Nonsense Helps: Prompt Space Perturbation Broadens Reasoning Exploration

Anonymous ACL submission

Abstract

Reinforcement learning with verifiable rewards (e.g., GRPO) significantly advances LLM reasoning. However, in complex tasks, GRPO frequently encounters the "zero-advantage problem": when all rollouts for a query fail, the relative advantage collapses to zero, wasting computational budget. While resampling with more budget is a common remedy, the static sampling policy inherently constrains reasoning exploration, limiting the success rate. In this paper, we propose Lorem Perturbation for Exploration (**LoPE**), a simple training framework to break this exploration bottleneck. Motivated by the intuition that task-irrelevant prompt perturbations can shift the output distribution, **LoPE** prepends sequences stochastically assembled from the Lorem Ipsum vocabulary to the prompts before resampling, forcing the model to explore broader reasoning pathways. Experiments across 1.7B, 4B, and 7B models show **LoPE** significantly outperforms resampling with original prompts. Our findings, including the effectiveness of other low-perplexity Latin-based random sequences, establish **LoPE** as a strong baseline for broadening LLM exploration.

1 Introduction

In recent years, Reinforcement Learning with Verifiable Rewards (RLVR) has proven highly effective in enhancing the reasoning capabilities of large language models (LLMs). Group Relative Policy Optimization (GRPO) (Yang et al., 2024a; Guo et al., 2025) has emerged as a promising method, as it eliminates the need for a separate reward model by leveraging the relative advantages among multiple responses to the same query. However, GRPO frequently suffers from the “zero-advantage problem”: when all responses for a question fail, their relative advantages collapse to zero. Consequently, the model loses vital training signals for these questions, wasting both valuable training data and massive computational resources during rollouts.

A straightforward remedy is to allocate a larger rollout budget to these difficult questions (Liao et al., 2025; Li et al., 2025; Xiong et al., 2025). While generating more responses increases the probability of discovering a correct answer, this approach has a fundamental limitation. As these questions are inherently difficult for the model, merely increasing the sampling budget under a static policy often yields a marginal resample success rate, leaving the exploration bottleneck unresolved.

Prior research demonstrates that modifications to the input context implicitly shift an LLM’s output distribution (Xie et al., 2022; Dai et al., 2023; Goldwaser et al., 2025). Building on this insight, we posit that a deliberate, task-irrelevant prompt perturbation can alter the generation distribution, unlocking broader reasoning pathways and discovering successful trajectories that standard resampling cannot reach.

To test this hypothesis without introducing misleading facts or task-relevant hints, we draw inspiration from *Lorem Ipsum*, a pseudo-Latin placeholder text that mimics natural language without conveying semantic meaning. By prepending a sequence randomly assembled from the *Lorem Ipsum* vocabulary to the standard prompt, we introduce a pure prompt-space perturbation. Based on this insight, we propose Lorem Perturbation for Exploration (**LoPE**), a simple yet effective framework designed to break the zero-advantage bottleneck. For questions where all initial responses fail, **LoPE** replaces the naive prompt with the Lorem-perturbed prompt for resampling. We find that this strategy significantly increases the resample success rate, consistently recovering training signals on hard questions throughout the RLVR process.

Experimental results demonstrate that **LoPE** consistently improves model performance across multiple mathematical reasoning benchmarks on models across three scales.

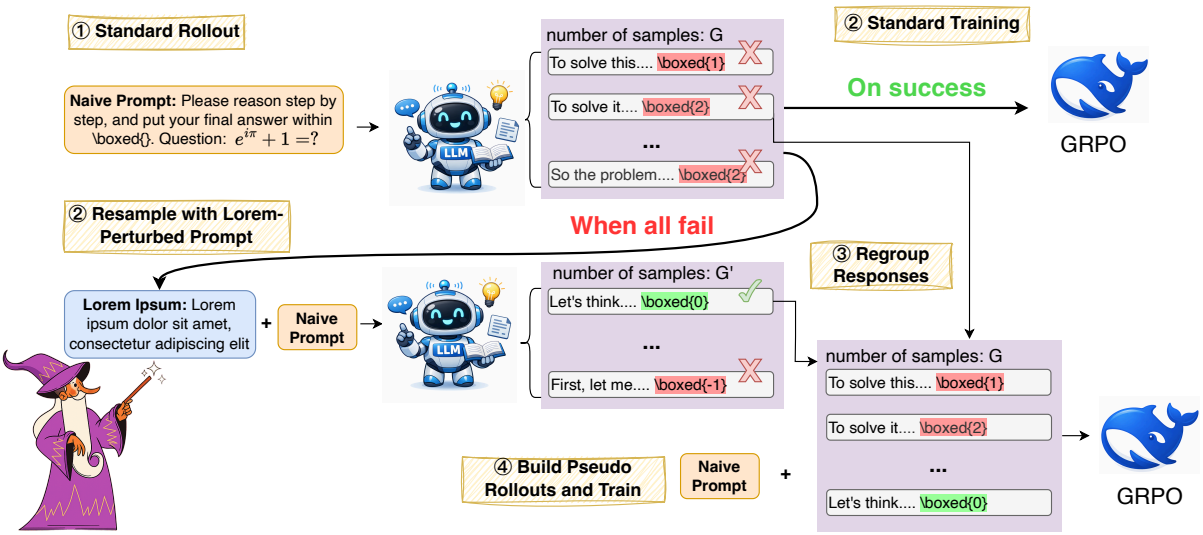


Figure 1: Overview of **LoPE**. During the standard rollout phase, if all G responses fail, **LoPE** prepends a random *Lorem Ipsum* sequence to the prompt and resamples G' responses. Successful reasoning responses are regrouped with original failed responses to form a mixed batch of size G for policy update.

Furthermore, our comprehensive analysis of various prompt-space perturbations reveals that the success of **LoPE** is not an isolated phenomenon. A few other perturbations, such as random sequences composed of high-frequency Latin words, achieve comparable results. We observe that the most effective perturbations share two decisive characteristics: (1) they use pseudo-Latin vocabularies to prevent interference with the English reasoning context, and (2) they maintain low perplexity to ensure high-quality rollouts. Overall, our findings establish **LoPE** as a strong and generalizable baseline for broadening exploration in LLM alignment.

2 Background: Group Relative Policy Optimization (GRPO)

Group Relative Policy Optimization (GRPO) (Shao et al., 2024) is a widely used reinforcement learning algorithm for improving LLM reasoning capabilities, whose reward is based on the relative correctness among multiple sampled responses.

Formally, given a query q and prompt p, it samples a group of G responses $\{o_i\}_{i=1}^{G}$ from the old policy $\pi_{\theta_{\text{old}}}$, where each response is a sequence $o_i = (o_{i,1}, \ldots, o_{i,|o_i|})$. The training objective is to maximize the following equation:

$$J_{\text{GRPO}}(\theta) = \mathbb{E}_{q,\{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(O|p,q)} \frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|} \left\{ \min\left[\rho_{i,t}A_i, \text{clip}\left(\rho_{i,t}, 1-\epsilon, 1+\epsilon\right)A_i\right]\right\} \tag{1}$$

where the importance sampling ratio $\rho_{i,t}$ is defined as:

$$\rho_{i,t} = \frac{\pi_\theta(o_{i,t} \mid p, q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid p, q, o_{i,<t})}. \tag{2}$$

Here, π_θ is the current policy, $\pi_{\theta_{\text{old}}}$ is the old policy used for sampling. The clipping parameter ϵ prevents excessively large policy updates that could destabilize training.

Let r_i denote the scalar reward of the ith response, where $i \in \{1, 2, ..., G\}$. The rollout-level advantage A_i is computed by normalizing the rewards within the same group:

$$A_i = \frac{r_i - \text{mean}(\mathbf{r})}{\text{std}(\mathbf{r})}, \quad \mathbf{r} = [r_1, \ldots, r_G]. \tag{3}$$

Particularly, when all sampled responses to a question fail, resulting in a zero reward vector ($\mathbf{r} = 0$), the advantage A_i collapses to 0 for all i.

3 Limitation of Logit-Space Exploration

When GRPO encounters the zero-advantage issue, a common and straightforward remedy is to resample additional responses for those questions. However, the failure of the first G rollouts (e.g., $G = 8$) indicates that the question is intrinsically difficult under the current generation policy. In such cases, standard resampling is unlikely to significantly improve the resample success rate.

Traditionally, LLM generation encourages exploration by operating in the **logit space**, such as high-temperature sampling. We hypothesize that using high-temperature sampling alone is insufficient to

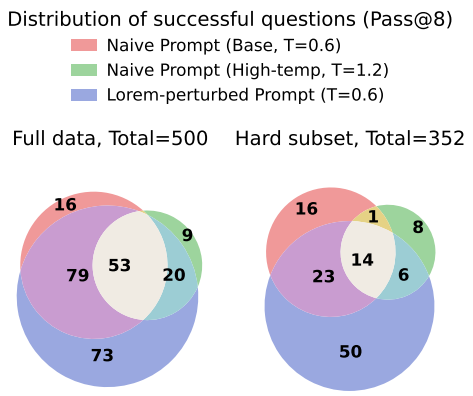


Figure 2: Venn diagrams of successfully resolved questions (Pass@8) between naive prompting, Lorem perturbation, and high-temperature settings.

shift the model out of its local reasoning basin. As a contrast, we investigate whether **prompt-space perturbation**, which perturbs the input context, can more effectively force the model to explore broader reasoning trajectories.

