PREFIXRLVR: MITIGATING CATASTROPHIC REPETITION IN LONG-CONTEXT RAG

Anonymous authors
Paper under double-blind review

ABSTRACT

Modern end-to-end agentic workflows rely on robust long-context generation for multi-step autonomous tasks. However, sustained long-form outputs make LLMs prone to catastrophic repetition, which is particularly evident in open-source models with smaller parameter sizes, resulting in wasted compute and stalled downstream pipelines. Prior work mitigates repetition through decoding strategies, attention interventions, or training-based methods without assessing their impact on general capabilities, leaving the risk of catastrophic forgetting unaddressed. In this work, we introduce DE-RP, accompanied by a character-level detector that precisely localizes the onset of catastrophic repetition. Through systematic analyses across a range of open-source models, we find that catastrophic repetition is not merely a random decoding failure but can be induced by specific trigger prefixes that are shared across models of different series and scales. Hidden-state analysis reveals a repetition-associated signal near the prefix boundary that is shared across models. Building on this insight, we propose PREFIXRLVR, a reinforcement-learning-based approach that combines training from these high-risk prefixes with length-aware advantage scaling. Experimental results show that PREFIXRLVR reduces catastrophic repetition rates by approximately 49–$330\times$ across the evaluated backbones while preserving general model capabilities. Further long-context evaluations demonstrate consistent generalization gains, suggesting that PREFIXRLVR offers an effective path toward more reliable long-context generation. Code and data are released at: `https://anonymous.4open.science/r/PrefixRLVR`.

1 INTRODUCTION

Large Language Models (LLMs) remain susceptible to catastrophic repetition (Xu et al., 2026; Yao et al., 2025) (representative failures are shown in Appendix E.1), which is particularly challenging given the increasingly long-context tasks that agentic applications face. Once a model falls into this state, it continuously outputs meaningless repetitive content, wasting computational resources; in an agent pipeline, a single collapsed turn can also stall downstream tool calls, pollute intermediate state, and force costly retries. Prior work links these repetitions to self-reinforcing repetition dynamics that trap generation in low-diversity states (Xu et al., 2022), while feature-level analyses identify internal activations associated with such repetitions (Yao et al., 2025); KV cache reuse further amplifies this effect by inadvertently preserving high-scoring repetitive tokens at the expense of diverse context (Xu et al., 2026). To mitigate catastrophic repetition, prior work has proposed interventions across multiple levels (Xu et al., 2022; Dong et al., 2025; Yao et al., 2025; Xu et al., 2026). However, none of these works examines whether such interventions affect general model capabilities, leaving the risk of catastrophic forgetting unaddressed. Moreover, they rely on aggregate metrics such as n-gram repeat (Li et al., 2023), entropy (Tsai et al., 2008), and BLEU (Papineni et al., 2002), which characterize repetition at the sequence level and are therefore unable to localize the onset of a loop within the generated text.

In this paper, we first construct DE-RP, a repetition benchmark for long-context retrieval-augmented generation (RAG) (Lewis et al., 2020), and propose a character-level repetition detector, which can precisely identify the position where catastrophic repetition occurs. Analyzing models from the Qwen3 (Yang et al., 2025), Llama-3.1 (Grattafiori et al., 2024), and Hunyuan (Tencent Hunyuan Team, 2025) series across scales, we find that repetition can be induced across the evaluated models

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

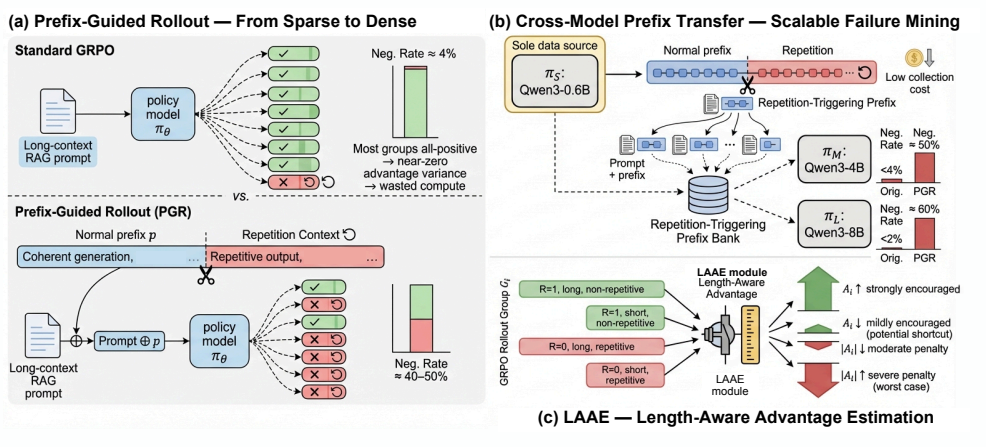


Figure 1: Overview of PREFIXRLVR: Prefix-Guided Rollout trains from high-risk trigger-prefix states, cross-model prefix transfer enables low-cost training, and length-aware advantage estimation mitigates the shortcut of favoring shorter generations.

by shared prefix patterns, which we define as *trigger prefixes* (Definition 1), generated by these models during the accomplishment of RAG tasks, suggesting shared vulnerabilities of the evaluated LLMs to such patterns. So, we frame the resolution of catastrophic repetition as a state-conditioned robustness problem against the trigger prefix, which is promisingly addressed by Reinforcement Learning with Verifiable Rewards (RLVR) (Mroueh, 2025).

GRPO and its variants optimize automatically verifiable outcomes (Guo et al., 2025; Yu et al., 2025; Shao et al., 2024), making repetition mitigation a natural application. However, directly applying these methods faces two challenges. Prior works have shown that group-based policy optimization can suffer from ineffective advantages when rollout groups contain highly imbalanced positive and negative samples (Yu et al., 2025). Moreover, reward-driven RL fine-tuning may encourage shortcut behaviors or reward hacking in certain settings (Ichihara et al., 2026); in our case, a model could reduce repetition risk by simply producing shorter continuations, thereby avoiding long-range failure states without genuinely improving robustness. To address these challenges, we introduce PREFIXRLVR, which presents two key innovations to the standard RLVR framework. First, we introduce Prefix-Guided Rollout (PGR) to improve the efficiency of negative sampling during rollouts while addressing the imbalance between positive and negative samples in the rollout process. Instead of optimizing the entire generation of the LLM, PREFIXRLVR only optimizes the portions of generation that are at high risk of falling into catastrophic repetition, which reduces the impact on the model's general capabilities during the training process. Second, we propose Length-Aware Advantage Estimation (LAAE), which prevents the model from exploiting the length shortcut of generating short outputs to lower the risk of falling into catastrophic repetition, ensuring the stability of the model's generation length before and after optimization while preserving its general capabilities . Main contributions are summarized as follows:

(1) We introduce DE-RP for quantifying repetition behavior of LLMs: a benchmark for repetition evaluation, together with a character-level detector, filling the gap where traditional metrics such as n-gram repeat are unable to quantify complex repetition behaviors.

(2) We are the first to reveal the cross-model transferability of trigger prefixes: we find that repetition failures can be reproducibly induced by trigger prefixes, which are transferable across the evaluated model series and scales. The probability of LLMs falling into catastrophic repetition is positively correlated with the length of trigger prefixes—taking the Qwen3 series as an example, this probability rises from 0.77%–10.11% to over 50% as prefix length increases.

(3) We introduce PREFIXRLVR: by incorporating PGR and LAAE into GRPO, PREFIXRLVR substantially reduces catastrophic repetition while preserving the model's general capabilities. After training, the rate of catastrophic repetition decreases by approximately 49–$330\times$, while performance in reasoning, coding, instruction-following, and knowledge remains nearly unchanged. Meanwhile,

it preserves or improves long-context performance, showing that PREFIXRLVR can substantially mitigate catastrophic repetition while further improving model capabilities across a broad range of long-context tasks.

2 EMPIRICAL MOTIVATION

Building on the self-reinforcement effect of repetition (Xu et al., 2022), we ask whether specific prefixes can trigger catastrophic repetition, and whether the high-risk signals associated with these prefixes transfer across models. We investigate these questions on DE-RP, a hard-prompt test set (Appendix B.3), using a detector that localizes repetition onset. The detector identifies periodic repeated spans of varying lengths at the character level. Unlike n-gram and RAP metrics (Li et al., 2023; Huang et al., 2025), which primarily characterize repetition at the whole-sequence level, it localizes the point at which a fluent generation collapses into a repetitive loop. Detector and evaluation details appear in Appendices B.1 and B.2.

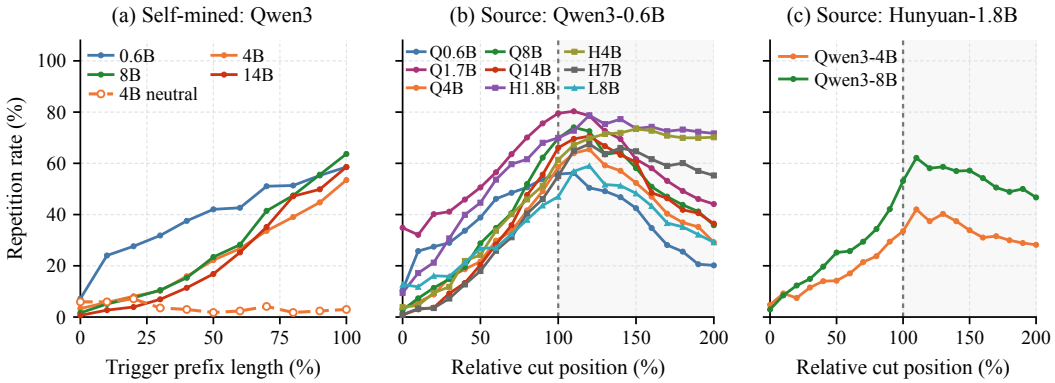


Figure 2: **Prefix-induced repetition and cross-model transfer.** (a) Within-model induction with an equal-length neutral control for Qwen3-4B (dashed orange). (b,c) Transfer from Qwen3-0.6B and Hunyuan-1.8B, respectively. Q/H/L denote Qwen3/Hunyuan/Llama-3.1. The full trigger prefix corresponds to 100%; shaded regions ($> 100\%$) include post-onset source text.

