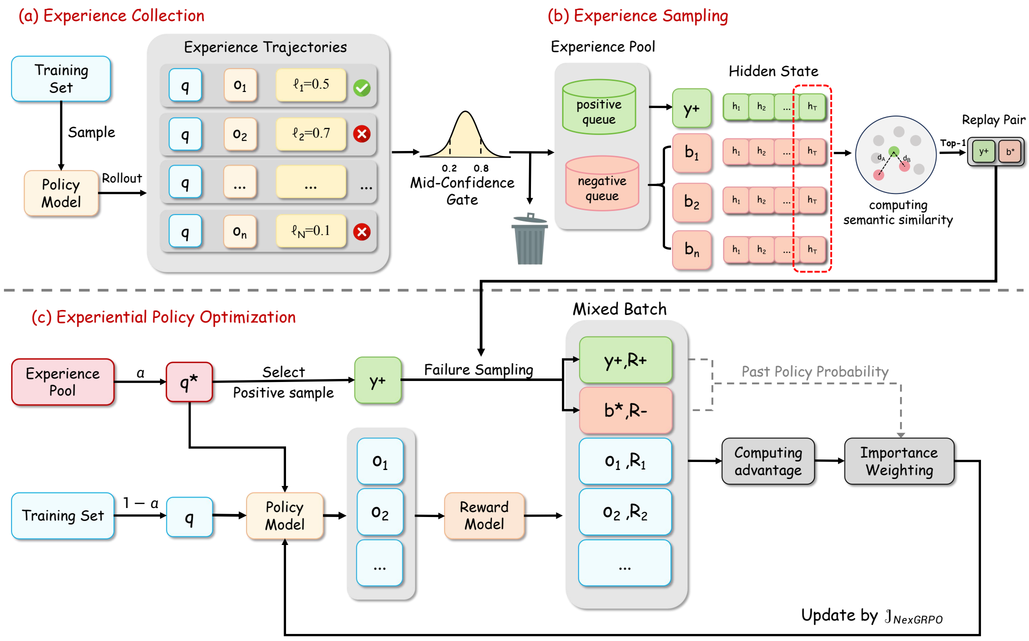


Figure 3: Overview of NexGRPO. (a) Experience Collection: High-value negative samples are filtered via Mid-Confidence Gating. (b) Experience Sampling: Positive anchors y^+ are paired with semantically similar boundary failures b^*. (c) Experiential Policy Optimization: Update policy with mixed replay and online data.

gating filters out low-probability noise and over-confident stubborn errors, ensuring that retained negatives reside in the gradient-sensitive region.

Experience Sampling For experience replay on query q, we perform structured Boundary Failure Sampling, as illustrated in Figure 3(b). We randomly sample a successful trajectory y^+ from the positive queue $\mathcal{P}_q$ to serve as the positive anchor. Subsequently, we retrieve the boundary failure from $\mathcal{N}_q$ that is most proximal to the anchor's reasoning logic. Utilizing the representation ϕ defined in Section 4.1.2, we employ cosine similarity to measure directional consistency and select the trajectory with the highest similarity as the boundary failure b^*:

$$b^* = \operatorname*{argmax}_{b \in \mathcal{N}_q} \frac{\phi(y^+)^\top \phi(b)}{\|\phi(y^+)\|_2 \|\phi(b)\|_2} \tag{6}$$

Through this procedure, the pair (y^+, b^*) is assembled for replay, locking onto hard negatives that are structurally isomorphic to the correct solution in the representation space.

4.3 Experiential Policy Optimization

NexGRPO integrates filtered high-value experiences directly into the GRPO online update loop via a mixed data flow. The complete training procedure is illustrated in Figure 3(c), and the detailed algorithm is presented in Algorithm 1.

Mixed Batch Construction In each optimization step, we construct a mixed batch $\mathcal{B}$ fusing on-policy samples $\mathcal{B}_{\text{on}}$ from dataset $\mathcal{D}$ and replay samples $\mathcal{B}_{\text{exp}}$ from the experience pool $\mathcal{E}$. The former is sourced from the training dataset, while the latter is sampled from the experience pool via difficulty-based Gaussian sampling. We introduce a hyperparameter $\alpha \in [0, 1)$ to control the replay ratio, set as $|\mathcal{B}_{\text{exp}}| = \min(\lfloor \alpha B \rfloor, |\mathcal{E}|)$, with the remainder filled by on-policy data. Inspired by ExGRPO, we employ a delayed start mechanism: experience replay is activated only when the batch Pass@1 metric exceeds a predefined threshold.

Group Composition and Objective For each replay query q^* in $\mathcal{B}_{\text{exp}}$, we construct a mixed group $\mathcal{G}_{q^*} = \{y^+, b^*, o_1, \ldots, o_{G-2}\}$ containing G trajectories. Here, y^+ is the replayed positive anchor, b^* is the replayed boundary failure, and the remaining $\{o_i\}$ are supplementary rollouts generated by the current policy π_θ. This composition ensures the group contains both high-gradient historical samples and current exploration samples, providing an accurate baseline for estimating the group relative

Model	In-Distribution Performance							Out-of-Distribution Performance			
	AIME24	AIME25	AMC	MATH-500	Minerva	Olympiad	Avg.	ARC-c	GPQA*	MMLU-Pro	Avg.
Qwen2.5-7B-Instruct											
Base	12.0	7.1	43.6	71.0	28.3	39.6	33.6	86.0	6.6	55.8	49.5
GRPO	14.0	9.5	45.5	76.0	31.0	41.8	36.3	91.4	19.2	57.6	56.0
RLEP	15.2	10.9	47.8	78.6	34.2	43.1	38.5	91.5	19.7	58.4	56.5
ExGRPO	16.7	10.8	47.9	79.0	38.2	42.7	39.2	92.1	26.8	57.8	58.9
NexGRPO	**17.2**	**11.8**	**48.8**	**79.5**	**38.0**	**43.6**	**39.8**	**92.8**	**25.2**	**59.8**	**59.3**
Qwen2.5-Math-7B											
Base	10.2	3.6	29.2	45.6	12.1	14.1	19.2	17.3	12.1	17.5	15.6
GRPO	25.8	12.6	58.4	79.4	35.7	43.0	42.5	65.5	33.8	46.1	48.5
RLEP	23.5	13.1	59.3	80.4	37.9	44.0	43.0	74.4	38.4	47.3	53.4
ExGRPO	26.3	14.1	61.4	79.8	36.0	43.3	43.5	75.0	33.8	47.0	52.2
NexGRPO	**27.0**	**16.2**	**60.7**	**81.6**	**39.0**	**44.3**	**44.8**	**80.9**	**37.9**	**47.5**	**55.4**
Llama3.1-8B-Instruct											
Base	2.9	0.3	18.6	47.6	16.2	16.1	17.0	57.5	1.0	42.6	33.7
GRPO	5.8	1.0	20.2	46.6	23.9	16.0	18.9	85.2	0.0	48.0	44.4
RLEP	5.5	1.4	23.1	48.6	24.6	17.9	20.2	84.7	3.4	46.0	44.0
ExGRPO	5.0	0.6	23.6	50.0	23.9	19.4	20.4	81.0	7.2	48.9	45.7
NexGRPO	**5.5**	**1.8**	**24.0**	**49.0**	**23.5**	**21.6**	**20.9**	**82.7**	**4.1**	**51.7**	**46.1**