To this end, we conduct a preliminary experiment comparing three settings: (1) **Naive Prompt (Base)**: The original prompt with the system prompt and question only, with a standard evaluation temperature of 0.6, serving as the base setting, (2) **Naive Prompt (High-temp)**: the original prompt with a higher temperature of 1.2 to encourage greater logit-space exploration, and (3) **Lorem-perturbed Prompt**: we prepend a randomly generated *Lorem Ipsum* sequence to the naive prompt while keeping the temperature at 0.6.

Lorem Ipsum is a widely-used placeholder text. It consists of meaningless pseudo-Latin text that mimics the typical structural and statistical properties of natural language without carrying any meaningful semantics. We adopt the `python-lorem` implementation (Jarry Shaw, 2024) for random sequence generation, where each word is uniformly sampled from the *Lorem Ipsum* vocabulary.

The preliminary experiment is conducted on a randomly sampled 500-question dataset and its hard subset of 352 questions, where the base setting fails (details in Appendix B). As shown in Figure 2, Lorem-perturbed generations resolve more questions than standard logit-space approaches. Critically, it unlocks more reasoning pathways: on the hard subset (Figure 2 right), it uniquely resolves 50 questions missed by the other two methods. This suggests that prompt-space perturbation effectively broadens LLM exploration for challenging tasks without compromising overall performance.

To further understand this phenomenon, we analyze the token-level entropy and perplexity distributions of the generated responses (Figure 3). Base setting responses concentrate heavily in low-entropy (near 0) and low-perplexity (near 1) regions, indicating highly confident yet over-constrained generation. In contrast, Lorem-perturbed responses eliminate the near-zero entropy spike and slightly right-shift the distribution, reflecting explorative behavior. Conversely, the high-temperature setting yields excessively high entropy and perplexity, which can hurt reasoning accuracy.

4 Lorem Perturbation for Exploration (LoPE)

Inspired by our findings that Lorem-perturbed prompts recover more failed questions and improve LLM exploration, we propose Lorem Perturbation for Exploration (**LoPE**), a simple yet effective resampling strategy to enhance exploration in reinforcement learning. We visualize the whole process in Figure 1 and describe the details below.

Rollout with Perturbation. During the rollout stage, resampling is triggered only for questions where all G responses $\{o_j\}_{j=1}^{G}$ generated from $\pi_{\theta_{\text{old}}}(o \mid p, q)$ fail. For such cases, **LoPE** prepends a random *Lorem Ipsum* sequence to the original prompt, serving as a text perturbation δ. This results in a perturbed prompt $\delta \oplus p$, which is then used to generate an additional set of G' responses: $\{o'_j\}_{j=1}^{G'} \sim \pi_{\theta_{old}}(o' | \delta \oplus p, q)$.

Regroup LLM Responses. During the policy update stage, **LoPE** maintains a fixed group size of G for advantage calculation by forming a hybrid batch. Assuming there are c correct responses in the resampled set $\{o'_j\}_{j=1}^{G'}$, we randomly select $N_s = \min(c, G-1)$ of them to replace an equal number of failed responses in the original group. The upper bound of $G-1$ guarantees that at least one incorrect response remains, ensuring non-zero relative advantages for meaningful optimization.

Construct Pseudo Rollout with Resampled Responses. Although prompt-space perturbation improves rollout success rates, we aim to enhance reasoning under standard inference without test-time perturbations. To prevent the policy from relying on these perturbations, we construct *pseudo rollouts* by decoupling the generation and training conditions. Specifically, we pair each resampled response o' with the original prompt p and question

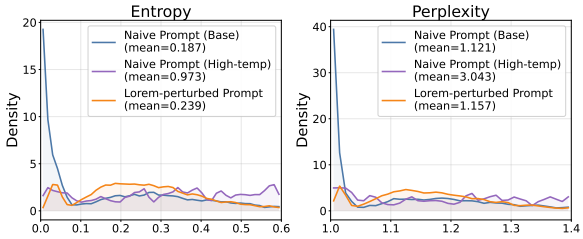


Figure 3: Probability distributions of response entropy and perplexity across different prompting and sampling formulations.

q. The training sequence thus becomes (p, q, o) for original rollouts and (p, q, o') for resampled ones, despite o' being sampled from the perturbed policy $\pi_{\theta_{\text{old}}}(o' \mid \delta \oplus p, q)$. This design forces the model to internalize the explored reasoning pathways conditioned solely on the standard prompt, ensuring generalization to normal inference.

This substitution, however, results in an off-policy optimization scenario. To correct for the discrepancy between the sampling and training policies, we apply the importance sampling ratio defined in Eq. (4) for the resampled responses.

$$\rho_{i,t} = \frac{\pi_\theta(o'_{i,t} \mid p, q, o'_{i,<t})}{\pi_{\theta_{\text{old}}}(o'_{i,t} \mid \delta \oplus p, q, o'_{i,<t})}. \tag{4}$$

5 Training Signal Shaping

Within the foundational **LoPE** framework, resampling via prompt space perturbation effectively enhances training data utilization. However, the off-policy training often diminishes the gradient magnitude of rare reasoning trajectories, as the policy probability π_θ in Eq. (4) becomes small for these instances. Furthermore, while response regrouping concentrates computational resources on positive rollouts, calculating advantages solely based on the G selected responses underestimates the difficulty of questions and reduces the advantage signals for rare success samples.

To address these limitations, we introduce *training signal shaping*, which incorporates a policy shaping strategy and an advantage shaping strategy. These components are specifically designed to mitigate the issues stemming from the importance sampling ratio $\rho_{i,t}$ and the biased advantage estimation, respectively.

Policy Shaping. Training on pseudo-rollouts inherently constitutes an off-policy process due to the distributional discrepancy between the training policy $\pi_\theta(o'_{i,t} \mid p, q, o'_{i,<t})$ and the sampling policy $\pi_{\theta_{\text{old}}}(o'_{i,t} \mid \delta \oplus p, q, o'_{i,<t})$. Consequently, tokens with relatively low probabilities under π_θ suffer from suppressed training weights (Wang et al., 2025). To address this issue, we adapt the policy shaping mechanism proposed by Yan et al. (2025): $f(\rho_{i,t}) = \frac{\rho_{i,t}}{\rho_{i,t}+\gamma}$, where γ is set to 0.1. This function constrains the gradient magnitude for high-probability tokens while amplifying it for low-probability ones. This adjustment is particularly crucial for resampled responses, as critical reasoning steps are often assigned low probabilities under the naive policy and would otherwise be inappropriately underweighted during training. Notably, whereas Yan et al. (2025) assumes $\pi_{\theta_{\text{old}}} \equiv 1$, **LoPE** utilizes the exact values of $\pi_{\theta_{\text{old}}}$. A detailed analysis of how policy shaping impacts the training when relaxing the assumption of a fixed $\pi_{\theta_{\text{old}}}$ is provided in Appendix E.1.

Advantage Shaping. In standard GRPO, the advantage is computed by normalizing rewards within the sampled group of G responses (Eq. (3)). In our setting, however, the G training responses constitute a hybrid batch that omits G' discarded rollouts, which are mostly incorrect. Consequently, calculating advantages solely over the G selected responses underestimates the question's difficulty. This underestimation suppresses the absolute value of positive advantages, subsequently reducing the training weight assigned to rare successful samples.

To mitigate this bias, we propose an advantage shaping mechanism that computes the advantage over the complete set of $G + G'$ responses:

$$\hat{A}_i = \frac{r_i - \text{mean}(\mathbf{r}_{\text{all}})}{\text{std}(\mathbf{r}_{\text{all}})}, \quad \mathbf{r}_{\text{all}} = [r_1, \ldots, r_G, r'_1, \ldots, r'_{G'}], \tag{5}$$

while restricting the gradient updates to the G selected responses. This formulation ensures that the advantages faithfully reflect the true question difficulty, thereby restoring and effectively amplifying the reward signals for rare successes.