(i) Repetition weakens with scale but remains model-dependent. We repeatedly sample DE-RP 10 times as prompts to test the inherent catastrophic repetition rate of model-generated content across different model series and scales. Across both model series, the catastrophic repetition rate decreases consistently with scale: Qwen3 drops from 10.11% at 0.6B to 0.77% at 14B, and Hunyuan drops from 11.68% at 1.8B to1.03% at 7B, indicating that larger models are more capable of avoiding repetition failures in long-text generation tasks. In addition, we find that Llama-3.1 8B exhibits the highest repetition rate among all evaluated models, even higher than the smallest model of 0.6B parameters in the Qwen3 series, which demonstrates the variability in repetition failure behavior across different model series.

(ii) Catastrophic repetition is positively correlated with the length of trigger prefixes. To verify the existence of special prefix patterns that induce repetition failures, we collected samples in which catastrophic repetition were detected in Paragraph(i), and extracted the generation preceding the first detected repetition onset to obtain trigger prefix. The first $n\%$ of each trigger prefix together with the original prompt were then used as the prompt to induce the corresponding models to generate continuations. Repetition rates generally increase with n across model scales (Figure 2a). When the complete trigger prefix (n=100%) is used to induce the model's generation, all Qwen3 models regardless of scale demonstrate a catastrophic repetition rate exceeding 50%, which is substantially higher than the baseline results without trigger prefix induction (ranging from 0.77% to 10.11%). These trigger prefixes thus provide reproducible high-risk states for studying repetition.

As a length control, non-repetitive neutral prefixes yield no comparable increasing trend. For Qwen3-4B, the repetition rate at the complete prefix is 53.45% with trigger prefixes versus 2.96% with equal-length neutral prefixes (Figure 2a). These results suggest that prefix content, as well as length, affects repetition.

(iii) The cross-model transferability of Trigger prefixes. We take the trigger prefixes collected in Paragraph(i) that are generated by Qwen3-0.6B and Hunyuan-1.8B, and apply them respectively to larger Qwen3 models, Hunyuan models, and Llama-3.1 8B to evaluate whether the trigger prefixes can influence their repetition failures across different models. We describe the detailed experimental setup in Appendix B.5. As shown in Figure 2(b,c), as more pre-onset text is supplied (0%–100%), repetition rates generally increase across target models, suggesting that the evaluated models share vulnerabilities to these trigger prefixes, leading them to fall into catastrophic repetition.

(iv) Behavioral and hidden-state signals of trigger prefixes. We identify a reproducible repetition-associated signal in hidden states near the trigger-prefix boundary, including when Qwen-generated prefixes are supplied to Llama. This provides representation-level support for the increased repetition risk and cross-model transferability observed in the preceding continuation experiments.

We first examine generation behavior around the trigger-prefix boundary in repetitive base-model responses, using local copy rate and distinct-2 to measure repetition and lexical diversity, respectively. Local copy rate is the fraction of words that have appeared in the preceding local context within the window; distinct-2 is the fraction of unique word bigrams. Comparing equal-length windows immediately before and after the boundary, we observe a 14.0–18.0 percentage-point increase in local copy rate and a 21.2–27.0 percentage-point decrease in distinct-2 across the evaluated Qwen3 models and Llama-3.1-8B (Appendix D.1). These changes associate the prefix boundary with increased local repetition and reduced lexical diversity.

To examine the corresponding internal changes, we construct query-matched pairs of repetitive and non-repetitive responses and align their hidden states at identical generated-token positions. Let $\Delta h_{i,\ell}(t)$ denote the repetitive-minus-normal difference in layer-ℓ hidden states for pair i. Here, $t = 0$ denotes the state after processing the complete trigger prefix, immediately before predicting the first continuation token. For each calibration query, we average the hidden-state differences within a fixed token window near the trigger-prefix boundary, then across queries, and normalize the resulting vector to obtain a unit direction v_ℓ. On held-out queries, we define the mean normalized signal as

$$C_\ell(t) = \frac{1}{N} \sum_{i=1}^{N} \frac{\langle \Delta h_{i,\ell}(t), v_\ell \rangle}{a_\ell}. \tag{1}$$

Here, N is the number of test queries, and a_ℓ is the mean projection of the calibration hidden-state differences onto v_ℓ, used as the normalization scale. Positive values indicate a repetition-associated signal along the fitted direction, with one unit corresponding to the mean signal strength in the calibration window. For each test fold, both the direction and scale are estimated from the remaining query folds (Appendix D.2). In the cross-model analysis, we fit both quantities exclusively on Llama-generated responses and hold them fixed when analyzing Llama's hidden states under Qwen-generated prefixes.

As shown in Figure 3, the mean signals in all four displayed intermediate-to-late layers peak at $t = 0$ under both Llama-native and Qwen-generated prefixes. This indicates that Qwen-generated prefixes also engage Llama's own repetition-associated features. Directions fitted within the evaluated Qwen3 models likewise reveal stronger signals near the boundary (Appendix D.2).

We further repeat the same fitting and evaluation procedure in Llama at boundaries shifted before and after the detected boundary. Both shifted controls remain relatively flat around their respective boundaries, whereas the original boundary exhibits a clear local peak. This suggests that the peak is associated with the detected trigger-prefix boundary rather than simply induced by the fitting procedure (Appendix D.3). We also conduct a sensitivity analysis of the detector parameters. In a paired comparison with the three-occurrence rule, the default configuration shows a more prominent local signal on average (Appendix D.4).

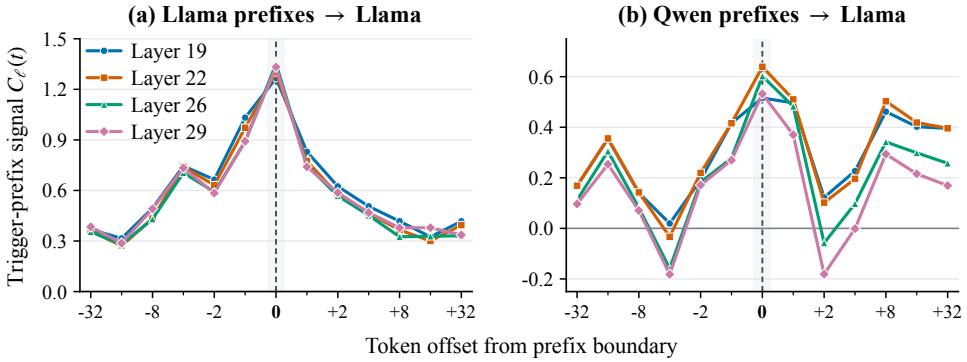


Figure 3: **Trigger-prefix signals in Llama under Llama- and Qwen-generated prefixes.** (a) Llama-3.1-8B conditioned on its own prefixes. (b) Llama-3.1-8B conditioned on Qwen3-0.6B-generated prefixes, using directions fitted only on Llama responses. Curves show held-out mean signals for four layers; the dashed line marks the complete-prefix boundary ($t = 0$). Colors denote the same layers in both panels. Panels use separate vertical scales and normalization factors.

Together with the continuation results, these findings support training from trigger prefixes. At the prefix boundary, the model already exhibits repetition-associated signals, while catastrophic repetition has not yet begun. Prefix-conditioned training therefore directly optimizes the model's continuation behavior from these high-risk states.

3 Learning from Trigger Prefixes

Building on Section 2, we formulate repetition mitigation as learning to generate non-repetitive continuations from high-risk trigger prefixes. PrefixRLVR combines Prefix-Guided Rollout (PGR) to alleviate reward sparsity with Length-Aware Advantage Estimation (LAAE), which dynamically adjusts advantages based on within-group length differences to discourage the shortcut of avoiding repetition through shorter outputs.

3.1 Problem Setup and Class-Imbalance Bottleneck

The output distribution of the LLM can be denoted by $\pi_\theta(\cdot \mid x)$, where x denotes a long-context RAG prompt, and $y \sim \pi_\theta(\cdot \mid x)$ is a sampled response. The policy π_θ is optimized with GRPO (Shao et al., 2024) under the binary verifiable reward:

$$r(y) = 1 - \mathrm{rep}(y), \qquad \mathrm{rep}(y) \in \{0, 1\}, \tag{2}$$

where $\mathrm{rep}(\cdot)$ denotes the indicator returned by the character-level repetition detector introduced in Section 2. For each prompt, GRPO draws a rollout group $g = \{y^{(i)}\}_{i=1}^{K}$ of K i.i.d. responses and standardizes their rewards into within-group advantages:

$$\hat{A}_i = \frac{r_i - \mu_g}{\sigma_g + \epsilon}, \qquad r_i = r(y^{(i)}), \tag{3}$$

with μ_g, σ_g the group reward mean and standard deviation. Let $\rho_{i,t} = \pi_\theta(y_t^{(i)} \mid x, y_{<t}^{(i)}) / \pi_{\theta_{\mathrm{old}}}(y_t^{(i)} \mid x, y_{<t}^{(i)})$ and $\bar{\rho}_{i,t} = \mathrm{clip}(\rho_{i,t}, 1-\epsilon_c, 1+\epsilon_c)$. The GRPO objective averages the clipped surrogate over the group and sequence positions, with a KL penalty to a frozen reference policy π_{ref}:

$$\mathcal{J}_{\mathrm{GRPO}}(\theta) = \mathbb{E}_{g \sim \pi_{\theta_{\mathrm{old}}}} \left[\frac{1}{K} \sum_{i=1}^{K} \frac{1}{|y^{(i)}|} \sum_{t} \min\left(\rho_{i,t}\hat{A}_i, \bar{\rho}_{i,t}\hat{A}_i\right) \right] - \beta D_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{ref}}). \tag{4}$$

We define rollouts with r_i=1 as *positive* and those with r_i=0 as *negative*.

The class-imbalance bottleneck. From Equations 2 and 3, it is straightforward to deduce that if a group g contains only positive samples or only negative samples—we define the set of such groups as g_{pure}—then $\hat{A}_i = 0$, resulting in the reward sparsity problem. For a fixed conditioning input, let q denote the probability that the model generates a negative sample. The probability that a rollout group does not belong to g_{pure} is given by:

$$P(g \notin g_{\text{pure}}) = 1 - (1-q)^K - q^K. \tag{5}$$

For instance, suppose $K = 8$ and the model has been optimized to a certain state where $q = 0.02$. Then $P(g \notin g_{\text{pure}}) \approx 14.9\%$, meaning that only approximately 14.9% of rollout groups provide non-zero within-group contrast, which degrades optimization efficiency.