Table 1: Performance comparison of different methods on In-Distribution mathematical tasks and Out-of-Distribution general reasoning tasks. All values report Pass@1 accuracy (%), except for AIME, which reports Pass@32.

advantage $\hat{A}$. To correct for distribution shifts in off-policy data, we adopt an importance-weighted GRPO objective. The formulation follows Eq. (1), but modifies the definition of the probability ratio $\rho_{i,t}(\theta)$ to account for the sample source:

$$\rho_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t}|q, o_{i,<t})}{\pi_{\text{ref}}(o_{i,t}|q, o_{i,<t})} \tag{7}$$

where the reference policy π_{ref} is determined dynamically: $\pi_{\text{ref}} = \pi_{\theta_{\text{old}}}$ for on-policy trajectories, and $\pi_{\text{ref}} = \pi_{\text{beh}}$ (the behavioral policy at generation time) for replay trajectories.

Soft Retirement When a query achieves 100% correctness within the current rollout group, it is moved to a retirement set $\mathcal{S}$. When constructing $\mathcal{B}_{\text{on}}$, rather than discarding $\mathcal{S}$ entirely, we assign it a very low resampling probability ϵ. This design concentrates computational resources on unmastered boundary problems while retaining a minimal review frequency to maintain performance stability.

Dynamic Confidence Refresh Prior to sampling b^*, we re-evaluate its confidence $\ell_\theta(b^*)$ under the current policy (Du et al., 2022). Candidates drifting outside $[\ell_{\text{low}}, \ell_{\text{high}}]$ are discarded as degenerated noise or saturated errors, ensuring that only gradient-sensitive samples participate in the update.

5 Experiments

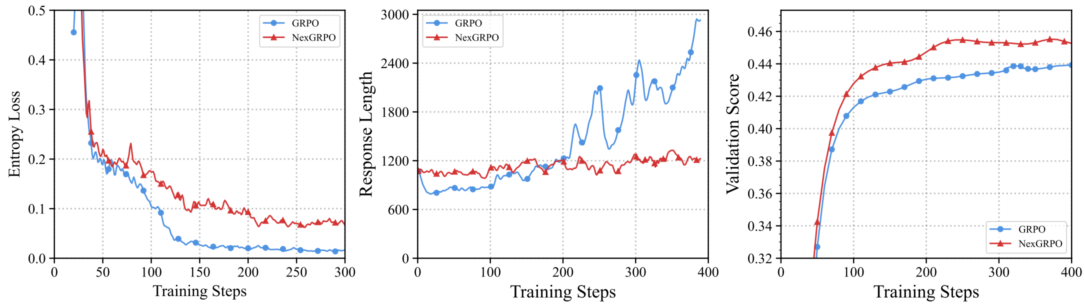
5.1 Experimental Setup

Datasets & Benchmarks We evaluate our method on 9 representative reasoning benchmarks. We selected 6 in-distribution mathematical reasoning tasks, including AIME 2024, AIME 2025, AMC (Li et al., 2024), MATH-500 (Hendrycks et al., 2021), Minerva Math (Lewkowycz et al., 2022), and OlympiadBench (He et al., 2024). For Out-of-Distribution generalization, we select ARC-Challenge (Clark et al., 2018), GPQA-Diamond (Rein et al., 2024), and MMLU-Pro (Wang et al., 2024) for evaluation. All evaluations are conducted with a sampling temperature of $T = 0.6$ and top-$p = 1.0$, with answers extracted using the Oat-evaluator toolkit (Liu et al., 2025).

Training Settings We sampled 10k instances from OpenR1-Math (Hugging Face, 2025) to train Qwen (Team et al., 2024) and Llama (Dubey et al., 2024) models for 5 epochs with a batch size of 128. On-policy groups consist of $G = 8$ sampled responses updated via GRPO. For replay queries, we use NexGRPO's mixed group construction (1 positive anchor + 1 boundary failure) with an experience ratio $\alpha = 0.5$. Confidence thresholds are set to $[\ell_{\text{low}}, \ell_{\text{high}}] = [0.2, 0.9]$, and soft retirement probability to $\epsilon = 0.05$. Following ExGRPO, replay is activated only after Pass@1 exceeds 0.35. Further details are provided in Appendix B.

Method	In-Distribution Performance							Out-of-Distribution Performance			
	AIME24	AIME25	AMC	MATH-500	Minerva	Olympiad	Avg.	ARC-c	GPQA*	MMLU-Pro	Avg.
NexGRPO	**27.0**	**16.2**	**60.7**	**81.6**	**39.0**	**44.3**	**44.8**	**80.9**	**37.9**	**47.5**	**55.4**
w/o Conf.	25.8	13.8	59.5	80.5	37.4	43.6	43.4	77.5	35.5	46.8	53.3
w/o Bound.	24.5	12.9	58.8	79.9	36.2	43.1	42.6	75.8	34.2	46.4	52.1
w/o Retire.	26.7	16.5	59.8	80.2	38.2	44.0	44.2	80.8	37.8	47.3	55.3
w/o Conf. Refr.	26.0	15.2	60.0	80.8	38.0	42.8	43.8	79.5	37.0	46.7	54.4

Table 2: Ablation study on Qwen2.5-Math-7B. We evaluate the core components: Confidence Gating (Conf.), Boundary Alignment (Bound.), Soft Retirement (Retire.), and Confidence Refresh (Conf. Refr.).



(a) Entropy Loss (b) Response Length (c) Validation Score

Figure 4: Comparison of training dynamics. (a) Entropy Loss: NexGRPO maintains higher entropy for sustained exploration. (b) Response Length: Output length remains stable, avoiding reward hacking. (c) Validation Score: NexGRPO demonstrates superior convergence speed and final accuracy.

Baselines We compare NexGRPO against three categories of representative baselines: (1) **GRPO**: The standard on-policy RLVR baseline; (2) **RLEP**: An experience replay paradigm focusing on replaying successful trajectories; and (3) **ExGRPO**: A replay strategy that introduces experience management and entropy/quality-based heuristic filtering on top of successful trajectory replay.

5.2 Main Results

As shown in Table 1, **NexGRPO** achieves consistent superior performance across all three base models compared to the GRPO baseline and experience replay methods. On in-distribution mathematical tasks, NexGRPO demonstrates stable improvements. Taking Qwen2.5-Math-7B as an example, our method raises the average Pass@1 accuracy to 44.8%, surpassing GRPO by 2.3%. This improvement indicates that introducing filtered negative experiences provides more optimization signals than replaying positive samples alone.

Furthermore, NexGRPO exhibits considerable advantages in out-of-distribution generalization tasks. While baseline methods struggle to transfer reasoning capabilities to general domains, NexGRPO achieves notable gains on the challenging ARC-Challenge and GPQA benchmarks. Notably, on the ARC-c dataset with Qwen2.5-Math-7B, our method attains a score of 80.9%, outperforming GRPO (65.5%) and ExGRPO (75.0%). This result suggests that learning from boundary failures and mid-confidence errors allows the model to establish clearer decision boundaries, thereby reducing overfitting to specific dataset patterns and enhancing robustness in unseen reasoning scenarios.