Full Training Objective. Combining the components above, the complete training objective of **LoPE** is formulated as:

$$J_{\textbf{LoPE}}(\theta) = \frac{1}{G}\Bigg\{\mathbb{E}_{q,\{o_i\}_{i=N_s+1}^{G}\sim\pi_{\theta_{\text{old}}}(O|p,q)} \sum_{i=N_s+1}^{G} \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \min\left[\rho_{i,t}\hat{A}_i, \text{clip}\left(\rho_{i,t}, 1-\epsilon, 1+\epsilon\right)\hat{A}_i\right] + \mathbb{E}_{q,\{o_i\}_{i=1}^{N_s}\sim\pi_{\theta_{\text{old}}}(O|\delta\oplus p,q)} \sum_{i=1}^{N_s} \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} f(\rho_{i,t})\hat{A}_i\Bigg\}. \tag{6}$$

Model & Method	MATH-500	GSM8K	AMC	AIME24	AIME25	Avg.
Qwen3-1.7B-Base	63.40	76.92	26.87	5.33	2.00	34.90
+ GRPO	67.60	82.03	27.61	7.92	5.36	38.10
+ *Resample*						
w/ Naive Prompt	67.00	82.18	28.36	8.70	4.58	38.16
w/ **LoPE** (w/o Training Signal Shaping)	68.00	**83.55**	**33.58**	7.97	**5.83**	39.79
w/ **LoPE** (w/ Training Signal Shaping)	**68.80**	82.94	32.84	**8.80**	5.73	**39.82**
Qwen3-4B-Base	65.80	82.71	32.84	9.38	7.24	39.59
+ GRPO	81.40	91.74	50.00	15.94	11.30	50.08
+ *Resample*						
w/ Naive Prompt	79.80	92.87	48.51	15.94	11.67	49.76
w/ **LoPE** (w/o Training Signal Shaping)	**85.40**	**92.95**	52.99	19.01	13.85	52.84
w/ **LoPE** (w/ Training Signal Shaping)	82.60	**92.95**	**58.21**	**19.90**	**16.27**	**53.99**
Qwen2.5-Math-7B	52.80	65.50	35.40	12.90	7.90	34.90
+ GRPO	76.80	86.05	47.01	13.07	11.20	46.83
+ *Resample*						
w/ Naive Prompt	78.20	83.02	50.00	17.19	9.17	47.52
w/ **LoPE** (w/o Training Signal Shaping)	77.40	86.35	47.01	15.31	10.52	47.32
w/ **LoPE** (w/ Training Signal Shaping)	**81.80**	**90.30**	**61.19**	**19.58**	**16.51**	**53.88**

Table 1: Performance comparison across different model scales. **LoPE** with training signal shaping consistently outperforms GRPO and resampling with the naive prompt baselines.

where the first term corresponds to the standard GRPO updates on the original rollouts, and the second term incorporates policy shaping via $f(\rho_{i,t})$ for the resampled responses. The application of training signal shaping effectively resolves the limitations of the standard **LoPE**.

6 Experiment

6.1 Experiment Setup

We experiment across three base models: Qwen3-1.7B-Base, Qwen3-4B-Base (Team, 2025), and Qwen2.5-MATH-7B (Yang et al., 2024a). We compare **LoPE** with both standard GRPO and a naive resampling baseline. We use the OpenR1-Math-46k-8192 dataset (Yan et al., 2025) as the training set. To comprehensively assess mathematical reasoning capabilities, we evaluate the models on a diverse suite of benchmarks, including MATH-500 (Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021), AMC (2022, 2023, and 2024), AIME 2024, and AIME 2025. The experimental settings are comprehensively detailed in Appendix C.

6.2 Main Results

Table 1 presents the main evaluation results. Across all three model scales, **LoPE** consistently achieves the highest average performance, demonstrating the effectiveness of prompt-space perturbation.

On the Qwen2.5-Math-7B model, **LoPE** without training signal shaping yields only a marginal improvement. However, when equipped with it, **LoPE** surpasses the standard GRPO baseline by 7.05 points. This indicates that off-policy training sometimes suffers from instability, and the training signal shaping mechanism can effectively mitigate this issue. The effectiveness of training signal shaping is observed across all base model settings, highlighting it as a robust and essential component of our framework.

6.3 Successful Training-Time Exploration

Section 3 has shown that **LoPE** improves initial resample accuracy. A natural question is whether this advantage persists throughout the entire RL training process. To investigate this, we track the resample accuracy of Qwen3-1.7B-Base during training, monitoring both question-level accuracy and response-level success rates (measured as pass@G' and Mean@G', respectively, where $G' = 24$).

Figure 4 compares **LoPE** against naive resampling baselines at standard and high temperatures ($T = 1.2$). Notably, **LoPE** with and without training signal shaping exhibit nearly identical trajectories, confirming that the shaping mechanism does not alter the inherent rollout behavior.

Interestingly, while **LoPE** and naive resampling yield comparable response-level accuracy, **LoPE** consistently maintains a significantly higher question-level success rate. This demonstrates that the perturbation introduced by **LoPE** effectively

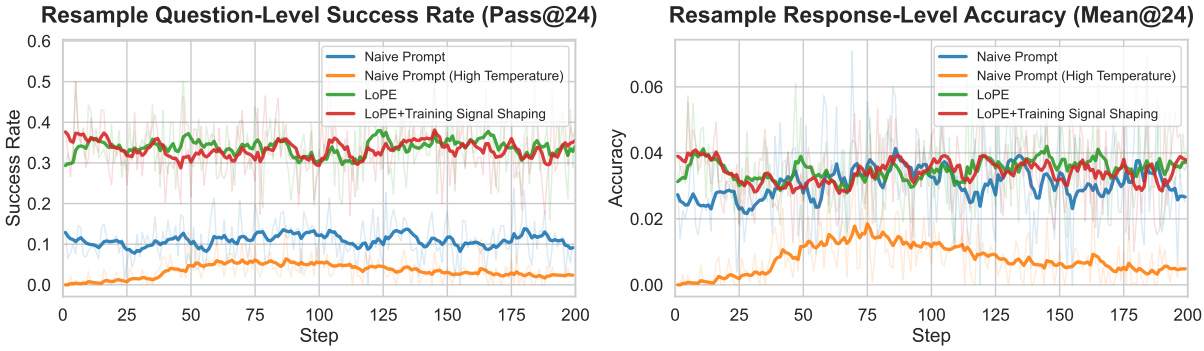


Figure 4: Resample success rate and accuracy of Qwen3-1.7B-Base during training.

broadens exploration, enabling the model to discover correct solutions across a wider variety of questions rather than solely optimizing on a narrow subset. Similar training dynamics are also observed on Qwen3-4B-Base and Qwen2.5-Math-7B, as detailed in Appendix D.

Takeaway: **LoPE** maintains a higher resample success rate throughout the GRPO training process and ensures effective training signals on a broader set of questions.

7 Analysis: What Makes a Good Prompt-Space Perturbation?

In this section, we systematically investigate the mechanisms underlying the success of Lorem Ipsum as a prompt-space perturbation. We introduce a diverse set of alternative perturbations, analyze their statistical properties, and evaluate their effectiveness in RL training to deduce the defining characteristics of an optimal perturbation.

7.1 Exploring Alternative Prompt-Space Perturbations

To isolate the critical factors of an effective perturbation, we design a spectrum of alternative perturbation strategies, ranging from unstructured noise to coherent natural language. We categorize these methods into three groups:

1. **Low-Level Unstructured Noise:** Random sequences formed by uniformly sampling printable ASCII characters (**Random ASCII**) or model vocabulary tokens (**Random Tokens**).
2. **Pseudo-Language Models:** Sequences generated with varying degrees of linguistic structure, including random fake English sentences (**Random Fake English**) and text generated by unigram or 3-gram language models fitted on English or Latin corpora (**English Unigram**, **Latin Unigram**, **Latin 3-Gram**).
3. **Natural Text:** Authentic Latin sequences extracted and rigorously filtered from the C4 multilingual corpus (**Latin Natural Language**).

Across all settings, the perturbation length is uniformly sampled between 100 and 300 tokens to align with the main experiments. Comprehensive implementation and generation details for each method are provided in Appendix F.

7.2 How LLMs Perceive Prompt-Space Noise

To understand how prompt-space perturbations influence generation, we investigate the intrinsic properties of the perturbation texts, specifically focusing on how well they align with the language model's distribution.

For each method, we generate 500 perturbation sequences (200 tokens each) and compute their perplexity under Qwen3-1.7B-Base. We also report the perplexity of the original question texts as an in-distribution reference. The resulting distributions (Figure 5) naturally partition the methods into three regimes based on their mean perplexity:

- **Near-natural perturbations** (mean < 100): *Lorem Ipsum*, *Filtered Latin Natural Language*, *Latin Unigram*, *English Unigram*, and *Latin 3-Gram* all sit within roughly an order of magnitude of the *Question Text* reference. This suggests they behave as structurally plausible noise.
- **Moderately out-of-distribution perturbations** (mean $\approx 10^2$–10^3): *Random ASCII* and *Random Fake English* drift significantly further from the natural language manifold.
- **Severely out-of-distribution perturbations**: *Random Token*'s mean value explodes to 4.6×10^5, reflecting a complete linguistic collapse.