3.2 Low-Cost Prefix Bank Construction

Drawing on the findings in Section 2, trigger prefixes substantially elevate the probability of generating negative samples. As the input-specific failure probability q moves toward 0.5, Eq. 5 gives a lower probability of pure-reward groups, thereby alleviating reward sparsity. We term this methodology PGR: during RLVR, rollouts are sampled as $o_i \sim \pi_\theta(\cdot \mid x_i \oplus p_i)$, where $x_i \oplus p_i$ denotes the input augmented with a trigger prefix, and p_i is mined from a generation sampled from an arbitrary LLM conditioned on x_i using the detector introduced in Appendix B.1. The prefix is fixed conditioning text: the policy loss, verifier, and continuation lengths are computed only on newly sampled o_i. Based on the finding in Section 2 that trigger prefixes are shared across model families and scales, we construct the prefix bank $\mathcal{P}$ using Qwen3-0.6B. Compared with utilizing Qwen3-4B, Qwen3-0.6B reduces the reported mining time by a factor of 3.3 and yields $2.3\times$ as many training examples under the same hardware setup (Appendix Table 8), making it a more cost-effective and sample-efficient choice for prefix mining.

3.3 Length-Aware Advantage Estimation (LAAE)

PGR addresses the bottleneck of reward sparsity; however, the design of its binary verifier is not structurally decoupled from length preferences, leading to over-optimization through generation length shifting (Shen et al., 2023; Chen et al., 2024; Li et al., 2025b). In our task, shorter continuations can avoid repetition without improving continuation quality. Length bias in standard GRPO can also arise from mathematical asymmetries in token-level aggregation, where response length disproportionately dilutes penalties for incorrect long responses while magnifying rewards for correct short ones (Liu et al., 2025; Ding et al., 2026). Directly introducing length-aware objectives is an effective approach to mitigating length bias (Li et al., 2025b). However, aggregating heterogeneous reward components before normalization may cause distinct reward compositions to collapse into identical advantage values, thereby reducing the resolution of the training signal (Liu et al., 2026b). We also evaluated reward-side length correction under the same prefix-conditioned setup, but it overcorrected toward longer continuations (Appendix Figure 4). Meanwhile, in our training tasks, response length is difficult to formulate as an explicit reward signal. We therefore introduce LAAE, which injects length awareness into advantage estimation without modifying the original verifier reward.

Length-Aware Scaling. We introduce a scaling factor at the advantage level to mitigate length bias. Specifically, for each sample i in group g, the scaling factor is defined as:

$$s_i = \min\Big\{\exp\Big[\tilde{m}_g \cdot \operatorname{sgn}(\hat{A}_i) \cdot \tanh(z_i)\Big], c\Big\}, \tag{6}$$

where c is a clipping threshold, and the two components $\tilde{m}_g$ and z_i are described as follows.

The macro-level scaling factor $\tilde{m}_g$ captures the relative length disparity between positive and negative continuations at the group level. It is defined as $\tilde{m}_g = \min\{\max(m_g - 1, 0), M\}$, where $m_g = \bar{L}_g^- / \bar{L}_g^+$; $\bar{L}_g^+$ and $\bar{L}_g^-$ denote the mean continuation token lengths of positive and negative samples in group g, respectively, and M is an upper-bound hyperparameter.

Micro-level length deviation z_i measures the relative length deviation of sample i with respect to the mean length of samples sharing the same polarity within the group: $z_i = \log((l_i + \delta)/(\bar{f}(i) + \delta))$, where l_i is the token count of sample i, $\delta > 0$ is a smoothing constant to avoid numerical instability, and the reference length $\bar{f}(i)$ is defined as $\bar{f}(i) = \bar{L}_g^+$ if $r_i = 1$ and $\bar{f}(i) = \bar{L}_g^-$ otherwise.

Re-centering yields the LAAE advantage

$$A_i^{\text{LAAE}} = \hat{A}_i s_i - \frac{1}{|g|} \sum_{j \in g} \hat{A}_j s_j, \tag{7}$$

which preserves GRPO's group-centered structure while leaving the binary verifier reward unchanged. A worked example and implementation details are given in Appendix A.2.

Training objective. PREFIXRLVR uses the standard GRPO paradigm (Shao et al., 2024), but computes rollout groups from the prefix-conditioned state $x \oplus p$ and replaces the standardized advantage $\hat{A}_i$ with the length-aware advantage A_i^{LAAE} in Eq. 7. The verifier reward remains unchanged, and the KL penalty is kept the same as in GRPO. Traditional GRPO and prefix-conditioned GRPO without LAAE can be viewed as two special cases of PREFIXRLVR. When $p = \emptyset$ and $s_i \equiv 1$, it degenerates to the traditional prompt-only GRPO; when $s_i \equiv 1$, it degenerates to prefix-conditioned GRPO without LAAE.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

We utilize Qwen3-0.6B to repeatedly generate samples with 20K long-context RAG examples as input. We employ the repetition detector to filter out samples that fall into catastrophic repetition, constructing trigger prefixes for PREFIXRLVR training and P+GRPO training (GRPO with PGR). For SFT (Ouyang et al., 2022), we fine-tune Qwen3-4B on non-repetitive responses generated by Qwen3-4B for the same training queries. The original prompts are used for GRPO training. All training approaches share the same set of 7,000 original prompts; the specific construction methodology is detailed in Appendices B.3, B.4, and A.3. For GRPO-based methods, the rollout group size is set to 8. In the main benchmark evaluations, models are evaluated on the original prompts rather than trigger-prefixed inputs. Implementation details, hyperparameters, and examples of training inputs are provided in Appendix A.3 and Appendix E.2. The 32K DE-RP test uses prompts disjoint by query from training (Appendix A.3). We additionally evaluate LZ Penalty (Ginart et al., 2026) as a decoding-time baseline using a fixed configuration of the authors' reference implementation (Appendix A.3).

4.2 RESULTS AND DISCUSSION

All variants are evaluated on DE-RP by duplicate n-gram statistics over $n \in \{3, 4, 5\}$ and repetition rate defined by our repetition detector to represent their collapse into repetitions. GPQA-Diamond (Rein et al., 2023), IFEval (Zhou et al., 2023), AIME 2024/2025 (AIME, 2025), MATH-500 (Lightman et al., 2023), and LiveCodeBench v5 (Jain et al., 2024) are also used to assess catastrophic forgetting in the base and post-training models. We summarize the results in Tables 1 and 2.

The results show that all training methods can mitigate the model's tendency to fall into catastrophic repetition. Among these methodologies, PREFIXRLVR reduces catastrophic repetition rates by approximately 49–330$\times$ across all models, while simultaneously exhibiting performance close to that of the base model on General Benchmarks. The same trend is observed on Llama-3.1-8B, further suggesting that models from different series share a common set of repetition patterns, so that they can be optimized utilizing the same trigger prefix. By contrast, SFT only partially reduces catastrophic repetition in Qwen3-4B, while degrading performance on several general benchmarks. Compared with LZ Penalty on DE-RP (Table 2), PREFIXRLVR (Ours) yields lower catastrophic and n-gram repetition on all four backbones. Catastrophic repetition is 0.07% versus 0.77% on Qwen3-4B and 0.09% versus 4.07% on Llama-3.1-8B. P+GRPO achieves similarly low repetition rates but substantially shortens non-repetitive outputs. Adding LAAE keeps their mean length closer to the base model while maintaining low repetition rates (Table 1). This length-preservation effect comes from the full LAAE design in Eq. 6: component ablations show that weakening its length-aware scaling leads to clear length drift, either toward overly long or substantially shorter continuations, whereas the full LAAE design keeps catastrophic repetition near zero while preserving the original generation length distribution. Table 3 reports the component ablations on Qwen3-4B, all using

Table 1: Main results. PREFIXRLVR reduces prompt-only repetition while preserving standard reasoning and instruction-following benchmarks across backbones. **Bold** and underline mark the best and second-best results within each model group; length is not ranked.

Model	Variant	General Benchmarks						DE-RP		
		GPQA-Diamond	IFEval (strict)	AIME 2024	AIME 2025	MATH-500	LCB v5	Rep. Rate (%) ↓	Pos. Seq. Length (vs. Base)	n-gram Repeat (%) ↓
Qwen3-0.6B	Base	25.3	57.1	3.1	2.8	53.2	4.9	9.90	526.7	11.83
	+P+GRPO	**27.6**	56.4	3.0	3.0	51.8	**6.9**	**0.03**	337.2 (−36.0%)	1.82
	+PREFIXRLVR	26.4	**57.3**	**3.3**	**3.1**	**53.4**	**6.9**	**0.03**	521.8 (−0.9%)	**1.66**
Qwen3-4B	Base	**44.9**	81.3	24.1	20.7	82.8	23.4	3.44	672.0	12.79
	+SFT	40.4	76.5	19.6	18.8	81.7	21.9	1.74	768.4 (+14.3%)	6.35
	+GRPO	42.6	80.0	22.1	20.1	82.6	22.9	0.19	478.0 (−28.9%)	5.87
	+P+GRPO	43.0	80.6	22.9	20.1	83.4	23.7	**0.03**	401.0 (−40.3%)	3.70
	+PREFIXRLVR	44.4	**81.7**	**24.6**	**20.9**	**84.4**	**24.3**	0.07	677.0 (+0.7%)	**3.35**
Qwen3-8B	Base	48.1	83.3	29.1	22.4	84.3	26.3	4.10	778.0	8.37
	+P+GRPO	47.5	81.9	27.8	**23.0**	**84.7**	**27.2**	0.05	505.0 (−35.1%)	1.71
	+PREFIXRLVR	**48.3**	**83.9**	**29.4**	**23.0**	84.3	**27.2**	**0.03**	706.0 (−9.3%)	**1.66**
Llama-3.1-8B	Base	**33.1**	73.0	**7.9**	**1.2**	**53.6**	6.4	14.16	722.0	20.80
	+P+GRPO	**33.1**	**73.7**	7.7	1.1	51.4	6.1	**0.07**	432.0 (−40.2%)	9.90
	+PREFIXRLVR	32.7	72.5	7.8	**1.2**	52.2	**6.7**	0.09	592.0 (−18.0%)	**7.50**

LCB denotes LiveCodeBench v5; IFEval uses strict prompt accuracy. Pos. Seq. Length is the mean length of non-repetitive outputs, with changes relative to Base. GRPO = prompt-only; P+GRPO = prefix-conditioned; both without LAAE.