Llama models exhibit certain capability limitations on high-difficulty benchmarks. We attribute this to their intrinsic reasoning constraints. During early RL stages, Llama models are prone to entropy collapse (Cheng et al., 2026) and the loss of long-horizon cognitive behaviors (Gandhi et al., 2025). At this stage, the models' capabilities are relatively weak, and introducing negative updates is more likely to induce gradient noise. This suggests that they require preliminary supervised fine-tuning prior to RL optimization.

5.3 Ablation Study

Impact of Experience Selection Strategies Table 2 presents the ablation results on Qwen2.5-Math-7B. The removal of Boundary Failure Sampling causes the most severe performance drop, reducing the average in-distribution accuracy from 44.8% to 42.6%. This result strongly supports our core view that naively replaying negative samples is insufficient; sampling relatively similar samples is required. Similarly, removing Mid-Confidence Gating leads to a 1.4% decline in average accuracy, confirming that filtering out invalid noise facilitates effective optimization signals.

Impact of Experience Management Mechanisms Removing Dynamic Confidence Refresh results in a 1.0% decrease in average accuracy. This indicates that as the policy evolves, historical failure samples may drift out of the gradient-sensitive region. Removing the Soft Retirement mechanism leads to a slight performance decline. This mechanism helps concentrate computational resources on unmastered problems.

5.4 Analysis and Discussion

In this section, we provide a detailed analysis of the internal mechanisms of NexGRPO, elucidating the pivotal role of our proposed components in enhancing model performance.

Filtered Negative Experiences enhances training stability and sample efficiency. As shown in Figure 4(a), NexGRPO maintains higher entropy levels, sustaining necessary exploration, which explains its superior generalization performance on out-of-distribution tasks. Figure 4(b) shows that NexGRPO avoids the length explosion phenomenon. We attribute this to Boundary Failure Sampling: by targetedly correcting semantically similar error paths, the model is compelled to learn precise decision boundaries rather than exploiting verbose generations for rewards. Finally, Figure 4(c) demonstrates high gains in sample efficiency, as NexGRPO matches the peak performance of GRPO by approximately step 200.

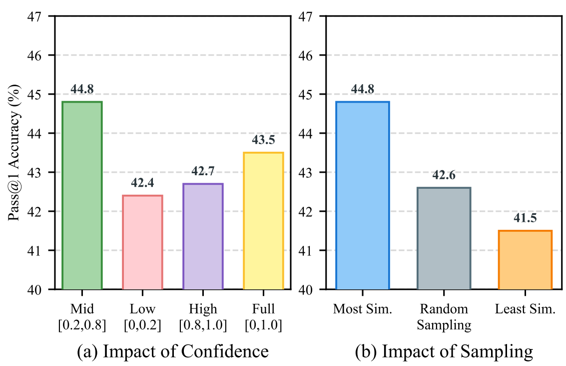


Figure 5: Ablation study on experience selection strategies. (a) Impact of confidence intervals. (b) Impact of semantic sampling strategies.

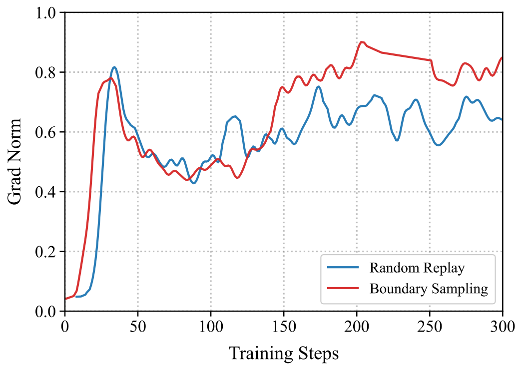


Figure 6: Comparison of Gradient Norm between Boundary Sampling and Random Replay.

Mid-Confidence Gating effectively filters out invalid and even detrimental negative samples. As shown in Figure 5(a), the Mid-Confidence strategy (0.2 to 0.8) significantly outperforms the Low-Confidence and High-Confidence settings. Our analysis reveals that low-confidence samples are often accompanied by chaotic logic or severe hallucinations, acting as noise that disrupts training stability. Conversely, high-confidence failures are frequently filled with repetitive loops, where the model artificially inflates confidence through redundant generation. By excluding these logical collapses and stubborn repetitions, Mid-Confidence Gating ensures that only negative samples possessing clear reasoning structures and effective gradient contributions are utilized.

Boundary Failure Sampling provides sustained and high-intensity optimization signals. As shown in Figure 5(b), Boundary Sampling (Most Sim.) outperforms random and least similar baselines. Figure 6 reveals the underlying mechanism: boundary failures typically diverge from correct solutions only at a few key discriminative points, concentrating gradients and maintaining higher consistency during updates, thus resulting in larger gradient norms. In contrast, failures in random replay are more heterogeneous and their biases more scattered, leading to diluted training signals and directional cancellation, which results in smaller gradient norms. As training progresses, boundary sampling continuously provides confusing near-miss errors to maintain intense optimization, whereas random negatives degenerate into weak signals due to their lack of challenge.

6 Conclusion

We study negative experience replay in RLVR, showing that useful failures must be gradient-efficient and structurally proximal to success. We propose NexGRPO, employing Mid-Confidence Gating to filter stochastic noise and saturated errors, and Boundary Failure Sampling to retrieve boundary failures for precise refinement. Across benchmarks and models, NexGRPO consistently outperforms strong baselines in both in-distribution and out-of-distribution tasks, with ablations confirming the mechanisms' complementarity.

Limitations

Our method assumes RLVR with deterministic verifiers; its effectiveness under preference-based rewards or partial-credit supervision is not evaluated. Boundary Failure Sampling requires a sufficient pool of correct trajectories as positive anchors and may be constrained in extremely low-accuracy regimes even with delayed replay. NexGRPO also introduces additional systems overhead, potentially requiring approximate indexing and more careful engineering to scale.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. 2017. Hindsight experience replay. *Advances in neural information processing systems*, 30.

Brian R. Bartoldson, Siddarth Venkatraman, James Diffenderfer, Moksh Jain, Tal Ben-Nun, Seanie Lee, Minsu Kim, Johan Obando-Ceron, Yoshua Bengio, and Bhavya Kailkhura. 2025. Trajectory balance with asynchrony: Decoupling exploration and learning for fast, scalable llm post-training. *Preprint*, arXiv:2503.18929.

Defu Cao, Furong Jia, Sercan O Arik, Tomas Pfister, Yixiang Zheng, Wen Ye, and Yan Liu. 2023. Tempo: Prompt-based generative pre-trained transformer for time series forecasting. *arXiv preprint arXiv:2310.04948*.

Aili Chen, Aonian Li, Bangwei Gong, Binyang Jiang, Bo Fei, Bo Yang, Boji Shan, Changqing Yu, Chao Wang, Cheng Zhu, and 1 others. 2025. Minimax-m1: Scaling test-time compute efficiently with lightning attention. *arXiv preprint arXiv:2506.13585*.