Beyond the mean perplexity, the dispersion of these distributions varies substantially. *Lorem Ip-*

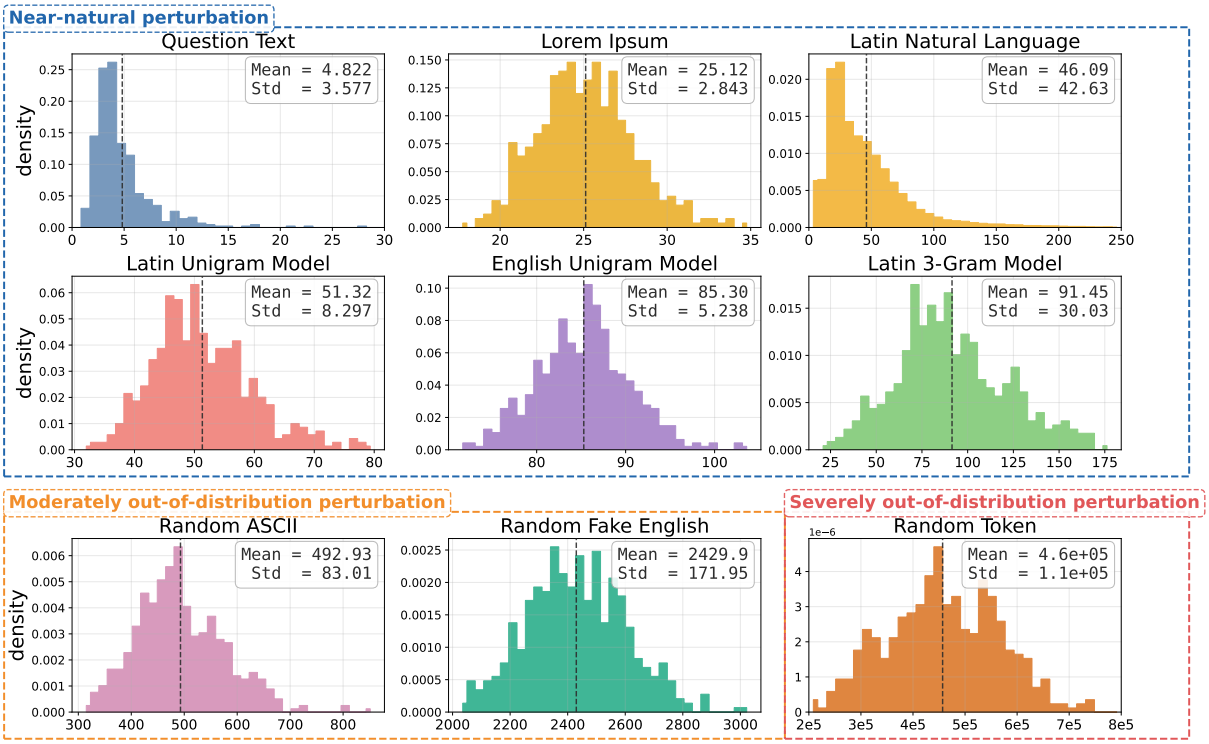


Figure 5: Perplexity distributions of randomly generated sequences from each perturbation type, with the mean and standard deviation reported in each subplot. "Question Text" from the 500-question evaluation set serves as an English natural language reference. Perturbations are grouped by their perplexity computed on Qwen3-1.7B-Base. Perturbations in the first two rows show a near-natural language property, yielding better final performance, while those in the last row are out of the model distribution, resulting in worse performance as shown in Table 2.

sum exhibits the lowest mean among synthetic perturbations and is tightly concentrated (std 2.84), providing a consistent, controlled distributional shift. In contrast, *Filtered Latin Natural Language* presents a long-tailed distribution (std 42.63). Such uneven perturbation strength risks inducing high within-batch variance during RL training. Therefore, we derive a strictly filtered variant for subsequent RL experiments, retaining only sequences with perplexities between 20 and 30 (yielding ∼38K sequences).

Additionally, further analysis of token-level entropy and sentence representations confirms that near-natural perturbations preserve the model's semantic comprehension of the original question, whereas severe noise distorts it (Appendix G).

Takeaway: Random sequences generated by sampling from a limited word pool have lower perplexity, showing a near-natural property.

7.3 The Recipe for an Effective Perturbation

Building on the insights, we evaluate these prompt-space perturbation methods in RL training. We evaluate resampling with the naive prompt and resampling at a higher temperature as baselines.

The results in Table 2 show varying degrees of performance. While **LoPE** proves highly effective, it is not the sole solution. Notably, the *Filtered Latin Natural Language* and *Latin Unigram Model* both achieve average scores exceeding 39.6. Recalling Figure 5, these three top-performing methods share two defining characteristics: they utilize Latin vocabulary and maintain the lowest perplexity among all synthetic perturbations.

We conjecture the reason for their success as follows. They provide sufficient perturbation to drive the rollout policy away from the naive one, encouraging model exploration. Meanwhile, their relatively low perplexity avoids introducing excessive noise, which would degrade reasoning quality.

Furthermore, the *English Unigram Model* underperforms the *Latin 3-Gram Model* despite possessing a slightly lower average perplexity. This implies that English-based perturbations are prone to interfering with the model's internal English reasoning context, hindering overall performance.

Model & Method	MATH-500	GSM8K	AMC	AIME24	AIME25	Avg.
Qwen3-1.7B-Base	63.40	76.92	26.87	5.33	2.00	34.90
+ GRPO	67.60	82.03	27.61	7.92	5.36	38.10
+ *Resample w/o perturbation*						
w/ Naive Prompt	67.00	82.18	28.36	8.70	4.58	38.16
w/ Naive Prompt (Temp=1.2)	64.40	82.87	31.34	8.65	4.48	38.35
+ *Resample w/ perturbation* + *Training Signal Shaping*						
w/ **LoPE**	68.80	82.94	**32.84**	8.80	5.73	39.82
w/ Random Fake English	65.80	81.96	32.09	7.50	5.42	38.55
w/ Random ASCII	66.20	82.94	28.36	8.12	5.32	38.19
w/ Random Token	64.20	81.50	29.85	8.08	4.63	37.65
w/ Filtered Latin Natural Language	68.80	82.71	**32.84**	**9.32**	5.57	**39.85**
w/ Latin Unigram Model	**69.40**	**83.32**	32.09	7.19	**6.35**	39.67
w/ Latin 3-Gram Model	68.80	81.88	29.85	7.92	5.93	38.88
w/ English Unigram Model	67.00	**83.32**	28.36	8.49	5.42	38.52

Table 2: Comparison of different prompt space perturbation methods. The three methods with the smallest perplexity value (**LoPE**, *Filtered Latin Natural Language*, and *Latin Unigram Model*) achieve the best performance. *Random Token* with the highest perplexity even harms the training. This suggests that mild perturbations are sufficient to drive improvement while avoiding detrimental effects of excessive noise.

Takeaway: The most effective prompt space perturbations share these characteristics: (i) pseudo-Latin vocabularies to prevent interference with the English reasoning context, and (ii) maintaining low perplexity to ensure high-quality rollouts.

8 Related Work

Zero-Advantage Recovery in RLVR. GRPO-style reinforcement learning (Shao et al., 2024; Guo et al., 2025; Yu et al., 2025) improves LLM reasoning but suffers from zero-reward signals on hard prompts. Recent efforts focus on improving rollout efficiency and recovering useful training signals, such as adaptive budget allocation, targeted exploration, scaffolded hints, and off-policy guidance (Zhang et al., 2026b; Le et al., 2026; Bamba et al., 2025; Zhang et al., 2026a; Yan et al., 2025; Zhao et al., 2026). Another line of work explores modifying prompt patterns or prefixes during training, such as prompt selection, augmentation, and prefix-level guidance (Wang et al., 2026; Lu et al., 2026; Mundada et al., 2026).

Context-level Perturbation. While in-context learning with logit-level arithmetics explicitly alters output distribution (Huang et al., 2026a), a group of work shows that prompt context can also implicitly influence the model's generation behavior, where small changes can induce substantial shifts (Xie et al., 2022; von Oswald et al., 2023; An et al., 2023). Empirical studies on prompting further demonstrate that reasoning performance is highly sensitive to prompt format, demonstrations, and instruction structure (Wei et al., 2023; Wang et al., 2023; Huang et al., 2025; Zhou et al., 2023; Yang et al., 2024b; Huang et al., 2026b). More recent work highlights that adding synthetic or meaningless tokens can shift model activations and alter reasoning behavior (Shao et al., 2026; Shi et al., 2025; Gan and Isola, 2026).

9 Conclusion

This paper introduces Lorem Perturbation for Exploration (**LoPE**), a simple yet effective method that prepends a random *Lorem Ipsum* sequence to the prompt. By introducing a controllable prompt-space perturbation, **LoPE** enables LLMs to explore alternative reasoning trajectories, effectively recovering training signals on previously failed questions. Integrating **LoPE** into GRPO training significantly boosts question-level success rates, data utilization, and overall reasoning performance.