Table 2: LZ Penalty comparison on DE-RP.

	Rep. (%) ↓		Length (%)		n-gram (%) ↓	
Model	LZ	Ours	LZ	Ours	LZ	Ours
Qwen3-0.6B	0.22	**0.03**	−25.1	−0.9	2.03	**1.66**
Qwen3-4B	0.77	**0.07**	−10.1	+0.7	7.70	**3.35**
Qwen3-8B	0.15	**0.03**	−14.0	−9.3	3.52	**1.66**
Llama-3.1-8B	4.07	**0.09**	−15.2	−18.0	10.19	**7.50**

Table 3: LAAE components (Qwen3-4B).

Variant	Rep. (%) ↓	Length (%) → 0
Full LAAE	0.07	+0.7
Binary gate	0.09	−27.2
Ungated scaling	0.12	+19.7
Positive-only scaling	0.07	−24.3

trigger prefixes. In Table 3, length denotes the percentage change in mean non-repetitive response length relative to base model. Training curves and additional metrics are provided in Appendix C.5.

We further leverage RULER (Hsieh et al., 2024) and LaRA (Li et al., 2025a) to evaluate the models' long-context capability ranging from 32K to 128K. For the Qwen3 series models, the claimed supported context length is 40K; we therefore follow the official long-context setup with YaRN (Peng et al., 2024). We use the GPT-4o-based (OpenAI, 2024) evaluation protocol from Li et al. (2025a) to conduct LaRA testing. More details are listed in Appendix C.1. Table 4 compares the overall scores of models trained with +PREFIXRLVR against their corresponding base models across all tasks from LaRA and RULER. It can be seen that models of different families and scales consistently improve their long-context handling ability on each benchmark after +PREFIXRLVR post-training. Notably, the task objectives of all tasks in LaRA and RULER are distinct from the optimization

Table 4: Long-context generalization. RULER Avg. averages tasks from 4K to 128K. †, ‡: Base scores from the original papers.

Model	Variant	RULER				LaRA-LC		LongWeave		
		Avg.	32K	64K	128K	32K	128K	Overall ↑	Rep. ↓	n-gram ↓
Llama-3.1-8B	Base	86.9	85.0	83.5	72.8	66.58†	55.10†	20.25‡	18.73	29.35
	Ours	**88.4**	**87.6**	**85.0**	**75.8**	**67.94**	**55.57**	**25.69**	**2.32**	**16.15**
Qwen3-8B	Base	86.7	87.4	79.2	75.6	65.33	46.45	27.17‡	14.80	25.62
	Ours	**89.6**	**90.1**	**83.2**	**76.3**	**66.44**	**46.50**	**31.98**	**5.70**	**19.80**
Qwen3-4B	Base	83.9	86.0	76.0	65.8	63.97	39.58	19.18‡	14.60	29.84
	Ours	**84.5**	**87.0**	**76.4**	**66.6**	**64.10**	**40.17**	**27.83**	**5.80**	**23.98**
Qwen3-0.6B	Base	37.5	23.2	19.8	21.6	10.66	9.06	–	–	–
	Ours	**38.8**	**23.5**	**21.1**	**22.1**	**12.86**	**9.13**	–	–	–

objective of +PREFIXRLVR, which penalizes catastrophic repetition. These results support transfer to long-context tasks beyond the repetition-specific training objective. The per-task LaRA breakdown is provided in Appendix Table 7.

We also evaluate the repetition loop rate on DE-RP with testing sequence lengths from 64K to 128K. All base models exhibit a catastrophic repetition rate exceeding 10%, indicating significant uncertainty when dealing with tasks beyond their training length. This uncertainty can be substantially mitigated via PREFIXRLVR post-training within the officially claimed training length. For Qwen3-4B, repetition rates fall from 27.35% to 4.40% at 64K and from 42.50% to 17.75% at 128K (Appendix Table 9). Because RULER and LaRA-LC mostly require short answers, we additionally evaluate longer outputs.

Generalization to long-input/long-output generation. We evaluate long-input/long-output generation with LongWeave (Xiao et al., 2025), using its 64K-input/8K-output setting across all tasks. We follow the original decoding configuration and LLM-as-judge protocol. Our local base model evaluations yield broadly similar scores; we report the published scores as reference baselines. As shown in Table 4, +PREFIXRLVR consistently improves LongWeave Overall scores by 17.7%–45.1% relative to the corresponding base models. More importantly, the gains coincide with substantially fewer repetition failures: repetition rates on LongWeave outputs drop by 60.3%–87.6%, while duplicate n-gram rates decrease by 19.6%–45.0%. On Qwen3-4B, the full method outperforms P+GRPO on LongWeave Overall (27.83 vs. 18.92). These results indicate that PREFIXRLVR provides more substantial long-context capability improvements on tasks requiring longer generation, in contrast to tasks in RULER and LaRA-LC, whose expected generation length is generally shorter than 1k tokens.

5 RELATED WORK

Prior work addresses repetition focuses on decoding controls (Ginart et al., 2026), training (Xu et al., 2022) , mechanistic and data analyses (Li et al., 2023; Yao et al., 2025; Hiraoka & Inui, 2025; Mahaut & Franzon, 2025), but typically intervenes after repetition becomes observable; we instead mine pre-collapse trigger prefixes for targeted RLVR post-training. RL post-training is vulnerable to reward hacking and length drift (Chen et al., 2024; Cai et al., 2025), often addressed by reward shaping (Shen et al., 2023; Li et al., 2025b) or token-level aggregation corrections (Liu et al., 2025; 2026a). We study a shortening shortcut: strict repetition verification can favor overly short continuations, which our length-aware advantage scaling mitigates without changing the verifier reward.

6 CONCLUSION

For catastrophic repetition in long-context RAG, we find that repetition can be reproducibly induced by transferable trigger-prefix patterns. The state analysis identifies a signal near this repetition onset that generalizes across queries and models. We introduce PREFIXRLVR to train on these high-risk states, combining prefix-guided RLVR with length-aware advantage estimation to mitigate the shortening shortcut. Experiments show that PREFIXRLVR substantially reduces catastrophic repetition rates across the Qwen3 series and Llama-3.1-8B, reduces length drift and broadly maintains reasoning, instruction-following, and coding capabilities. Long-context evaluations indicate that PREFIXRLVR offers a promising pathway to substantially enhance the long-context capabilities of LLMs, thereby facilitating their deployment in long-context-based agentic applications.

7 LIMITATIONS

PREFIXRLVR is designed to mitigate catastrophic repetition, not the broader spectrum of long-form generation failures such as redundancy, hallucination, citation errors, or topic drift. Although RULER, LaRA and LongWeave indicate some robustness beyond the training detector, they are not a comprehensive stress test for all long-context failure modes. Our main corpus is a long-context RAG stress test over web pages, scientific papers, and patents, so repetition rates may vary across domains, prompting formats, retrieval distributions, and decoding regimes. Finally, we train models up to 8B parameters and evaluate 64K/128K contexts through YaRN extrapolation from 32K-trained policies; behavior at substantially larger scales and under native long-context training remains open.

Reproducibility Statement

Method, data, evaluation, and additional analysis details are provided in the appendices. Anonymous code and data are linked in the abstract.

AI Use Statement

AI tools were used to assist with analysis organization and English language polishing.

References

AIME. AIME problems and solutions, 2025. URL https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions.

Jianfeng Cai, Jinhua Zhu, Ruopei Sun, Yue Wang, Li Li, Wengang Zhou, and Houqiang Li. Disentangling length bias in preference learning via response-conditioned modeling, 2025. URL https://arxiv.org/abs/2502.00814.

Lichang Chen, Chen Zhu, Jiuhai Chen, Davit Soselia, Tianyi Zhou, Tom Goldstein, Heng Huang, Mohammad Shoeybi, and Bryan Catanzaro. ODIN: Disentangled reward mitigates hacking in RLHF. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Fei Ding, Yongkang Zhang, Runhao Liu, Yuhao Liao, Zijian Zeng, Huiming Yang, Sibo wang, and Linglin Liao. Rethinking the comparison unit in sequence-level reinforcement learning: An equal-length paired training framework from loss correction to sample construction, 2026. URL https://arxiv.org/abs/2604.17328.

Yihong Dong, Yuchen Liu, Xue Jiang, Bin Gu, Zhi Jin, and Ge Li. Rethinking repetition problems of LLMs in code generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 965–985, 2025.

Antonio A. Ginart, Naveen Kodali, Jason Lee, Caiming Xiong, Silvio Savarese, and John R. Emmons. LZ penalty: An information-theoretic repetition penalty for autoregressive language models. *Transactions on Machine Learning Research*, 2026.

Aaron Grattafiori et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

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

Tatsuya Hiraoka and Kentaro Inui. Repetition neurons: How do language models produce repetitions? In *Proceedings of the 2025 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2025.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models?, 2024. URL https://arxiv.org/abs/2404.06654.

Donghao Huang, Thanh-Son Nguyen, Fiona Liausvia, and Zhaoxia Wang. RAP: A metric for balancing repetition and performance in open-source large language models. In *Proceedings of the 2025 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2025.

Yuki Ichihara, Yuu Jinnai, Tetsuro Morimura, Mitsuki Sakamoto, Ryota Mitsuhashi, and Eiji Uchibe. Mo-grpo: Mitigating reward hacking of group relative policy optimization on multi-objective problems, 2026. URL https://arxiv.org/abs/2509.22047.

Naman Jain, King Han, Alex Gu, Wen Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. *CoRR*, abs/2403.07974, 2024.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, pp. 611–626, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400702297. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, 2020.

Huayang Li, Tian Lan, Zihao Fu, Deng Cai, Lemao Liu, Nigel Collier, Taro Watanabe, and Yixuan Su. Repetition in repetition out: Towards understanding neural text degeneration from the data perspective. *Advances in Neural Information Processing Systems*, 36:72888–72903, 2023.