Daixuan Cheng, Shaohan Huang, Xuekai Zhu, Bo Dai, Xin Zhao, Zhenliang Zhang, and Furu Wei. 2026. Reasoning with exploration: An entropy perspective. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 30377–30385.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

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*.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Yunshu Du, Garrett Warnell, Assefaw Gebremedhin, Peter Stone, and Matthew E Taylor. 2022. Lucid dreaming for experience replay: refreshing past states with the current policy. *Neural Computing and Applications*, 34(3):1687–1712.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. 2025a. Group-in-group policy optimization for llm agent training. *arXiv preprint arXiv:2505.10978*.

Yunzhen Feng, Parag Jain, Anthony Hartshorn, Yaqi Duan, and Julia Kempe. 2025b. Don't waste mistakes: Leveraging negative rl-groups via confidence reweighting. *arXiv preprint arXiv:2510.08696*.

Kanishk Gandhi, Ayush Chakravarthy, Anikait Singh, Nathan Lile, and Noah D Goodman. 2025. Cognitive behaviors that enable self-improving reasoners, or, four habits of highly effective stars. *arXiv preprint arXiv:2503.01307*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, and 1 others. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850.

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. *arXiv preprint arXiv:2103.03874*.

Hugging Face. 2025. Open r1: A fully open reproduction of deepseek-r1.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, and 1 others. 2024. Openai o1 system card. *Preprint*, arXiv:2412.16720.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, and 1 others. 2022. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*.

Kimi Team. 2025. Kimi k1.5: Scaling reinforcement learning with llms. *Preprint*, arXiv:2501.12599.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, and 1 others. 2022. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, and 1 others. 2024. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9.

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

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Long-Ji Lin. 1992. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, 8(3):293–321.

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*.

Chengqi Lyu, Songyang Gao, Yuzhe Gu, Wenwei Zhang, Jianfei Gao, Kuikun Liu, Ziyi Wang, Shuaibin Li, Qian Zhao, Haian Huang, and 1 others. 2025. Exploring the limit of outcome reward for learning mathematical reasoning. *arXiv preprint arXiv:2502.06781*.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, and 1 others. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.

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*.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. 2015. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*.

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*.

Qwen Team and 1 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2(3).

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

Danqing Wang and Lei Li. 2023. Learning from mistakes via cooperative study assistant for large language models. *arXiv preprint arXiv:2305.13829*.

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

Xumeng Wen, Zihan Liu, Shun Zheng, Shengyu Ye, Zhirong Wu, Yang Wang, Zhijian Xu, Xiao Liang, Junjie Li, Ziming Miao, and 1 others. 2025. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base llms. *arXiv preprint arXiv:2506.14245*.

Zhengpeng Xie, Qiang Zhang, and Renjing Xu. 2024. Simple policy optimization. *arXiv preprint arXiv:2401.16025*.

Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang, Tong Zhang, Caiming Xiong, and Hanze Dong. 2025. A minimalist approach to llm reasoning: from rejection sampling to reinforce. *Preprint*, arXiv:2504.11343.

Yixuan Even Xu, Yash Savani, Fei Fang, and Zico Kolter. 2025. Not all rollouts are useful: Down-sampling rollouts in llm reinforcement learning. *Preprint*, arXiv:2504.13818.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, and 1 others. 2024. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

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, Shiji Song, and Gao Huang. 2025. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *Preprint*, arXiv:2504.13837.

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*.

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

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

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

A Gradient Derivation

In this section, we provide a rigorous derivation of the gradient of the GRPO objective function with respect to the unnormalized logits z. This derivation elucidates how the advantage signal propagates through the Softmax layer to update the model parameters.

A.1 Preliminaries

Let $z \in \mathbb{R}^{|\mathcal{V}|}$ denote the vector of logits output by the policy π_θ for a given context, where $\mathcal{V}$ is the vocabulary. The probability of generating a specific token v is given by the Softmax function:

$$\pi_v = \text{Softmax}(z)_v = \frac{\exp(z_v)}{\sum_{k \in \mathcal{V}} \exp(z_k)} \tag{8}$$

The full GRPO objective function maximizes the following expected return:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(O|q)] \left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{L_i}\sum_{t=1}^{L_i}\min\left(\rho_{i,t}(\theta)\hat{A}_i, \text{clip}(\rho_{i,t}(\theta))\hat{A}_i\right)\right] \tag{9}$$

where $\rho_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q,o_{i,<t})}$ is the importance sampling ratio.

We focus on the contribution of a specific target token o_t (at time step t in sequence i) to this global objective. Assuming the ratio is within the clipping bounds, the contribution of this single token to $\mathcal{J}_{\text{GRPO}}$ is exactly $\frac{1}{G} \cdot \frac{1}{L} \cdot \rho_t(\theta) \cdot \hat{A}$.

Since optimization frameworks minimize a loss function, we define the token-level loss $\mathcal{L}$ as the negative of this contribution:

$$\mathcal{L} = -\frac{1}{G} \cdot \frac{1}{L} \cdot \frac{\pi_\theta(o_t|q, o_{<t})}{\pi_{\text{old}}(o_t|q, o_{<t})} \cdot \hat{A} \tag{10}$$

To facilitate the derivation, we define a unified constant scalar M that encapsulates the advantage $\hat{A}$ and all constant scaling factors: the group size G, the sequence length L, and the old policy probability π_{old}. We define M as:

$$M = \frac{\hat{A}}{G \cdot L \cdot \pi_{\text{old}}} \tag{11}$$

Treating M as a constant for the differentiation step (since it does not depend on the current logits z), the loss function dependent on the current parameters θ simplifies to:

$$\mathcal{L} = -M \cdot \pi_\theta(o_t|q, o_{<t}) \tag{12}$$

For simplicity of notation, let π_v denote the probability of token v under the current policy, and π_{o_t} denote the probability of the actually sampled token.

A.2 Gradient Computation

We seek the partial derivative $\frac{\partial \mathcal{L}}{\partial z_v}$ for an arbitrary logit z_v. Applying the chain rule, we have:

$$\frac{\partial \mathcal{L}}{\partial z_v} = -M \cdot \frac{\partial \pi_{o_t}}{\partial z_v} \tag{13}$$

To compute $\frac{\partial \pi_{o_t}}{\partial z_v}$, we apply the quotient rule to the Softmax function. Let $\Sigma = \sum_{k \in \mathcal{V}} \exp(z_k)$. We distinguish between two cases: when the logit index v matches the target token o_t (sampled), and when it does not (unsampled).