Furthermore, our systematic analysis of various prompt-space noises reveals that **LoPE** is not an isolated success. The most effective perturbations share two defining characteristics: a Latin-based vocabulary and relatively low perplexity. Collectively, these findings underscore the immense potential of controlled prompt-space perturbation in broadening exploration for LLM reasoning.

Limitations

This work explores prompt-space perturbations using randomly generated sequences. As these sequences are generated automatically, we cannot guarantee that they are entirely free of potentially toxic, biased, or inappropriate words or expressions.

In addition, we observe that when the perturbation is excessively strong, the model may produce incoherent or nonsensical outputs that, in some cases, include undesirable or harmful content. This highlights a limitation of our approach: increased exploration via perturbation may come at the cost of reduced controllability over model outputs.

To mitigate these risks, our method emphasizes controlled perturbation, and our experiments suggest that moderate, language-like perturbations are less likely to negatively affect model behavior. Nevertheless, ensuring the safety and robustness of such perturbations remains an open challenge.

In future work, we will systematically study how different types and strengths of perturbations influence model behavior, with the goal of minimizing the risk of generating toxic or harmful content while preserving the benefits of improved exploration.

References

Ruopeng An, Yuyi Yang, Fan Yang, and Shanshan Wang. 2023. Use prompt to differentiate text generated by chatgpt and humans. *Machine Learning with Applications*, 14:100497.

Udbhav Bamba, Minghao Fang, Yifan Yu, Haizhong Zheng, and Fan Lai. 2025. Xrpo: Pushing the limits of grpo with targeted exploration and exploitation. *Preprint*, arXiv:2510.06672.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. 2023. Why can gpt learn in-context? language models implicitly perform gradient descent as meta-optimizers. *Preprint*, arXiv:2212.10559.

Yulu Gan and Phillip Isola. 2026. Neural thickets: Diverse task experts are dense around pretrained weights. *Preprint*, arXiv:2603.12228.

Adrian Goldwaser, Michael Munn, Javier Gonzalvo, and Benoit Dherin. 2025. Equivalence of context and parameter updates in modern transformer blocks. *Preprint*, arXiv:2511.17864.

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

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *Preprint*, arXiv:2103.03874.

Chengsong Huang, Langlin Huang, and Jiaxin Huang. 2026a. Divide, reweight, and conquer: A logit arithmetic approach for in-context learning. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 229–249, Rabat, Morocco. Association for Computational Linguistics.

Chengsong Huang, Langlin Huang, Jixuan Leng, Jiacheng Liu, and Jiaxin Huang. 2025. Efficient test-time scaling via self-calibration. *arXiv preprint arXiv:2503.00031*.

Chengsong Huang, Langlin Huang, Jixuan Leng, Jiacheng Liu, and Jiaxin Huang. 2026b. CaTS: Calibrated test-time scaling for efficient LLM reasoning. In *The Fourteenth International Conference on Learning Representations*.

Jarry Shaw. 2024. Lorem ipsum generator.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. Fasttext.zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

Thanh-Long V. Le, Myeongho Jeon, Kim Vu, Viet Lai, and Eunho Yang. 2026. No prompt left behind: Exploiting zero-variance prompts in llm reinforcement learning via entropy-guided advantage shaping. *Preprint*, arXiv:2509.21880.

Ziniu Li, Congliang Chen, Tianyun Yang, Tian Ding, Ruoyu Sun, Ge Zhang, Wenhao Huang, and Zhi-Quan Luo. 2025. Knapsack rl: Unlocking exploration of llms via optimizing budget allocation. *Preprint*, arXiv:2509.25849.

Mengqi Liao, Xiangyu Xi, Ruinian Chen, Jia Leng, Yangen Hu, Ke Zeng, Shuai Liu, and Huaiyu Wan. 2025. Enhancing efficiency and exploration in reinforcement learning for llms. *Preprint*, arXiv:2505.18573.

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. *Preprint*, arXiv:2503.20783.

Wenquan Lu, Hai Huang, and Randall Balestriero. 2026. Prompt augmentation scales up grpo training on mathematical reasoning. *Preprint*, arXiv:2602.03190.

Gagan Mundada, Zihan Huang, Rohan Surana, Sheldon Yu, Jennifer Yuntong Zhang, Xintong Li, Tong Yu, Lina Yao, Jingbo Shang, Julian McAuley, and Junda Wu. 2026. Ws-grpo: Weakly-supervised group-relative policy optimization for rollout-efficient reasoning. *Preprint*, arXiv:2602.17025.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, Yulia Tsvetkov, Hannaneh Hajishirzi, Pang Wei Koh, and Luke Zettlemoyer. 2026. Spurious rewards: Rethinking training signals in rlvr. *Preprint*, arXiv:2506.10947.

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

Zeru Shi, Yingjia Wan, Zhenting Wang, Qifan Wang, Fan Yang, Elisa Kreiss, and Ruixiang Tang. 2025. Meaningless tokens, meaningful gains: How activation shifts enhance llm reasoning. *Preprint*, arXiv:2510.01032.

ModelScope Team. 2024. EvalScope: Evaluation framework for large models.

Qwen Team. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Johannes von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. 2023. Transformers learn in-context by gradient descent. *Preprint*, arXiv:2212.07677.

Hanbin Wang, Jingwei Song, Jinpeng Li, Fei Mi, and Lifeng Shang. 2026. Group pattern selection optimization: Let lrms pick the right pattern for reasoning. *Preprint*, arXiv:2601.07238.

Jiakang Wang, Runze Liu, Lei Lin, Wenping Hu, Xiu Li, Fuzheng Zhang, Guorui Zhou, and Kun Gai. 2025. Aspo: Asymmetric importance sampling policy optimization. *Preprint*, arXiv:2510.06062.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. *Preprint*, arXiv:2203.11171.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An explanation of in-context learning as implicit bayesian inference. *Preprint*, arXiv:2111.02080.

Wei Xiong, Chenlu Ye, Baohao Liao, Hanze Dong, Xinxing Xu, Christof Monz, Jiang Bian, Nan Jiang, and Tong Zhang. 2025. Reinforce-ada: An adaptive sampling framework under non-linear rl objectives. *Preprint*, arXiv:2510.04996.

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. *Preprint*, arXiv:2504.14945.

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. 2024a. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *Preprint*, arXiv:2409.12122.

Yuyi Yang, Charles Alba, Chenyu Wang, Xi Wang, Jami Anderson, and Ruopeng An. 2024b. Gpt models can perform thematic analysis in public health studies, akin to qualitative researchers. *Journal of Social Computing*, 5(4):293–312.

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

Xichen Zhang, Sitong Wu, Yinghao Zhu, Haoru Tan, Shaozuo Yu, Ziyi He, and Jiaya Jia. 2026a. Scaf-grpo: Scaffolded group relative policy optimization for enhancing llm reasoning. *Preprint*, arXiv:2510.19807.

Zhi Zhang, Zhen Han, Costas Mavromatis, Qi Zhu, Yunyi Zhang, Sheng Guan, Dingmin Wang, Xiong Zhou, Shuai Wang, Soji Adeshina, Vassilis Ioannidis, and Huzefa Rangwala. 2026b. Train less, learn more: Adaptive efficient rollout optimization for group-based reinforcement learning. *Preprint*, arXiv:2602.14338.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. 2026. Self-distilled reasoner: On-policy self-distillation for large language models. *Preprint*, arXiv:2601.18734.

Yaowei Zheng, Junting Lu, Shenzhi Wang, Zhangchi Feng, Dongdong Kuang, Yuwen Xiong, and Zhang

Richong. 2025. Easyr1: An efficient, scalable, multi-modality rl training framework. https://github.com/hiyouga/EasyR1.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. *Preprint*, arXiv:2205.10625.

A Example of Lorem Ipsum Prompt

Lorem Ipsum Prompt

System
in laboris velit ex est ex est exercitation sint ex enim ut incididunt tempor cupidatat nisi occaecat deserunt laboris quis voluptate qui consequat proident minim pariatur velit est aliquip do labore ut dolor et ullamco proident in non culpa est amet ipsum officia velit exercitation aute consectetur aliquip ullamco laboris laborum deserunt quis enim consectetur ea veniam officia commodo cupidatat culpa fugiat consequat eu enim mollit commodo enim eiusmod deserunt laborum ullamco minim irure consectetur quis ut nisi cillum adipiscing proident culpa id cupidatat ad amet sunt qui adipiscing mollit proident dolor pariatur minim eu quis ad excepteur nulla reprehenderit ea enim
Please reason step by step, and put your final answer within \boxed{}.
User
Given $abc \neq 0$, and $a + b + c = 0$. Then the value of the algebraic expression

$$\frac{a^2}{bc} + \frac{b^2}{ac} + \frac{c^2}{ab}$$

is:

(A) 3

(B) 2

(C) 1

(D) 0

You FIRST think about the reasoning process as an internal monologue and then provide the final answer. The reasoning process MUST BE enclosed within <think> </think> tags. The final answer MUST BE put in \boxed{}.
Assistant

B Preliminary Experiment Details

In the preliminary experiment described in Section 3, we evaluate the three configurations on 500 randomly sampled questions from the Openr1-Math-46k-8192 dataset (Yan et al., 2025), using the Qwen3-1.7B-Base model (Team, 2025).