Kuan Li, Liwen Zhang, Yong Jiang, Pengjun Xie, Fei Huang, Shuai Wang, and Minhao Cheng. Lara: Benchmarking retrieval-augmented generation and long-context llms – no silver bullet for lc or rag routing, 2025a. URL https://arxiv.org/abs/2502.09977.

Ruosen Li, Ziming Luo, Quan Zhang, Ruochen Li, Ben Zhou, Ali Payani, and Xinya Du. AALC: Large language model efficient reasoning via adaptive accuracy-length control. *arXiv preprint arXiv:2506.20160*, 2025b.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *CoRR*, abs/2305.20050, 2023.

Fanfan Liu, Youyang Yin, Peng Shi, Siqi Yang, Zhixiong Zeng, and Haibo Qiu. Length-unbiased sequence policy optimization: Revealing and controlling response length variation in rlvr, 2026a. URL https://arxiv.org/abs/2602.05261.

Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, Yejin Choi, Jan Kautz, and Pavlo Molchanov. GDPO: Group reward-decoupled normalization policy optimization for multi-reward RL optimization. *arXiv preprint arXiv:2601.05242*, 2026b.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective, 2025. URL https://arxiv.org/abs/2503.20783.

Matéo Mahaut and Francesca Franzon. Repetitions are not all alike: distinct mechanisms sustain repetition in language models. *arXiv preprint arXiv:2504.01100*, 2025.

Youssef Mroueh. Reinforcement learning with verifiable rewards: Grpo's effective loss, dynamics, and success amplification, 2025. URL https://arxiv.org/abs/2503.06639.

OpenAI. GPT-4o system card. *arXiv preprint arXiv:2410.21276*, 2024. URL https://arxiv.org/abs/2410.21276.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318, 2002.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. Yarn: Efficient context window extension of large language models. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 31932–31951, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/874a4d89f2d04b4bcf9a2c19545cf040-Paper-Conference.pdf.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level Google-proof Q&A benchmark. *CoRR*, abs/2311.12022, 2023.

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

Wei Shen, Rui Zheng, Wenyu Zhan, Jun Zhao, Shihan Dou, Tao Gui, Qi Zhang, and Xuanjing Huang. Loose lips sink ships: Mitigating length bias in reinforcement learning from human feedback. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 2859–2873, 2023.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. HybridFlow: A flexible and efficient RLHF framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, pp. 1279–1297. Association for Computing Machinery, 2025. doi: 10.1145/3689031.3696075. URL https://doi.org/10.1145/3689031.3696075.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *ArXiv*, abs/1909.08053, 2019. URL https://api.semanticscholar.org/CorpusID:202660670.

Tencent Hunyuan Team. Hunyuan dense model collection. https://huggingface.co/collections/tencent/hunyuan-dense-model, 2025.

Du-Yih Tsai, Yongbum Lee, and Eri Matsuyama. Information entropy measure for evaluation of image quality. *Journal of digital imaging*, 21:338–347, 2008.

Zikai Xiao, Fei Huang, Jianhong Tu, Jianhui Wei, Wen Ma, Yuxuan Zhou, Jian Wu, Bowen Yu, Zuozhu Liu, and Junyang Lin. LongWeave: A long-form generation benchmark bridging real-world relevance and verifiability. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 10386–10417, Suzhou, China, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-emnlp.549. URL https://aclanthology.org/2025.findings-emnlp.549/.

Dongjie Xu, Hao Wu, Weijie Shi, Yue Cui, Yuanjun Liu, Jiawei Li, Haolun Ma, An Liu, Jia Zhu, and Jiajie Xu. LoopGuard: Breaking self-reinforcing attention loops via dynamic KV cache intervention. *arXiv preprint arXiv:2604.10044*, 2026.

Jin Xu, Xiaojiang Liu, Jianhao Yan, Deng Cai, Huayang Li, and Jian Li. Learning to break the loop: Analyzing and mitigating repetitions for neural text generation. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 3082–3095. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/148c0aeea1c5da82f4fa86a09d4190da-Paper-Conference.pdf.

An Yang et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Junchi Yao, Shu Yang, Jianhua Xu, Lijie Hu, Mengdi Li, and Di Wang. Understanding the repeat curse in large language models from a feature perspective. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 7787–7815, 2025.

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

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *CoRR*, abs/2311.07911, 2023.

A METHOD AND TRAINING DETAILS

A.1 FORMAL DEFINITIONS

Definition 1 (Trigger prefix)**.** *Given a prompt x and a response y for which the detector identifies the repetition boundary $\tau(y)$, the* trigger prefix *is the substring of y before character offset $\tau(y)$.*

Definition 2 (Prefix-Guided Rollout)**.** *Given a prefix bank $\mathcal{P}$,* Prefix-Guided Rollout *(PGR) samples $(x, p) \sim \mathcal{P}$ and draws a group $\{o_i\}_{i=1}^{K}$ with $o_i \sim \pi_\theta(\cdot \mid x \oplus p)$, treating p as fixed conditioning context. The verifier reward $r_i = r(o_i)$ is computed on the continuation only.*

A.2 LENGTH-AWARE ADVANTAGE ESTIMATION

This appendix gives the implementation details behind LAAE, the length-aware advantage estimator used in PREFIXRLVR. Its purpose is narrow: the binary repetition verifier only checks whether a continuation falls into a catastrophic repetition, and therefore lacks a completeness signal. Without an additional correction, a model can reduce repetition risk by producing very short non-repetitive continuations. LAAE addresses this short-positive shortcut at the advantage level while leaving the verifier reward unchanged.

Intuitive example. Consider a prefix-conditioned rollout group of size $K{=}8$ on a single prompt, producing 5 positive (non-repetitive) and 3 negative (repetitive) continuations. Suppose the positive continuations have lengths $\{60, 75, 90, 110, 400\}$ tokens and the negatives have lengths $\{1800, 2200, 2500\}$ tokens. The mean positive length is $\bar{L}^{+}{=}147$ and the mean negative length is $\bar{L}^{-}{=}2167$, giving a group-level gate $m_g{=}2167/147{\approx}14.7$ and, with $M{=}5$, thus $\tilde{m}_g = \min\{\max(m_g - 1, 0), M\} = 5$. Because $\tilde{m}_g > 0$, LAAE activates and adjusts pre-centering scales as follows:

- The 60-token positive ($z_i{<}0$, shorter than class mean) receives $s_i{<}1$, *downweighting* its advantage—discouraging short positives.
- The 400-token positive ($z_i{>}0$, longer than class mean) receives $s_i{>}1$, *upweighting* its advantage—restoring update strength for longer non-repetitive continuations.
- The 1800-token negative ($z_i{<}0$, shorter than class mean) receives $s_i{>}1$, *amplifying* its penalty—short repetitive continuations are penalized more.
- The 2500-token negative ($z_i{>}0$, longer than class mean) receives $s_i{<}1$, slightly *softening* its penalty.

LAAE breaks this degeneracy by coupling length information into the advantage without modifying the verifier-defined reward. Thus, LAAE is not a general preference for longer outputs: it activates only when negatives are longer than positives at the group level, and then uses within-class length deviations to modulate update magnitudes while preserving the verifier-defined reward labels.

Formal details. For each prefix-conditioned rollout group g, let o_i denote the continuation of sample i, l_i its token length, and $\hat{A}_i$ the standard GRPO group-normalized advantage computed from the binary verifier reward. Positive samples are non-repetitive continuations ($r_i = 1$), and negative samples are repetitive continuations ($r_i = 0$). LAAE rescales $\hat{A}_i$ using group-level evidence of a short-positive shortcut and then recenters the result within the group.

Reference lengths. The implementation uses group-wise mean continuation lengths $\bar{L}_g^{+}$ and $\bar{L}_g^{-}$ for positive and negative samples, respectively. Their ratio $m_g = \bar{L}_g^{-}/\bar{L}_g^{+}$ measures whether the group exhibits the short-positive pattern. The clipped gate $\tilde{m}_g = \min\{\max(m_g - 1, 0), M\}$ is therefore zero unless negatives are longer than positives on average. For the class-matched reference length, we use the same notation as the main text: $\bar{f}(i) = \bar{L}_g^{+}$ for $r_i = 1$ and $\bar{f}(i) = \bar{L}_g^{-}$ for $r_i = 0$.

Clipped implementation form. For each sample i, we first form the class-matched log-length ratio

$$z_i = \log \frac{l_i + \delta}{\bar{f}(i) + \delta}, \tag{8}$$

Table 5: Hyperparameters for SFT and GRPO-based training.

Category	Parameter	Value
SFT (Qwen3-4B)	Learning rate	$7 \times 10^{-6} \rightarrow 7 \times 10^{-7}$ (cosine decay)
	Batch size	64
GRPO / P+GRPO / PREFIXRLVR	Learning rate	1×10^{-6}
	LR warmup steps	10
	Weight decay	0.1
	Train batch size	64
	Rollout group size K	8
vLLM (Kwon et al., 2023) (rollout & eval)	`top_k`	20
	`top_p`	0.95
	`temperature`	0.6
	`max_new_tokens`	8192
	Input / output budget	32K / 8K tokens
	Total context budget	40K tokens
LAAE constants	δ	1
	c (scaling cap)	2
	M (gate ceiling)	5

then apply the clipped gate and capped multiplicative scale

$$\tilde{m}_g = \min\{\max(m_g - 1,\, 0)\,,\, M\}\,, \tag{9}$$

$$s_i = \min\Big\{\exp\Big(\tilde{m}_g \operatorname{sgn}(\hat{A}_i)\, \tanh(z_i)\Big),\, c\Big\}\,. \tag{10}$$

If either polarity is absent in a group, or the binary rewards are homogeneous so that GRPO provides no stable within-group contrast, we fall back to $s_i = 1$. This reduces the update to the standard GRPO advantage for that group and avoids introducing a length signal when the required comparison is unavailable.

Recentering after scaling. The scaled quantity actually inserted into the GRPO loss is obtained by re-centering $\hat{A}_i s_i$ within each group:

$$A_i^{\text{LAAE}} = \hat{A}_i\, s_i - \frac{1}{|g|} \sum_{j \in g} \hat{A}_j\, s_j. \tag{11}$$

Re-centering restores zero-mean advantages within each group.