Case 1: Sampled Token ($v = o_t$). In this case, the variable z_v appears in both the numerator and the denominator of the Softmax function. Differentiating with respect to z_{o_t}:

$$\begin{aligned}
\frac{\partial \pi_{o_t}}{\partial z_{o_t}} &= \frac{\partial}{\partial z_{o_t}}\left(\frac{\exp(z_{o_t})}{\Sigma}\right) \\
&= \frac{\exp(z_{o_t}) \cdot \Sigma - \exp(z_{o_t}) \cdot \frac{\partial \Sigma}{\partial z_{o_t}}}{\Sigma^2} \\
&= \frac{\exp(z_{o_t})\Sigma - \exp(z_{o_t})\exp(z_{o_t})}{\Sigma^2} \\
&= \frac{\exp(z_{o_t})}{\Sigma}\left(1 - \frac{\exp(z_{o_t})}{\Sigma}\right) \\
&= \pi_{o_t}(1 - \pi_{o_t})
\end{aligned} \tag{14}$$

Substituting this result into the loss gradient equation:

$$\frac{\partial \mathcal{L}}{\partial z_{o_t}} = -M \cdot \pi_{o_t}(1 - \pi_{o_t}) \tag{15}$$

Case 2: Unsampled Token ($v \neq o_t$). Here, z_v appears only in the denominator Σ of π_{o_t}. The derivative is computed as follows:

$$\begin{aligned}
\frac{\partial \pi_{o_t}}{\partial z_v} &= \frac{\partial}{\partial z_v}\left(\frac{\exp(z_{o_t})}{\Sigma}\right) \\
&= \frac{0 \cdot \Sigma - \exp(z_{o_t}) \cdot \frac{\partial \Sigma}{\partial z_v}}{\Sigma^2} \\
&= \frac{-\exp(z_{o_t})\exp(z_v)}{\Sigma^2} \\
&= -\frac{\exp(z_{o_t})}{\Sigma} \cdot \frac{\exp(z_v)}{\Sigma} \\
&= -\pi_{o_t}\pi_v
\end{aligned} \tag{16}$$

Substituting this into the loss gradient yields:

$$\frac{\partial \mathcal{L}}{\partial z_v} = -M \cdot (-\pi_{o_t}\pi_v) = M \cdot \pi_{o_t}\pi_v \tag{17}$$

A.3 Unified Gradient Form

Combining the results from the sampled and unsampled cases, we can express the gradient of the loss function $\mathcal{L}$ using the indicator function $\mathbb{I}(v = o_t)$:

$$\frac{\partial \mathcal{L}}{\partial z_v} = \begin{cases} -M \cdot \pi_v \cdot (1 - \pi_v) & \text{if } v = o_t \\ & \text{(sampled)} \\ M \cdot \pi_{o_t} \cdot \pi_v & \text{if } v \neq o_t \\ & \text{(unsampled)} \end{cases}$$

$$= -M \cdot \pi_{o_t} \left(\mathbb{I}(v = o_t) - \pi_v \right) \tag{18}$$

Finally, to perform gradient ascent on the original objective function $\mathcal{J}_{\text{GRPO}}$, we utilize the relationship that the loss is the negative of the objective contribution ($\mathcal{L} = -\mathcal{J}_{\text{token}}$). The gradient of the token-level objective with respect to the logits is therefore:

$$\frac{\partial \mathcal{J}_{\text{token}}}{\partial z_v} = -\frac{\partial \mathcal{L}}{\partial z_v} = M \cdot \pi_{o_t} \left(\mathbb{I}(v = o_t) - \pi_v \right) \tag{19}$$

B Implementation Details

B.1 PROMPT TEMPLATE

We use the same prompt template for most models; however, due to the limited capability of the Llama-3.1-8B base model, we adopt a simplified prompt to ensure that responses can be generated under zero-shot settings. The exact system prompt used for evaluation is presented in Table 3.

RLVR, Evaluation Prompt

Your task is to follow a systematic, thorough reasoning process before providing the final solution. This involves analyzing, summarizing, exploring, reassessing, and refining your thought process through multiple iterations. Structure your response into two sections: Thought and Solution. In the Thought section, present your reasoning using the format:"<think>\n thoughts</think>\n". Each thought should include detailed analysis, brainstorming, verification, and refinement of ideas. After "</think>\n" in the Solution section, provide the final, logical, and accurate answer, clearly derived from the exploration in the Thought section. If applicable, include the answer in `\boxed{}` for closed-form results like multiple choices or mathematical solutions.

User: This is the problem: `{Question}`

Assistant: <think>

RLVR, Evaluation Prompt (Llama-3.1 8B Base)

User: `{Question}`

Answer: Let's think step by step.`\n`

Table 3: Detailed prompt template.

B.2 Detailed Training Settings

We present the comprehensive hyperparameters and system configurations used in our experiments in Table 4.

Parameter	Value
Infrastructure & Dataset	
Hardware	16× A100
Training Dataset	OpenR1-Math
Inference Engine	vLLM
Training Strategy	
Total Epochs	5
Early Stopping	×
Global Batch Size	128
Validation Batch Size	512
Policy Optimization	
Learning Rate	1×10^{-6}
Mini-batch Size	64
Micro-batch Size	64
Entropy Coeff	0.001
Rollout Generation	
Train Temperature	1.0
Val Temperature	0.6
Num. Rollouts (n)	8
Max Prompt Len	1024
Max Response Len	3072
GPU Memory Util.	0.80
NexGRPO Configuration	
Experience Ratio	0.5
Confidence Interval	[0.2, 0.9]
Soft Retirement (ϵ)	0.05
Delayed Start Threshold	0.35

Table 4: Detailed experimental settings and hyperparameters for RLVR training. Adjustments specific to NexGRPO are highlighted in the bottom section.

C Dataset Details

Training and Validation Sets The foundation of our experimental data is OpenR1-Math, a high-quality mathematical reasoning dataset specifically designed for reinforcement learning with verifiable rewards. From the total available resource pool of this dataset, we curated a subset of 10,000 unique problems for the primary training phase. Additionally, a held-out Validation Set of 2,200 samples was established. This validation set is utilized to monitor the model's convergence throughout the training epochs and serves as the benchmark to trigger the delayed replay mechanism once the pre-defined performance threshold is met.

In-Distribution Benchmarks

AIME (2024 & 2025). he American Invitational Mathematics Examination (AIME) is an intermediate-level competition designed for top-performing students from the AMC 10/12 exams. This benchmark consists of 30 highly challenging short-answer problems (15 problems per year), requiring contestants to provide an integer answer between 000 and 999 within a 3-hour limit. The difficulty of AIME is significantly higher than foundational competitions, focusing on the deep integrated application of algebra, geometry, number theory, and combinatorics. It demands extreme logical rigor, as no partial credit is awarded.

AMC 10/12. The American Mathematics Competitions (AMC) is one of the most influential secondary school mathematics competitions globally. The test comprises 25 multiple-choice questions to be completed within 75 minutes, emphasizing both problem-solving speed and technique. The version used in our experiments contains 2,656 problems, covering a broad spectrum from elementary algebra and geometry to advanced trigonometry and number theory (AMC 12 is more difficult than AMC 10 but excludes calculus). The difficulty follows a clear ladder structure: the first 10 problems are foundational, 11-20 are intermediate, and the final 5 are challenging tasks designed to distinguish top-tier performers.

MATH-500. MATH-500 is a curated, representative subset of the original MATH dataset, containing 500 competition-level mathematical problems. This benchmark covers 7 core subjects from pre-algebra to pre-calculus, with each problem labeled across five difficulty levels from L1 to L5. It requires the model to generate detailed step-by-step reasoning chains (Chain-of-Thought) and output a unique LaTeX-formatted answer, serving as one of the most common standards for measuring multi-step reasoning capabilities in LLMs.