The initial evaluation is performed on the entire set of 500 questions (Figure 2 left) with a sampling budget of 8 rollouts per question. A question is classified as a failure if all 8 sampled rollouts are incorrect; otherwise, it is considered successful. Based on this initial evaluation, we filter out the questions where the base setting fails to form the hard subset, which comprises 352 questions. The subsequent evaluation is then conducted exclusively on this hard subset (Figure 2 right), maintaining the same budget of 8 rollouts per question.

We utilize Venn diagrams to illustrate the distribution of successfully resolved questions across different configurations. In these diagrams, the area of each region is proportional to the number of successful questions. The overlapping intersections of the shapes represent the queries jointly resolved by multiple methods, while the disjoint peripheral regions denote the unique successes achieved by a specific method. Notably, the Lorem-perturbed prompt exhibits a substantially larger unique area in both evaluations, demonstrating its efficacy in broadening the reasoning exploration of LLMs.

C Main Experimental Details

Training Configurations. Our implementation is based on EasyR1 (Zheng et al., 2025). For Qwen2.5-MATH-7B, whose original context length is 4096, we follow Yan et al. (2025) to extend the context

window to 16384. During training, the maximum input and response lengths are set to 2048 and 8192 tokens, respectively. For policy optimization, the group size is set to $G = 8$, and the resampling size is $G' = 24$. All rollouts are sampled with a default temperature of 1.0. Following Yu et al. (2025) and Liu et al. (2025), we exclude the KL regularization term from the standard GRPO objective. Experiments on the 1.7B and 4B models are conducted on 4 $\times$ 80GB A100 GPUs, while those on the 7B model utilize 8 $\times$ 80GB A100 GPUs.

Evaluation Details. We use EvalScope (Team, 2024) for all benchmark evaluations, setting the sampling temperature to 0.6 and top-p to 0.95. We report Acc@1 for MATH-500, GSM8K, and AMC. For the more challenging AIME 2024 and AIME 2025 benchmarks, we report Mean@32.

Perturbation Generation. The Lorem Ipsum text used for prompt-space perturbation is generated using the `python-lorem` package[1]. The sequence length for each perturbation is uniformly sampled between 100 and 300 tokens. Empirically, we append a short **boundary instruction** to the end of each perturbation sequence: *"\nPlease reason step by step, and put your final answer within \boxed."*

It is crucial to clarify that this instruction is designed strictly as a structural delimiter to separate the noise sequence from the actual question. This simple design effectively prevents the perturbation from negatively interfering with the model and mitigating the generation of corrupted outputs (e.g., random symbols or characters). To rule out the concern that this boundary instruction might provide an unfair reasoning advantage, we emphasize that the default system prompt already contains a more comprehensive and stringent directive: *"You FIRST think about the reasoning process as an internal monologue and then provide the final answer. The reasoning process MUST BE enclosed within <think> </think> tags. The final answer MUST BE put in \boxed."* Therefore, the boundary instruction introduces no new reasoning priors. It merely reinstates the formatting requirements to ensure normal generation behavior after the injection of noise.

D Training-Time Resample Accuracy for Qwen3-4B-Base and Qwen2.5-Math-7B

We present the training-time resample success rate and accuracy of Qwen3-4B-Base and Qwen2.5-Math-7B in Figure 6 and Figure 7, where **LoPE** consistently achieves a significantly higher question-level success rate.

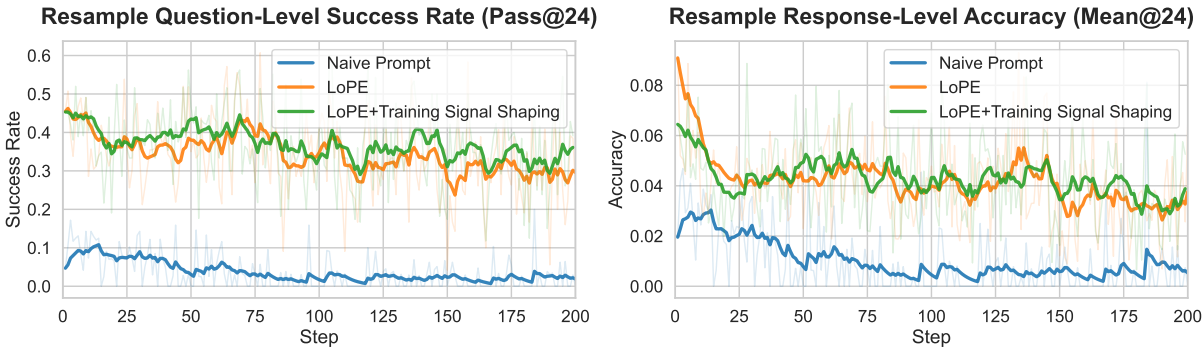


Figure 6: Resample success rate and accuracy during Qwen3-4B-Base training.

E The Effectiveness of Training Signal Shaping on Training

E.1 Policy Shaping

Gradient Analysis of the Resampling Objective. To better understand the role of the reward shaping function $f(\cdot)$ in our setting, we follow the derivation in (Yan et al., 2025) to analyze the gradient of the resampling part of the training objective in Eq. (6). Notably, unlike their formulation, we relax the

[1] https://pypi.org/project/python-lorem/

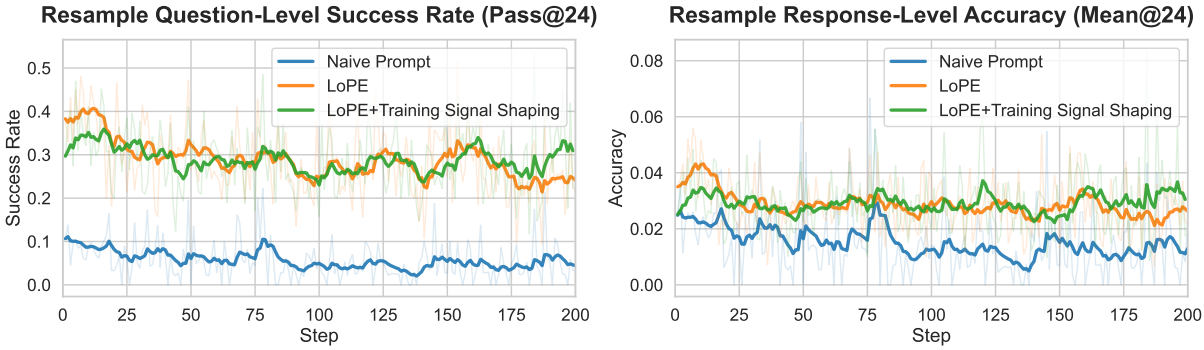


Figure 7: Resample success rate and accuracy during Qwen2.5-MATH-7B training.

assumption that $\pi_{\theta_{\text{old}}} = 1$. The objective is given by:

$$J_{\text{resample}}(\theta) = \mathbb{E}_{q,\,\{o_i\}\sim\pi_{\theta_{\text{old}}}} \sum_{i=1}^{N_s} \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \Big[f(\rho_{i,t})\, \hat{A}_i \Big], \tag{7}$$

where $\rho_{i,t} = \pi_\theta / \pi_{\theta_{\text{old}}}$ is the importance sampling ratio. For brevity, we omit the conditioning variables in the following derivation and denote $\pi_\theta := \pi_\theta(o_{i,t} \mid q, o_{i,<t})$ and $\pi_{\theta_{\text{old}}} := \pi_{\theta_{\text{old}}}(o_{i,t} \mid \delta \oplus p, q, o_{i,<t})$.

Gradient with respect to θ. Applying the logarithmic derivative $\nabla_\theta \pi_\theta = \pi_\theta \nabla_\theta \log \pi_\theta$, the derivative of $f(\rho_{i,t})$ w.r.t. θ is $f'(\rho_{i,t})\, \rho_{i,t}\, \nabla_\theta \log \pi_\theta$. Thus, the gradient of the resampling objective is:

$$\nabla_\theta J_{\text{resample}}(\theta) = \mathbb{E}_{q,\,\{o_i\}\sim\pi_{\theta_{\text{old}}}} \left[\sum_{i,t} \frac{1}{|o_i|} \underbrace{f'(\rho_{i,t})}_{\text{shaped weight}} \rho_{i,t}\, \nabla_\theta \log \pi_\theta \cdot \hat{A}_i \right]. \tag{8}$$

The term $f'(\rho_{i,t})$ acts as a weighting function of the gradient. When $f(x) = x$, we have $f'(x) = 1$, degenerating to the vanilla importance sampling gradient.