Constants. The implementation uses δ=1, c=2, and M=5 in all experiments. These constants are manually chosen stability controls: δ smooths the length ratio near zero, c caps extreme multiplicative scaling, and M limits the group-level gate magnitude. They are not intended as theoretically optimal hyperparameters.

A.3 HYPERPARAMETERS

The 32K DE-RP stress test contains approximately 730 hard RAG prompts, disjoint by query from the 7K training prompts. We sample eight continuations per test prompt with an 8K-token output budget. Base and post-training variants share stochastic decoding settings (temperature 0.6, top-p 0.95, top-k 20), and repetition rates aggregate detector outcomes across continuations. These rates characterize the selected hard prompts. We implement all RLVR training runs (GRPO, P+GRPO, and PREFIXRLVR) with the Verl framework (Sheng et al., 2025). All training is conducted on NVIDIA A800 GPUs using Megatron-LM (Shoeybi et al., 2019) equipped with flash attention modules, with a batch size of 64. The number of rollouts for GRPO and PREFIXRLVR is set to 8. We adopt a cosine learning rate schedule, decaying the final learning rate to 10% of the peak learning rate. In the main benchmark evaluations, the input of all models is the original prompt rather than the trigger prefix. Table 5 lists the main training and generation hyperparameters.

Training and evaluation use 32K input tokens with an 8K-token output budget (`max_new_tokens`=8192), for a 40K total context budget unless stated otherwise.

LZ Penalty baseline. We port the authors' SGLang implementation of LZ Penalty (Ginart et al., 2026) (commit `4e9c0157`) to vLLM 0.18.0. Across all four backbones, the penalty uses only generated-token history, with strength 0.15, window size 512, and matching buffer size 32. LZ Penalty uses the same DE-RP test prompts, decoding settings, sampling budget, output budget, and evaluation metrics described above.

B DATA AND EVALUATION DETAILS

B.1 CHARACTER-LEVEL REPETITION DETECTOR

Our verifier is intentionally narrow: it targets the catastrophic repetition that make long-context RAG outputs unusable, not general semantic redundancy. This is sufficient because the RL objective only requires a stable label for the target failure mode; the learned behavior is robustness to the repetition patterns that precede it.

Algorithm. The implementation lowercases the response and splits it on literal spaces. A completion y is labeled *repetitive* ($\mathrm{rep}(y)=1$) when at least four distinct words each recur four or more times at a constant word-level period $d \in \{1\} \cup \{6, 7, \dots\}$. We exclude periods $d \in \{2, \dots, 5\}$ to reduce false positives observed during manual inspection. The detector locates the boundary retrospectively by tracing the selected qualifying periodic-word run back to its first occurrence. The corresponding character cut defines $\tau(y)$, and the trigger prefix used by PGR is the substring of y strictly before $\tau(y)$. Because $\tau(y)$ is a character offset on the raw response, the trigger prefix is a plain substring that any target tokenizer can re-encode without re-mining—enabling the cross-model transfer of Section 3.2.

The full implementation is released with the code in the anonymous repository.

B.2 HUMAN EVALUATION OF THE REPETITION DETECTOR

To evaluate whether the character-level repetition detector is reliable enough to serve as the RLVR verifier (Section 2), we annotate 200 responses balanced by the final human label (100 repetitive and 100 non-repetitive). Two annotators independently judged whether each response contains a catastrophic repetition loop, and disagreements were resolved by discussion.

Table 6 summarizes the results. The detector achieves 98.0% precision, 99.0% recall, and 98.5% accuracy. This supports its use as a high-precision training-time verifier for the specific collapse mode studied in this paper.

Table 6: Human evaluation of the character-level repetition detector on 200 balanced samples.

Human label	**Detector prediction**	
	Rep.	**Non-rep.**
Rep. (100)	99 (TP)	1 (FN)
Non-rep. (100)	2 (FP)	98 (TN)

B.3 DE-RP CONSTRUCTION

DE-RP is a long-context RAG repetition stress benchmark consisting of 731 hard prompts. It is curated from a 20K-query long-context RAG corpus, where each example contains roughly 32K input tokens. The corpus aggregates heterogeneous sources, including web pages, scientific papers, and patents; each example consists of a user query paired with retrieved reference literature and is formatted with the prompt template in Appendix B.4.

Benchmark construction and usage. We construct DE-RP by applying stochastic generation to candidate prompts and filtering the outputs with the character-level repetition detector. A prompt is retained when at least one sampled response exhibits catastrophic repetition. The resulting 731 hard

Table 7: Per-task LaRA scores at 32K and 128K. †Llama-3.1-8B Base from Li et al. (2025a).

		LaRA 32K					LaRA 128K				
Model	**Variant**	**Location**	**Reasoning**	**Comp.**	**Hallu.**	**Overall**	**Location**	**Reasoning**	**Comp.**	**Hallu.**	**Overall**
Llama-3.1-8B	Base†	**77.60**	51.65	58.29	**78.80**	66.58	**72.64**	48.48	**40.87**	58.40	55.10
	+PREFIXRLVR	77.03	**58.34**	**62.65**	73.74	**67.94**	68.90	**52.63**	40.78	**59.96**	**55.57**
Qwen3-8B	Base	**75.05**	52.12	**67.25**	66.90	65.33	62.22	**46.07**	40.28	**37.24**	46.45
	+PREFIXRLVR	74.99	**57.62**	64.13	**69.01**	**66.44**	**62.41**	43.77	**42.63**	37.20	**46.50**
Qwen3-4B	Base	76.27	**50.80**	58.95	**69.85**	63.97	54.17	36.97	**29.34**	**37.86**	39.58
	+PREFIXRLVR	**79.09**	49.17	**61.99**	66.14	**64.10**	**58.33**	**37.04**	28.88	36.43	**40.17**
Qwen3-0.6B	Base	33.18	3.71	1.15	4.58	10.66	20.23	5.57	**1.83**	**8.60**	9.06
	+PREFIXRLVR	**34.38**	**8.39**	**2.66**	**6.00**	**12.86**	**20.36**	**6.64**	1.47	8.05	**9.13**

prompts are used both for the motivating analyses in Section 2 and for the final model validation in Sections 4.1. For final evaluation, we score each prompt with 8 stochastic samples.

Training split. Independently of DE-RP, we collect 7K training samples by sampling Qwen3-0.6B on the same 20K-query corpus. This training split is disjoint from the DE-RP benchmark, ensuring that the final repetition results are evaluated on prompts unseen during training-data construction. All training variants use the same 7K original queries: traditional GRPO uses the original prompts directly, SFT fine-tunes Qwen3-4B on its own non-repetitive responses to these queries, and PREFIXRLVR / PGR use trigger-prefix-augmented inputs mined from the same training queries.

Long-context extensions. To evaluate repetition robustness under substantially longer retrieval contexts, we additionally construct 64K and 128K variants of DE-RP, each containing approximately 250 examples. These variants are derived from the same DE-RP prompts and data sources, but extend the retrieved reference literature to the target context lengths. Thus, the task distribution and user queries remain aligned with DE-RP, while the evidence context becomes substantially longer. The 64K/128K variants are used only for evaluation; all training data and trigger-prefix mining for the reported models remain based on 32K inputs unless otherwise specified.

B.4 PROMPT TEMPLATE

We use the same instruction template for hard-prompt sampling, RL training rollouts, and final evaluation. Thus, differences across stages come from the policy and rollout state distribution, not from prompt rephrasing. For each example, the placeholder `{query}` is replaced with the user question and `{ref}` is replaced with the retrieved reference literature.

```
Request:
Please answer the question: {query} by referencing the provided
literature: {ref}.
Requirements:
Your output must be detailed.
You must answer based only on the facts presented in the reference
literature; do not fabricate any information.
You must include citations to the corresponding reference numbers in your
answer, e.g., [1].
```

B.5 TRIGGER-PREFIX CONSTRUCTION

Trigger prefixes are constructed with the same detector-based procedure in two settings. For training, prefixes are mined only from the 7K training split described in Appendix B.3. For the prefix-induction analyses in Figure 2, prefixes are constructed on DE-RP to study whether repetition-inducing states are transferable across models.

The prefix-induction analyses in Figure 2 use 10 stochastic samples per constructed input. For a source model, we first sample responses on DE-RP and collect negative responses identified by the repetition detector. For each negative response, we locate the first detected repetition onset and define

the trigger prefix as the generated text before that onset. We then split each prefix by percentage into induction ratios $r \in \{0, 10, \ldots, 100\}$, where $r = 0$ corresponds to continuing from the original prompt without an added generated prefix and $r = 100$ uses the full trigger prefix. At each ratio, the target model continues from the original prompt concatenated with the corresponding prefix segment. We draw 10 stochastic continuations for each prefix-conditioned input and report the repetition rate over these continuations.

For Figure 2(a), the source and target are matched within the Qwen3 series: for each Qwen3 model scale, we collect that model's own negative responses on DE-RP, construct percentage prefixes from its own trigger prefixes, and ask the same model to continue with 10 stochastic samples per prefix ratio. This measures whether each Qwen3 model's self-mined prefixes become more repetition-inducing as they approach the detected repetition boundary.

For Figure 2(b), the source model is fixed to Qwen3-0.6B. We collect Qwen3-0.6B negative responses on DE-RP, construct a 0–200% grid normalized by pre-onset length, and use these prefixes to condition different target models, including larger Qwen3 models, Hunyuan models, and Llama-3.1, again with 10 stochastic continuations for each constructed input. This tests whether prefixes mined from a low-cost source model transfer across target scales and model series.

For Figure 2(c), the source model is Hunyuan-1.8B. We collect Hunyuan-1.8B negative responses on DE-RP, construct the same normalized 0–200% grid, and use it to condition Qwen3-4B and Qwen3-8B continuations, with 10 stochastic samples per prefix-conditioned input. This reverse-direction experiment checks whether the trigger-prefix structure is specific to Qwen3-mined prefixes or also transfers from Hunyuan-mined failures back to Qwen3 targets.

For the neutral control in Figure 2(a), we use non-repetitive model-generated prefixes matched to the trigger-prefix lengths at each retained fraction and score only the new continuations. Neutral and trigger prefixes come from separate diagnostic collections.