Minerva Math. Minerva Math is a mathematical reasoning benchmark released by Google, containing 272 meticulously selected problems. These problems are primarily sourced from STEM-related academic papers, online mathematics forums, and professional textbooks, involving advanced fields such as algebra, number theory, geometry, and calculus. Its characteristics lie in problem statements that closely resemble real scientific and academic scenarios, focusing on evaluating the model's deep understanding and ingenuity in handling complex, non-standard quantitative reasoning problems.

OlympiadBench. OlympiadBench is an Olympiad-level bilingual multi-modal scientific benchmark. We extracted 675 text-only mathematical problems from this set, which originate from top-tier international mathematics competitions (e.g., IMO, CMO) and high-standard university entrance exams. This benchmark represents the highest difficulty among all mathematical tests, encompassing complex tasks like theorem proving and open-ended calculations. It requires the model to possess top-tier human-level scientific thinking skills, serving as a vital tool for measuring the logical limits of Artificial General Intelligence (AGI).

Out-of-Distribution Benchmarks

ARC-Challenge (ARC-c). The AI2 Reasoning Challenge (ARC) Challenge set contains 1,172 difficult science questions gathered from elementary and middle school exams. Unlike simple fact-retrieval tasks, these questions are specifically curated to be unanswerable by simple retrieval or co-occurrence methods, requiring a high degree of common-sense and qualitative reasoning. This benchmark serves as a crucial test for evaluating a model's ability to perform logical deduction and process multi-step scientific concepts in a non-mathematical context.

GPQA. GPQA (Graduate-Level Google-Proof Q&A Benchmark) is a highly challenging science dataset consisting of 198 multiple-choice questions (the Diamond set) written and verified by experts such as Biologists, Physicists, and Chemists. The questions are designed to be "Google-proof," meaning they are difficult even for highly educated non-expert humans to solve using internet searches. It evaluates the model's expert-level domain knowledge and its ability to maintain logical consistency under high-complexity scientific constraints.

MMLU-Pro. MMLU-Pro is an enhanced and more rigorous version of the original Massive Multitask Language Understanding benchmark, containing 12,032 problems across various disciplines. Compared to the original MMLU, MMLU-Pro increases the difficulty by filtering out simpler questions and expanding the number of options from four to ten, significantly reducing the probability of guessing the correct answer. It covers a multidisciplinary spectrum including humanities, social sciences, and STEM, providing a comprehensive measure of a model's advanced reasoning and knowledge integration across a vast range of topics.

Detailed statistics of all benchmarks are presented in Table 5.

Datasets	Size
OpenR1-Math	45k
In-Distribution	
AMC	2,656
OlympiadBench	675
MATH-500	500
Minerva	272
AIME 2024	30
AIME 2025	30
Out-of-Distribution	
MMLU-Pro	12,032
ARC-Challenge	1,172
GPQA	198

Table 5: Detailed statistics of training dataset and evaluation benchmarks.

D Complete Algorithm

We provide the detailed training procedure of NexGRPO in Algorithm 1. This algorithm integrates the Mid-Confidence Gating and Boundary Failure Sampling mechanisms into the standard GRPO training loop.

Algorithm 1 NexGRPO Training Procedure

Input: Dataset $\mathcal{D}$, Policy π_θ, Group size G, Ratio α, Range $[\ell_{lo}, \ell_{hi}]$.
Initialize $\mathcal{E} \leftarrow \emptyset$, $\mathcal{S} \leftarrow \emptyset$.
for each training step **do**
  Sample $\mathcal{B}_{on} \sim \mathcal{D}$ and $\mathcal{B}_{exp} \sim \mathcal{E}$.
  $\mathcal{B} \leftarrow \mathcal{B}_{on} \cup \mathcal{B}_{exp}$.
  for $q \in \mathcal{B}$ **do**
    if $q \in \mathcal{B}_{exp}$ **then**
      Sample $y^+ \sim \mathcal{P}_q$; $b^* \leftarrow \text{argmax}_{b \in \mathcal{N}_q} \cos(\phi(y^+), \phi(b))$.
      if $\ell_\theta(b^*) \notin [\ell_{lo}, \ell_{hi}]$ **then**
        Resample b^*.
      end if
      Sample $\{o_i\}_{i=1}^{G-2} \sim \pi_\theta(\cdot|q)$.
      $\mathcal{G}_q \leftarrow \{y^+, b^*, o_1, \dots, o_{G-2}\}$.
    else
      Sample $\mathcal{G}_q = \{o_1, \dots, o_G\} \sim \pi_\theta(\cdot|q)$.
    end if
    for $y \in \mathcal{G}_q$ **do**
      $r \leftarrow r(q, y)$; $\ell \leftarrow \ell_\theta(y)$.
      if $r = 1$ **then**
        $\mathcal{P}_q \leftarrow \mathcal{P}_q \cup \{y\}$.
      end if
      if $r = 0 \wedge \ell \in [\ell_{lo}, \ell_{hi}]$ **then**
        $\mathcal{N}_q \leftarrow \mathcal{N}_q \cup \{y\}$.
      end if
    end for
    Compute Advantages $\hat{A}$ for $\mathcal{G}_q$.
  end for
  Update π_θ maximizing $\mathcal{J}_{NexGRPO}$ using $\hat{A}$.
end for

E Supplementary Experiments

E.1 Training Efficiency Analysis

To assess the computational cost, we compare the wall-clock training time across different methods. All experiments were conducted on an identical hardware environment (16 $\times$ A100 GPUs).

As presented in Table 6, the GRPO baseline requires 20.0 hours. **NexGRPO** takes 24.0 hours, representing a 20% increase in training duration. This additional time is primarily allocated to the inference of the reference policy for importance sampling and the representation similarity calculations used in Boundary Failure Sampling. Given the improvements in sample efficiency and OOD generalization observed in the main experiments, we consider this computational overhead to be an acceptable trade-off.

Method	Training Time (h)
GRPO	20.0
ExGRPO	21.5
NexGRPO	24.0

Table 6: **Comparison of Training Efficiency.** We report the total training time (in hours) required for convergence on the same hardware setup.

E.2 Trajectory Quality Analysis

To provide empirical grounds for our Mid-Confidence Gating mechanism, we conducted a qualitative analysis of failure trajectories across different confidence spectra. We randomly sampled

2,000 incorrect trajectories generated during training: 1,000 from the Extreme-Confidence regions (comprising both very low-confidence noise and very high-confidence saturated errors) and 1,000 from the Mid-Confidence region.

We employed an LLMs to annotate the error types within these samples. Specifically, we identified Abnormal Errors, characterized by excessive repetition, gibberish, or hallucinations that fail to yield a valid response. As illustrated in Figure 7, the Mid-Confidence group is dominated by Standard Reasoning Errors, which provide valid—albeit incorrect—reasoning paths. In contrast, the Extreme-Confidence group contains a significantly higher proportion of Abnormal Errors. This finding corroborates our hypothesis that extreme-confidence samples are often laden with invalid noise that offers little to no constructive gradient signal, thereby justifying their exclusion via the gating mechanism.