Per-token Logit Gradient and Upper Bound. To analyze how the resampling objective updates the model's predictive distribution at the token level, we project the parameter gradient $\nabla_\theta \log \pi_\theta(o_{i,t})$ onto an individual output logit $M_\theta(\tau)$, where τ ranges over the action space at position t. Since the policy is parameterized as a softmax over logits,

$$\pi_\theta(o_{i,t}) = \frac{\exp\big(M_\theta(o_{i,t})\big)}{\sum_{\tau'} \exp\big(M_\theta(\tau')\big)}, \tag{9}$$

taking logarithms gives $\log \pi_\theta(o_{i,t}) = M_\theta(o_{i,t}) - \log \sum_{\tau'} \exp\big(M_\theta(\tau')\big)$, and differentiating with respect to $M_\theta(\tau)$ yields the standard softmax Jacobian identity:

$$\frac{\partial \log \pi_\theta(o_{i,t})}{\partial M_\theta(\tau)} = \mathbb{1}[\tau = o_{i,t}] - \pi_\theta(\tau). \tag{10}$$

By taking the derivative of Eq. (8) with respect to $M_\theta(\tau)$ and applying Eq. (10), we obtain the per-token logit gradient and its upper bound:

$$\begin{aligned} \frac{\partial J_{\text{resample}}(\theta)}{\partial M_\theta(\tau)} &= \mathbb{E}_{q,\,\{o_i\}\sim\pi_{\theta_{\text{old}}}} \Big[f'(\rho_{i,t})\, \rho_{i,t}\, (\mathbb{1}[\tau = o_{i,t}] - \pi_\theta(\tau)) \cdot \hat{A}_i \Big] \\ \Rightarrow \left| \frac{\partial J_{\text{resample}}(\theta)}{\partial M_\theta(o_{i,t})} \right| &\le \mathbb{E}_{q,\,\{o_i\}\sim\pi_{\theta_{\text{old}}}} \Big[\big| f'(\rho_{i,t}) \big|\, \rho_{i,t}\, (1 - \pi_\theta) \cdot \big| \hat{A}_i \big| \Big], \end{aligned} \tag{11}$$

where the upper bound corresponds to the identity case $\tau = o_{i,t}$, yielding $|\mathbb{1}[\tau = o_{i,t}] - \pi_\theta(\tau)| = 1 - \pi_\theta$. This case captures the dominant gradient signal that elevates the logit of the resampled token itself.

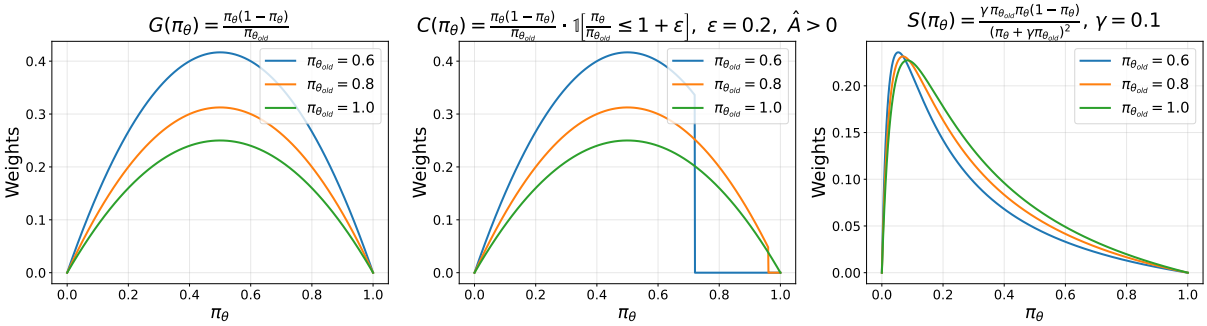


Figure 8: Per-token gradient weight under three formulations, plotted as a function of π_θ. **Left:** Vanilla gradient bound $G(\pi_\theta)$. **Middle:** GRPO-clipped gradient $C(\pi_\theta)$ under positive advantage ($\hat{A} > 0$). Gradients are truncated to zero when $\rho_{i,t} > 1 + \epsilon$. **Right:** Policy-shaped gradient $S(\pi_\theta)$. The peak is relocated to the low-probability regime, with bounded peak value $1/4$.

From Eq. (11), under the vanilla setting $f(x) = x$, the gradient scale is bounded by $\rho_{i,t}(1 - \pi_\theta) = \pi_\theta(1 - \pi_\theta)/\pi_{\theta_{\text{old}}}$.

To visualize this bound and motivate our shaping choice, Figure 8 plots the per-token gradient weight as a function of π_θ across several $\pi_{\theta_{\text{old}}}$ values, under the situation of a positive advantage ($\hat{A} > 0$). The left panel illustrates the unclipped vanilla gradient bound $G(\pi_\theta) = \pi_\theta(1 - \pi_\theta)/\pi_{\theta_{\text{old}}}$, while the middle panel displays its GRPO-clipped counterpart $C(\pi_\theta)$. In $C(\pi_\theta)$, the gradient is additionally truncated to zero when $\rho_{i,t} > 1 + \epsilon$, which corresponds to the active branch in the GRPO objective for $\hat{A} > 0$.

As illustrated in Figure 8, both the vanilla and GRPO-clipped gradient bounds suffer from two fundamental limitations:

(i) Vanishing gradient at low π_θ. The bound decays to zero as $\pi_\theta \to 0$, yielding weak learning signals for unfamiliar tokens where the current policy is highly uncertain. Crucially, the distribution discrepancy between the resampled trajectories $\{o_i\} \sim \pi_{\theta_{\text{old}}}$ and the training policy π_θ systematically drives π_θ to small values, suppressing gradients exactly where off-policy guidance is most needed. GRPO clipping fails to address this issue, as it only operates on the $\rho_{i,t} > 1 + \epsilon$ region.

(ii) Inappropriate handling of low $\pi_{\theta_{\text{old}}}$ tokens. The peak value of the vanilla bound is $1/(4\pi_{\theta_{\text{old}}})$, which diverges as $\pi_{\theta_{\text{old}}} \to 0$. This induces excessive gradient magnitudes on rare tokens of the resampling policy, causing training instability. While GRPO clipping avoids this by hard-truncating the gradients to zero, it entirely sacrifices the learning signals associated with these tokens.

To overcome these limitations, we adopt the policy shaping function $f(x) = x/(x + \gamma)$ proposed by (Yan et al., 2025) to reshape the gradients.

Specialization to $f(x) = x/(x + \gamma)$. The derivative of the shaping function is $f'(x) = \gamma/(x + \gamma)^2$. Substituting this into the bound in Eq. (11) yields:

$$\left| \frac{\partial J_{\text{resample}}(\theta)}{\partial M_\theta(o_{i,t})} \right| \le \mathbb{E}\left[\frac{\gamma\, \pi_{\theta_{\text{old}}}\, \pi_\theta(1 - \pi_\theta)}{(\pi_\theta + \gamma\, \pi_{\theta_{\text{old}}})^2} \cdot |\hat{A}_i| \right]. \tag{12}$$

Treating the integrand of Eq. (12) as a function of π_θ for a fixed $\pi_{\theta_{\text{old}}}$, it attains its maximum at:

$$\pi_\theta^\star = \frac{\gamma\, \pi_{\theta_{\text{old}}}}{1 + 2\gamma\, \pi_{\theta_{\text{old}}}}, \qquad \text{with a peak value of} \quad \frac{1}{4\big(1 + \gamma\, \pi_{\theta_{\text{old}}}\big)}. \tag{13}$$

Equation (13) formalizes two complementary properties of the shaping function, which are visually corroborated in Figure 8 (right).

(i) Low-probability emphasis. The peak location shifts from $\pi_\theta = 1/2$ (the vanilla case) into the low-probability regime at $\pi_\theta^\star \approx \gamma\, \pi_{\theta_{\text{old}}}$. Consequently, the shaping function amplifies the learning signal for the unfamiliar yet highly rewarded tokens that off-policy resampling aims to introduce.

(ii) Bounded and stable peak value. The peak value $1/[4(1 + \gamma\, \pi_{\theta_{\text{old}}})]$ is strictly bounded by $1/4$ for all $\pi_{\theta_{\text{old}}} \in [0, 1]$ and remains stable across different π_θ and $\pi_{\theta_{old}}$ values. In contrast, the vanilla bound scales as $1/(4\pi_{\theta_{\text{old}}})$ and can grow unboundedly as $\pi_{\theta_{\text{old}}} \to 0$.

By reshaping the learning signal toward under-confident tokens and stabilizing the gradient magnitude across the full spectrum of probabilities, policy shaping effectively reweights the parameter updates. It assigns greater importance to low-probability yet effective actions while gracefully attenuating updates for tokens the model has already mastered.