C ADDITIONAL RESULTS

C.1 LARA DETAILED RESULTS

Table 7 reports the per-task LaRA scores at both 32K and 128K context lengths. We adopt the original LaRA data, evaluation protocol, and GPT-4o-based LLM judge from Li et al. (2025a). Llama-3.1-8B Base scores are taken directly from the original paper; all other scores are from our own evaluation. Category-level changes are mixed, but the overall scores are maintained with small gains for every reported model and context length, consistent with the main-text claim that robustness to repetition patterns does not simply overfit the repetition verifier.

Table 8: Cross-model mining efficiency for Qwen3-4B (evaluated on DE-RP).

Prefix Source	**Reported Time (8×A800)**	**Training Examples**	**Rep. Rate (%) ↓**	**n-gram Repeat (%) ↓**
Qwen3-4B (self)	40 h	∼3K	0.07	4.07
Qwen3-0.6B	12 h	∼7K	**0.07**	**3.35**

C.2 CROSS-MODEL MINING EFFICIENCY

Table 8 compares two prefix-mining strategies for training Qwen3-4B: self-mining from Qwen3-4B versus cross-model mining from Qwen3-0.6B. Both pipelines use the same hardware setup (8×A800 GPUs) and the same RL recipe. Cross-model mining is substantially cheaper and produces more data, because the small model's higher base repetition rate (∼10% vs. ∼3%) exposes negatives far more frequently. The resulting policy is also slightly better on n-gram repeat, supporting the main-text claim that low-cost source models can provide dense and useful coverage of transferable repetition patterns.

Table 9: DE-RP 64K/128K results (YaRN extrapolation; all models trained on 32K data only).

Backbone	Variant	DE-RP 64K		DE-RP 128K	
		Rep. Rate (%) ↓	n-gram Repeat (%) ↓	Rep. Rate (%) ↓	n-gram Repeat (%) ↓
Qwen3-0.6B	Base	26.70	28.76	35.37	32.47
	+PREFIXRLVR	**13.50**	**15.91**	**31.45**	**23.13**
Qwen3-4B	Base	27.35	30.13	42.50	39.29
	+PREFIXRLVR	**4.40**	**7.17**	**17.75**	**19.17**
Qwen3-8B	Base	11.10	14.11	21.60	20.18
	+PREFIXRLVR	**2.00**	**5.16**	**2.50**	**6.06**

C.3 ADDITIONAL LONG-CONTEXT RESULTS

Table 9 collects the longer-context DE-RP stress tests omitted from the main text table. Although all policies are trained only on 32K prefix-conditioned data, the anti-repetition effect transfers after YaRN extrapolation. At the same time, the base-model results show that YaRN-enabled 64K/128K generation can still suffer from severe repetition, especially on smaller models: on Qwen3-0.6B, PREFIXRLVR reduces repetition from 26.7% to 13.5% at 64K and from 35.37% to 31.45% at 128K; on Qwen3-4B, from 27.35% to 4.40% at 64K and from 42.5% to 17.75% at 128K; and on Qwen3-8B, from 11.1% to 2.0% at 64K and 21.6% to 2.5% at 128K. The weaker transfer on the smallest model is reflected in the limitations discussion.

C.4 LLM-AS-JUDGE QUALITY EVALUATION ON DE-RP

To complement repetition-rate and length metrics, we check whether repetition suppression degrades the quality of answers that the base model already produces cleanly. We randomly sample 100 DE-RP queries on which *both* Qwen3-4B Base and Qwen3-4B+PREFIXRLVR generate non-repetitive answers, and ask GPT-4o (OpenAI, 2024)—the same judge used by LaRA (Li et al., 2025a)—to rate each answer independently along five dimensions (relevance, faithfulness, completeness, coherence, citation quality) on a 1–5 scale. For each pair we record a win for the variant with the higher average score, or a tie when the averages are equal.

Table 10: GPT-4o LLM-as-Judge on 100 DE-RP queries for which both Qwen3-4B Base and +PREFIXRLVR produce non-repetitive answers. Wins are decided by the higher average of five 1–5 ratings.

Outcome	Count
+PREFIXRLVR wins	18
Base wins	16
Tie	66

Table 10 reports the counts. The two variants are comparable (18 wins for PREFIXRLVR, 16 for Base, and 66 ties), providing a sanity check that the repetition gains do not obviously degrade answer quality. The judge prompt is reproduced below; `{query}`, `{ref_text}`, and `{answer}` are filled in per sample before the prompt is sent to GPT-4o.

```
You are an expert evaluator for long-context RAG systems. Given a query,
reference literature, and a model-generated answer, rate the answer on
the following dimensions using a 1-5 scale (1 = very poor, 5 =
excellent).
Dimensions:
1. Relevance: Does the answer address the query directly?
2. Faithfulness: Is the answer grounded in the provided references
without fabrication?
3. Completeness: Does the answer cover the key aspects of the query?
```

```
4. Coherence: Is the answer well-organized and logically structured?
5. Citation Quality: Are references cited appropriately and accurately?
Output format (strict JSON):
{"relevance": <int>, "faithfulness": <int>, "completeness": <int>,
"coherence": <int>, "citation_quality": <int>, "overall": <float>,
"brief_justification": "<1-2 sentences>"}
where "overall" is the average of the five scores.
—
Query: {query}
Reference Literature: {ref_text}
Model Answer: {answer}
```

C.5 Ablations and Training Curves

Contributions of the LAAE components. Table 11 ablates the two design choices in Eq. 6—the gate $\tilde{m}_g$ and the sign-symmetric scaling across positives and negatives—under prefix-conditioned training on Qwen3-4B. Removing the gate (*Ungated scaling*) lets length-aware updates fire indiscriminately and overcorrects toward longer outputs ($+19.7\%$). Applying scaling only to positives (*Positive-only scaling*) or replacing the continuous gate with binary activation (*Binary gate*) also fails to prevent the opposite drift, with continuations becoming substantially shorter (-24.3% and -27.2%). Only the full design keeps the mean positive continuation length close to the base model ($+0.7\%$).

Figure 4 shows the Qwen3-4B ablation dynamics behind Table 11. Full LAAE remains comparatively stable, whereas reward-side penalties and other ablations induce either length decay or over-correction toward longer outputs.

Table 11: LAAE ablation on Qwen3-4B under prefix-conditioned training. The full design is the only variant that simultaneously keeps repetition low and keeps mean positive continuation length close to Base.

Variant	**Rep. (%)** $\downarrow$	n**-gram (%)** $\downarrow$	**Len. diff. (%)** $\rightarrow 0$
PrefixRLVR (full)	**0.07**	**3.35**	**+0.7**
Binary gate	<u>0.09</u>	<u>3.54</u>	-27.2
Ungated scale	0.12	4.86	<u>+19.7</u>
Positive-only	**0.07**	3.70	-24.3

Training Curves. Figure 5 collects the training curves for all model scales. We track both reward and positive continuation response length because the central optimization risk is not only failure to reduce repetition, but reducing it by collapsing continuation length.