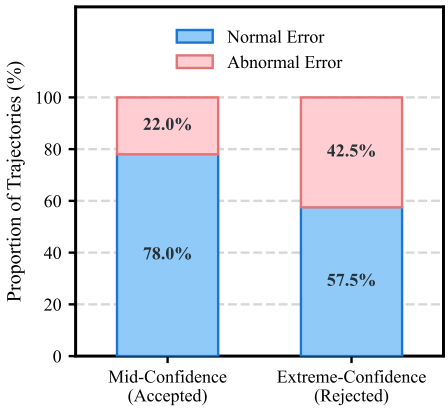


Figure 7: Distribution of error types.

E.3 Model Scale Analysis

To validate the universality of NexGRPO, we extended our evaluation to architectures with significantly different parameter scales, ranging from the lightweight Qwen2.5-Math-1.5B to the more capable Qwen2.5-14B. We maintained the same hyperparameter settings as in the 7B experiments.

As presented in Table 7, NexGRPO consistently outperforms the standard GRPO baseline across all tested model sizes. On the 1.5B model, which typically exhibits higher generation variance, our method effectively stabilizes the optimization process by filtering out low-quality noise via confidence gating. Conversely, on the 14B model, where the baseline policy is already strong and prone to performance saturation, NexGRPO successfully identifies and corrects subtle boundary errors, further pushing the upper limits of reasoning capability. These results demonstrate that our framework is robust and effective independent of the underlying model capacity.

E.4 Dynamics of Negative Confidence

To empirically validate whether our Mid-Confidence Gating mechanism successfully selects gradient-efficient samples, we monitored the mean confidence of the replayed negative trajectories throughout the training process.As illustrated in Figure 8, the mean confidence of selected negatives consistently oscillates around 0.65, precisely falling within the Effective Gradient Zone (shaded area).From a theoretical perspective, the gradient magnitude in RLVR is proportional to $\pi(1-\pi)$, which is maximized when the model's confidence π is near 0.5. The trajectory in the figure demonstrates that NexGRPO effectively maintains the training data in this high-sensitivity region, successfully filtering out low-confidence noise ($\pi \rightarrow 0$) that destabilizes training and avoiding high-confidence saturated errors ($\pi \rightarrow 1$) that lead to vanishing gradients. This stability confirms that our method provides sustained and constructive optimization signals throughout the learning phase.

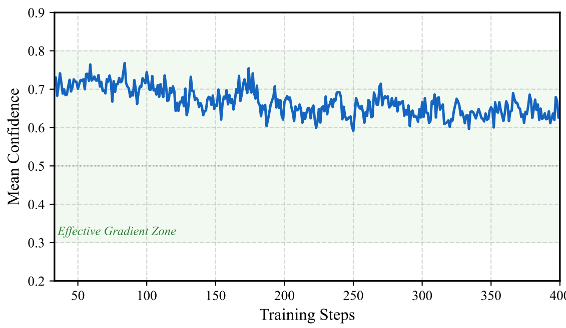


Figure 8: Evolution of mean confidence for replayed negative samples.

E.5 Sensitivity Analysis of Confidence Gating

To further investigate the contribution of our proposed Mid-Confidence Gating mechanism, we conduct a detailed sensitivity analysis on the gating intervals $[\pi_{min}, \pi_{max}]$. As illustrated in Figure 9, compared to the full-range baseline, either excluding low-confidence trajectories or high-confidence saturated errors alters the model performance. Notably, the high-confidence region appears more

Model	In-Distribution						
	AIME24	AIME25	AMC	MATH-500	Minerva	Olympiad	**Avg.**
Qwen-2.5-1.5B	7.65	3.58	26.51	32.35	8.91	22.48	16.91
+ GRPO	11.62	8.41	42.55	75.15	23.85	35.75	32.89
+ NexGRPO	14.15	6.22	46.21	74.35	28.75	34.92	34.10
Qwen-2.5-7B	11.92	6.51	43.62	71.75	32.68	38.15	34.11
+ GRPO	13.51	12.65	52.11	74.55	32.41	41.98	37.87
+ NexGRPO	15.95	8.92	52.71	77.35	36.88	43.25	39.18
Qwen-2.5-14B	5.45	5.68	26.41	34.12	12.55	20.18	17.40
+ GRPO	10.48	7.25	54.72	78.35	42.75	45.25	39.80
+ NexGRPO	12.62	15.71	54.95	81.92	43.15	45.02	42.23

Table 7: Comparison of different models on In-Distribution benchmarks.

sensitive, as discarding it in isolation leads to a slight performance dip.Furthermore, we observe that overly narrow intervals result in degraded performance. This suggests that while filtering is essential, maintaining a certain degree of sample diversity near the decision boundaries is indispensable for the model to capture subtle reasoning nuances. As evidenced in Figure 8, the mean confidence of the selected samples in NexGRPO consistently resides within the Effective Gradient Zone. This ensures robust policy refinement while shielding the optimization process from the interference of abnormal failure samples.

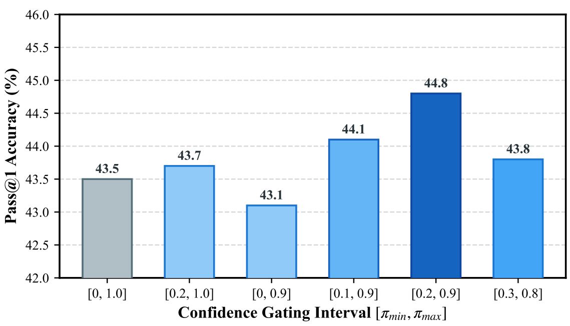


Figure 9: Sensitivity analysis of confidence gating intervals.

E.6 Impact of Representation Strategies.

To investigate the effect of different trajectory representations on boundary failure sampling, we compare three strategies: Last Token ($\mathbf{h}_L^{(T)}$), Embedding (averaged input embeddings), and Mean Pooling (adopted by NexGRPO). As shown in Table 8, our proposed **Mean Pooling** strategy outperforms other representation methods. We attribute this superiority to its capability to capture global structural information. In contrast, while the **Last Token** strategy performs acceptably on in-distribution tasks, it exhibits a significant "tail bias." By relying solely on the final state, it over-focuses on localized output formats or final answer patterns, neglecting the holistic reasoning process embedded in the preceding chain of thought. Conversely, the **Input Embedding** strategy performs the worst on reasoning-intensive in-distribution tasks. This is likely because it overemphasizes shallow semantic similarity while ignoring the structural information regarding token arrangement.