E.2 Advantage Shaping

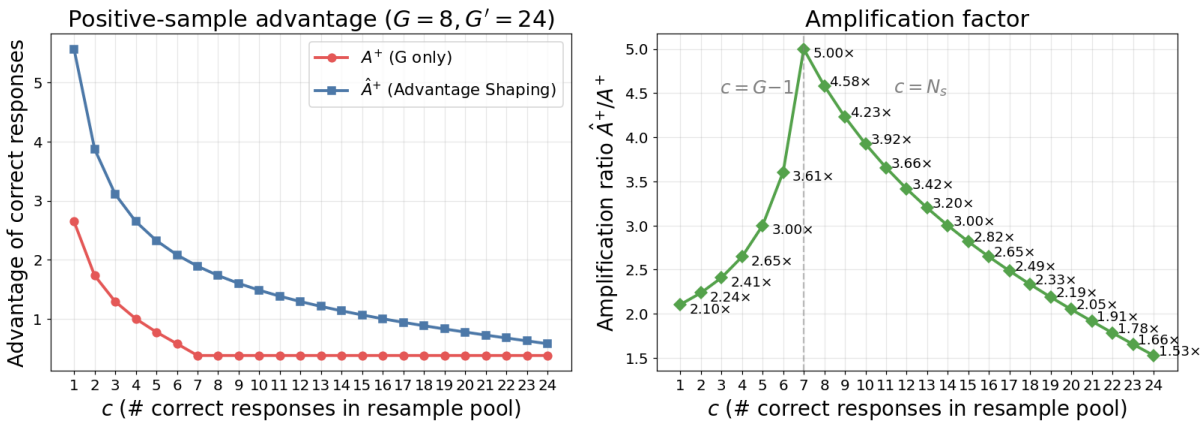


Figure 9: Comparison of advantages for positive responses before and after advantage shaping. **Left:** The absolute advantage values of the vanilla advantage (A^+) and shaped advantage ($\hat{A}^+$), as a function of the number of correct responses c. **Right:** The amplification factor ($\hat{A}^+/A^+$) achieved by applying advantage shaping.

Quantitative Effect of Advantage Shaping. Since the advantage term acts as a constant multiplier in the training objective (Eq. 6) and the gradient (Eq. 8), its effect is independent of other variables. Therefore, we quantitatively analyze the advantage term in isolation.

We follow the notations in Sections 4 and 5: the rollout size for the first-time sampling and the resampling is G and G', where the number of correct responses is 0 and c, respectively. The gradient update is performed on a regrouped set of G responses, consisting of $N_s = \min(c, G-1)$ correct resampled responses and $G - N_s$ incorrect first-time sampling responses. The vanilla **LoPE** calculates the advantage, denoted as A, solely within the G selected rollouts. In contrast, advantage shaping computes the advantage, denoted as $\hat{A}$, over the entire pool of $G + G'$ rollouts.

In these two scenarios, the means and standard deviations of the rewards are $\mu = N_s/G$, $\sigma = \sqrt{(G-N_s)N_s}/G$ and $\hat{\mu} = c/(G+G')$, $\hat{\sigma} = \sqrt{(G+G'-c)c}/(G+G')$, respectively. Substituting these into Eq. 3 and Eq. 5, we obtain the advantages for the positive samples:

$$A^+ = \sqrt{\frac{G - N_s}{N_s}}, \qquad \hat{A}^+ = \sqrt{\frac{(G+G') - c}{c}}. \tag{14}$$

Figure 9 visualizes these quantities and their ratio using our actual sampling budget ($G = 8, G' = 24$) across the regime of $c \in [1, G']$. Since resampling is triggered exclusively on hard questions where the initial G samples all fail, c **is typically smaller than** G **in practice** ($c < G$). With in this range, the amplification factor $\hat{A}^+/A^+$ ranges from $2.10\times$ at $c = 1$ to a peak of $5.00\times$ at $c = G - 1 = 7$.

According to Eq. (8), the amplification of advantage directly applies to the gradient weight. Therefore, advantage shaping effectively assigns a larger training weight to the rare correct trajectories that drive learning on hard questions.

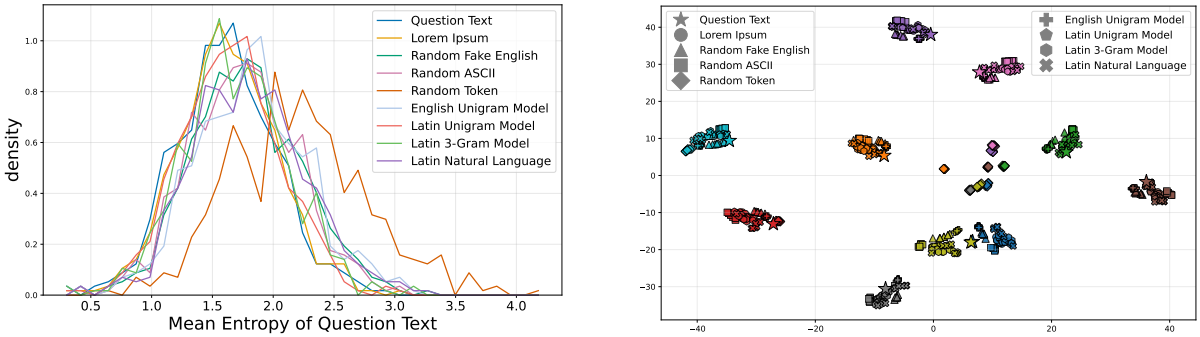
F Details of Alternative Prompt-Space Perturbations

In Section 7, we introduce several methods for generating prompt-space perturbations to analyze the critical properties of effective noise. The detailed implementation for each method is described below:

- **Random Fake English.** This setting uses the `Faker` package[2] to generate fake English sentences, where each word is independently sampled from a commonly used English word pool[3]. In addition, it applies initial capitalization and inserts random punctuation to mimic natural sentence structures.
- **Random ASCII.** This setting uniformly samples printable ASCII characters to form random sequences.
- **Random Tokens.** This setting uniformly samples tokens from the model's vocabulary to form random sequences. Special tokens are explicitly excluded to prevent functional errors.
- **English Unigram Model.** In this setting, we follow the same random generation procedure as **LoPE**, but replace the candidate word pool with the top-50 most frequent words extracted from the English subset of the C4 multilingual corpus (Raffel et al., 2020)[4], which is widely used for pretraining. Non-English text is filtered out using `langid`[5] and `fastText` (Joulin et al., 2016). Words are sampled uniformly from the derived word pool.
- **Latin Unigram Model.** This setting is similar to the **English Unigram Model**, but uses the top-50 most frequent Latin words from the Latin subset of the C4 corpus as the candidate word pool.
- **Latin 3-Gram Model.** In this setting, we explore locally coherent random sequences generated by a 3-gram language model. Concretely, we train a 3-gram language model using `markovify`[6] on the Latin subset of the C4 corpus, and utilize this model for random sequence generation during training.
- **Latin Natural Language.** In this setting, we use authentic natural Latin text as the prompt-space perturbation. Specifically, we adopt the Latin subset of the C4 corpus. We filter out non-Latin sentences using `langid` and `fastText`, and additionally remove any sentences containing the canonical Lorem Ipsum incipit (*"lorem ipsum dolor sit amet, consectetur adipiscing elit"*). After deduplication and retaining only sequences within the 100–300 token range, we obtain a clean corpus of approximately 65K sequences.

G Impact of Random Perturbations on Question Comprehension

In Section 7.2, we demonstrated that random sequences introduce varying degrees of perplexity shift into the prompt space. Here, we further investigate the extent to which these perturbations disrupt the model's semantic comprehension of the input question itself.



(a) Entropy distribution of 500 question prompts. Most perturbations preserve a distribution close to the Question Text. The distribution of Random Token is clearly right-shifted, indicating a corrupted understanding of the question.

(b) Question representation visualization via t-SNE under different perturbations. Each color corresponds to one question (8 samples per perturbation). Random Token perturbations drift far from the original meaning.

Figure 10: The influence of various prompt space perturbations on question comprehension.

We analyze both token-level entropy and sentence-level representations. Figure 10a presents the entropy distribution across 500 question prompts, characterized by the average entropy of their constituent tokens.

[2] https://pypi.org/project/Faker/
[3] https://github.com/joke2k/faker/blob/master/faker/providers/lorem/en_US/__init__.py
[4] https://huggingface.co/datasets/allenai/c4/viewer/la
[5] https://pypi.org/project/langid/
[6] https://pypi.org/project/markovify/

For most perturbations, the entropy distributions largely overlap with the clean *Question Text*, indicating minimal disruption to comprehension. However, the *Random Token* perturbation exhibits a significant rightward shift, signaling higher uncertainty.

To evaluate semantic drift, we select 10 questions consistently solved across all methods and compute their sentence representations (mean pooling of final-layer hidden states across question tokens), visualized via t-SNE in Figure 10b. Representations under near-natural or moderately out-of-distribution perturbations cluster tightly with the clean *Question Text*, confirming stable semantic understanding. In contrast, *Random Token* representations drift substantially, suggesting that severe noise heavily distorts the interpretation of the input.

Collectively, excessively high-perplexity perturbations corrupt the model's understanding of the core question, impeding correct solution discovery during RL training.

Takeaway: Near-natural perturbation (low perplexity) successfully shifts the model's generation distribution without corrupting its semantic understanding of the core question. Conversely, excessively noisy perturbation destroys the input representation.