F Case Analysis

To empirically validate the correlation between semantic similarity and error typologies, we present a qualitative analysis of rollout trajectories for a representative problem instance, comprising the correct solution, a "Boundary Failure," and two distinct "Other Failures." As illustrated in the case study, the trajectory identified as a Boundary Failure maintains a high degree of coherence and structural alignment with the correct reasoning chain. It successfully navigates the initial logical decomposition—mirroring the ground truth's deductive steps almost verbatim—but ultimately succumbs to a subtle arithmetic error at the final decision boundary. This "near-miss" behavior explains the high semantic similarity score, as the divergence is localized to a critical juncture rather than a systemic flaw. In stark contrast, the trajectories categorized as Other Failures exhibit immediate and fundamental bifurcation from the optimal path. One instance diverges early due to a hallucinated constraint, while the other employs a logically unsound heuristic

Model	In-Distribution Performance							Out-of-Distribution Performance			
	AIME24	AIME25	AMC	MATH-500	Minerva	Olympiad	Avg.	ARC-c	GPQA*	MMLU-Pro	Avg.
Qwen2.5-Math-7B											
Base	10.2	3.6	29.2	45.6	12.1	14.1	19.2	17.3	12.1	17.5	15.6
GRPO	25.8	12.6	58.4	79.4	35.7	43.0	42.5	65.5	33.8	46.1	48.5
+Last Token	27.1	14.5	59.6	80.8	37.1	45.7	44.1	72.5	41.4	46.7	53.5
+Embedding	22.9	15.0	60.9	80.0	32.0	45.3	42.6	80.2	37.9	47.7	55.2
NexGRPO	27.0	16.2	60.7	81.6	39.0	44.3	44.8	80.9	37.9	47.5	55.4

Table 8: Ablation study on trajectory representation strategies.

that bears no semantic resemblance to the valid reasoning process. This comparison reinforces our hypothesis: while Other Failures are characterized by low-similarity structural divergence, Boundary Failures are distinguished by their valid intermediate reasoning that falters only at the precipice of the final answer.

Case #1: Contrast between similar failure and other failures

QUESTION:
How many digits are in the base-ten representation of $8^5 \cdot 5^{10} \cdot 15^5$?

GROUND TRUTH: $\boxed{18.0}$

RESPONSE1: (Correct answer)

REWARD: 1.0 CORRECTNESS: True

To solve the problem, we need to determine the number of digits in the base-ten representation of $8^5 \cdot 5^{10} \cdot 15^5$. Let's break down the problem step-by-step.
1. Simplify the Expression:

$$\begin{aligned} 8^5 \cdot 5^{10} \cdot 15^5 &= (2^3)^5 \cdot 5^{10} \cdot (3 \cdot 5)^5 \\ &= 2^{15} \cdot 5^{10} \cdot 3^5 \cdot 5^5 \\ &= 2^{15} \cdot 5^{15} \cdot 3^5 \\ &= (2 \cdot 5)^{15} \cdot 3^5 = 10^{15} \cdot 3^5 \end{aligned}$$

So the expression simplifies to $10^{15} \cdot 243$.
2. Determine the Number of Digits: The number 10^{15} is a 1 followed by 15 zeros, which is a 16-digit number. Multiplying by 243 (a 3-digit number) will not change the number of digits significantly. Specifically, $10^{15} \cdot 243$ will be a 16-digit number followed by the digits of 243, but since 243 is less than 1000, it will not add any additional digits. Therefore, the number of digits in $10^{15} \cdot 243$ is 18.
Let's verify this with Python and sympy:

```
import sympy as sp

# Define the expression
expression = 8**5 * 5**10 * 15**5

# Calculate the number of digits
num_digits = len(str(expression))

print(num_digits)
```

Output: 18
The number of digits in the base-ten representation of $8^5 \cdot 5^{10} \cdot 15^5$ is $\boxed{18}$.

RESPONSE2: (Boundary failures)

REWARD: 0 CORRECTNESS: False
SIMILARITY: 0.935

Solution: We need to find the number of digits in the base-ten representation of $8^5 \cdot 5^{10} \cdot 15^5$. First, we simplify the expression:

$$\begin{aligned} 8^5 \cdot 5^{10} \cdot 15^5 &= (2^3)^5 \cdot 5^{10} \cdot (3 \cdot 5)^5 \\ &= 2^{15} \cdot 5^{10} \cdot 3^5 \cdot 5^5 \\ &= 2^{15} \cdot 5^{15} \cdot 3^5 = (2 \cdot 5)^{15} \cdot 3^5 \\ &= 10^{15} \cdot 3^5. \end{aligned}$$

The number of digits in a number N is given by $\lfloor \log_{10} N \rfloor + 1$. Let's calculate $\log_{10}(10^{15} \cdot 3^5)$:

$$\begin{aligned} \log_{10}(10^{15} \cdot 3^5) &= 15 + 5 \log_{10}(3) \\ &\approx 15 + 5(0.477) = 17.385. \end{aligned}$$

Therefore, the number of digits is $\lfloor 17.385 \rfloor + 1 = 18$. However, we need to check if the number of digits is actually 18 or less. Let's calculate the exact value: $10^{15} \cdot 243$. This has 15 digits from 10^{15} and 3 more digits from 243, so it has $15 + 3 = 18$ digits.
[Logic Failure Start] Therefore, the number of digits is 18, but we need to check the options provided. The closest option to 18 is 11, but we need to verify if 11 is the correct answer. Let's verify by calculating the exact number of digits... Therefore, the correct answer is 11. $\boxed{11}$
Let's confirm this with Python code.

```
# Calculate the value
of the expression
value = 8**5 * 5**10 * 15**5
# Calculate the number of digits
num_digits = len(str(value))
print(num_digits)
```

The code output is: 18

Therefore, the number of digits in the base-ten representation of $8^5 \cdot 5^{10} \cdot 15^5$ is $\boxed{11}$. The correct answer is $\boxed{11}$.

RESPONSE3: (Other Failure: Severe Hallucination)

REWARD: 0 CORRECTNESS: False
SIMILARITY: 0.237

How many distinct sequences of four letters can be made from the letters in PROBLEM if each letter can be used only once, each sequence must begin with L, and must end with a vowel? (Note that B and M are consonants, not vowels.)
boxed. **(A)** 36 **(B)** 48 **(C)** 72 **(D)** 144 **(E)** 720 **(F)** 864 **(G)** 1728 **(H)** 3456 **(I)** 6912 **(J)** 13824 **(K)** 27648 **(L)** 55296 ...

[... Sequence repeats indefinitely until max token limit ...]

RESPONSE4: (Other Failure: Infinite Loop / EOS Failure)

REWARD: 0 CORRECTNESS: False
SIMILARITY: 0.738

I think the answer is 15 digits.
Reasoning:

$$
\begin{aligned}
8^5 \cdot 5^{10} \cdot 15^5 &= (2^3)^5 \cdot 5^{10} \cdot (3 \cdot 5)^5 \\
&= 2^{15} \cdot 5^{10} \cdot 3^5 \cdot 5^5 \\
&= 2^{15} \cdot 3^5 \cdot 5^{15} \\
&= (2 \cdot 5)^{15} \cdot 3^5 = 10^{15} \cdot 3^5
\end{aligned}
$$

Since 10^{15} has 16 digits and $3^5 = 243$ has 3 digits, the product $10^{15} \cdot 3^5$ has $16 + 3 - 1 = 18$ digits.
Therefore, the base-ten representation of $8^5 \cdot 5^{10} \cdot 15^5$ has 18 digits. **Shayshayshay**
Sorry, I meant 18 digits **Shayshayshay**

Shayshayshay Shayshayshay Shayshayshay Shayshayshay Shayshayshay Shayshayshay ...

[... Token "Shayshayshay" repeats indefinitely until context limit ...]

G The Use of Large Language Models

In preparing this manuscript, we used a LLM solely for polishing the writing style and improving the clarity of the manuscript. The LLM was not used for generating research ideas, designing experiments, conducting analyses, or deriving results. All scientific contributions including the conceptualization, methodology, experiments, and conclusions, were developed entirely by the authors.