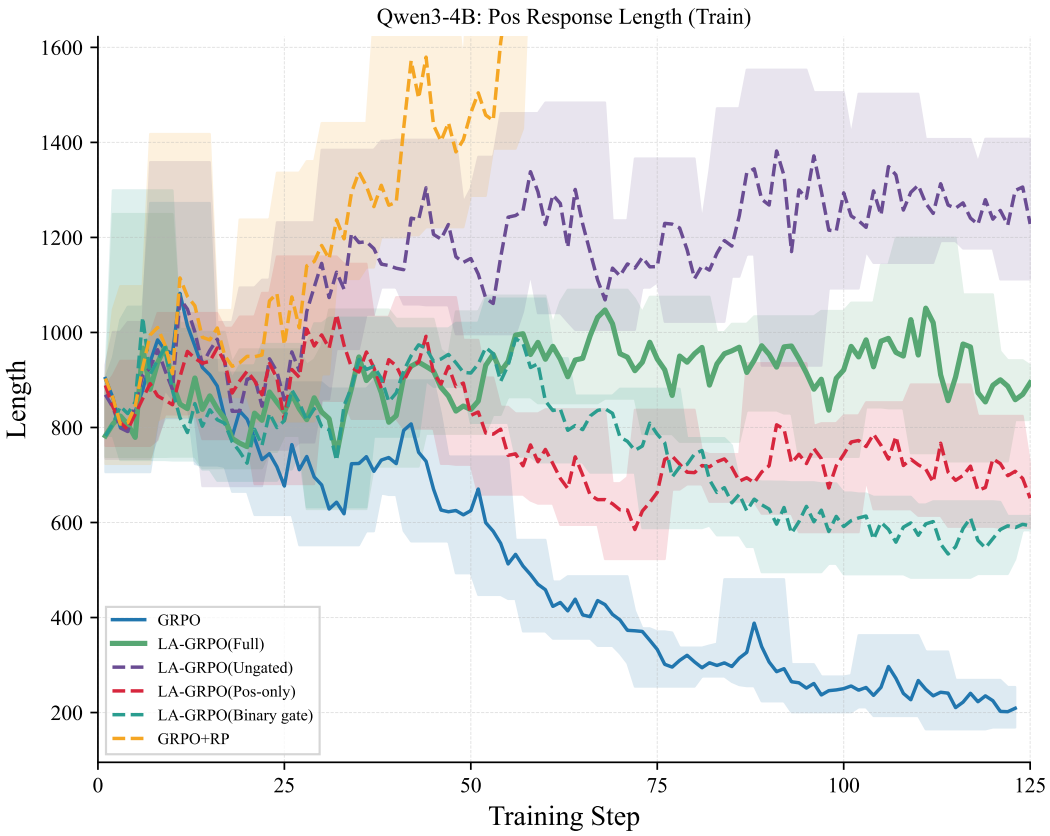
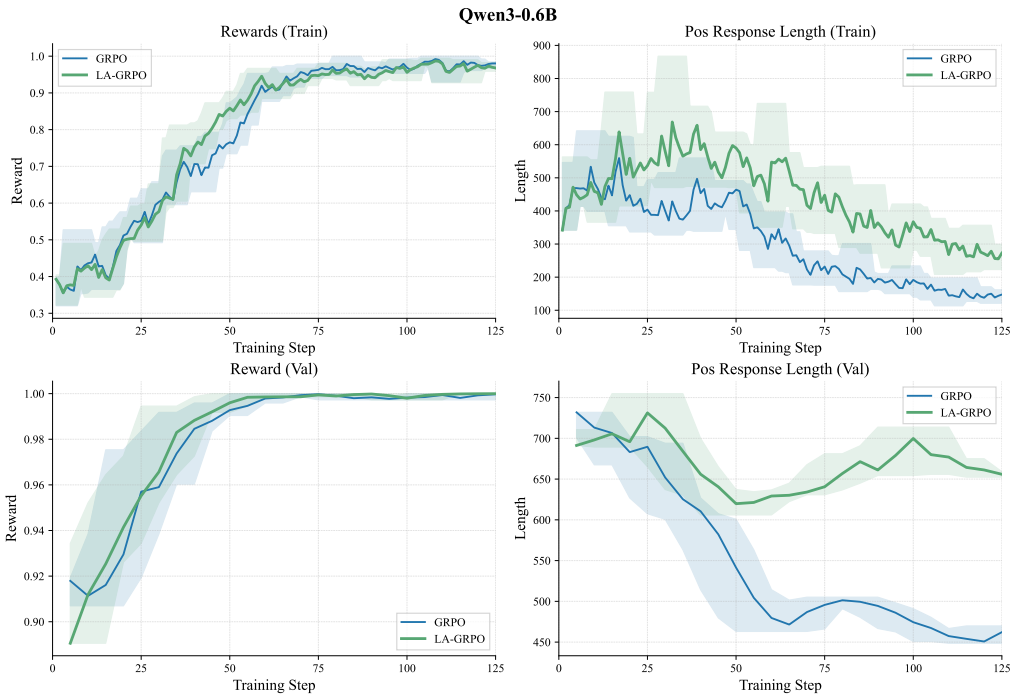
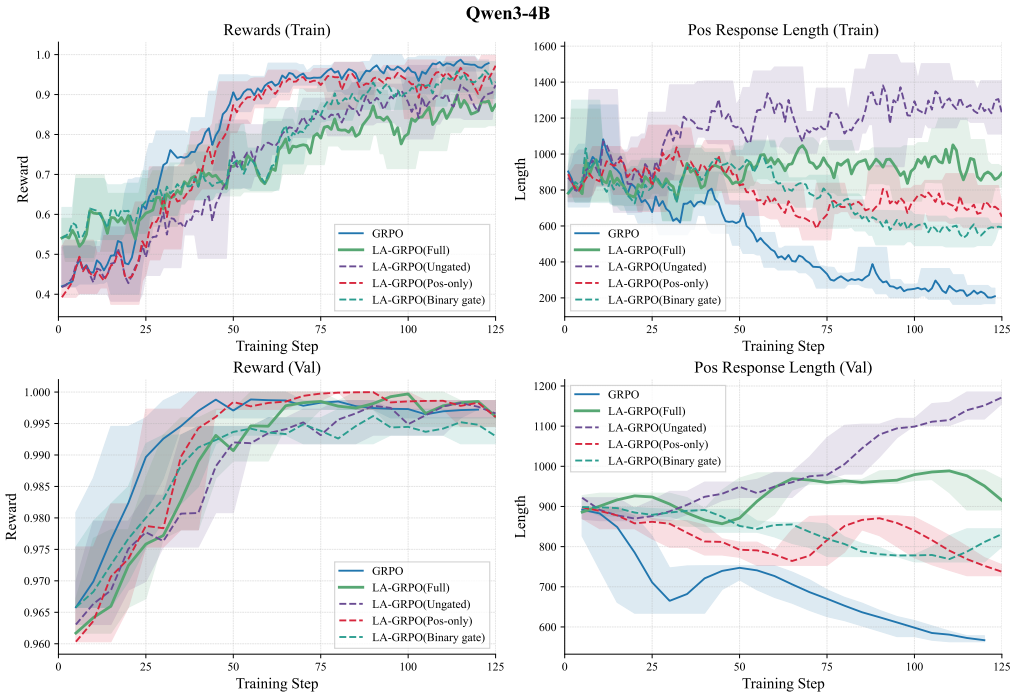


Figure 4: Qwen3-4B positive-length dynamics under PREFIXRLVR training. Full LAAE remains comparatively stable, whereas reward-side penalties and other ablations induce either length decay or over-correction toward longer outputs.

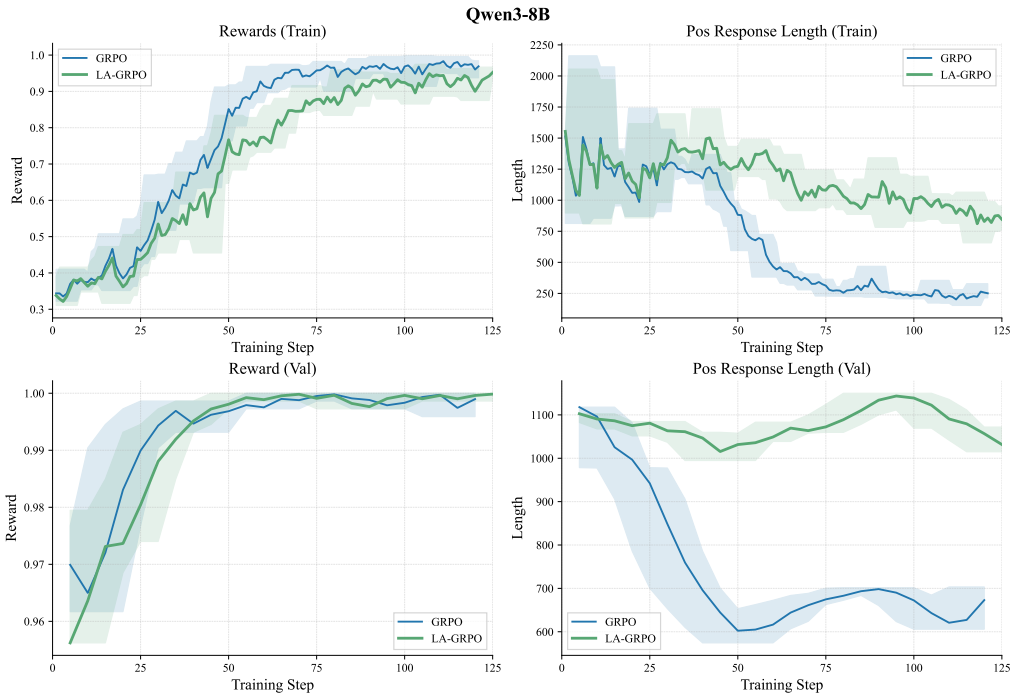


(a) Qwen3-0.6B

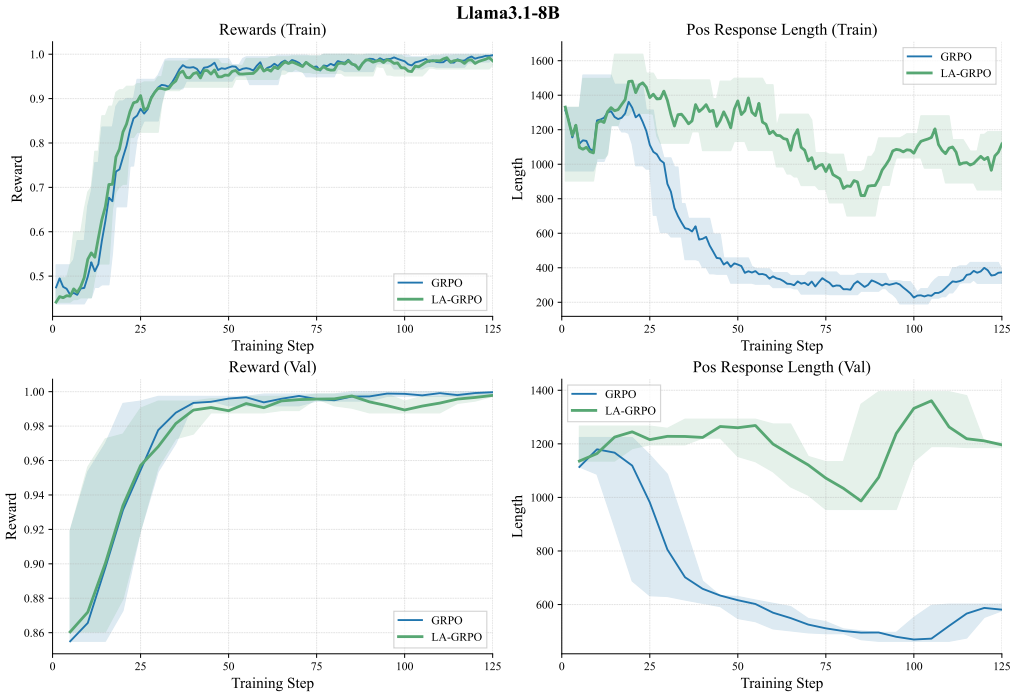


(b) Qwen3-4B

Figure 5: Training curves for all model scales. Each panel shows training reward, training positive response length, validation reward, and validation positive response length. Solid lines are exponentially smoothed; shaded regions indicate min–max ranges within a sliding window.



(c) Qwen3-8B



(d) Llama-3.1-8B

Figure 5: Training curves for all model scales (continued). The plotting conventions are the same as in panels (a) and (b).

D Additional Details on Trigger-Prefix Analysis

This section supplements the behavioral and hidden-state analyses in Section 2 with experimental settings, full trajectories, and control results.

D.1 Boundary-Aligned Behavioral Trajectories

We use the local copy rate and distinct-2 metrics defined in the main text. Both metrics are computed in 128-word windows centered at offsets $\{-448, -320, -192, -64, +64\}$ words from the trigger-prefix boundary.

For each model, we use a fixed set of queries with valid data at all five window positions. All included queries pass the context-cropping and token-slice checks. To compute local copy rate, each word is compared with up to 80 preceding words within the same window.

Figure 6 shows the full behavioral trajectories. Between the windows centered at -64 and $+64$ words, local copy rate increases and distinct-2 decreases in all three models. The paired 95% confidence intervals for both changes exclude zero.

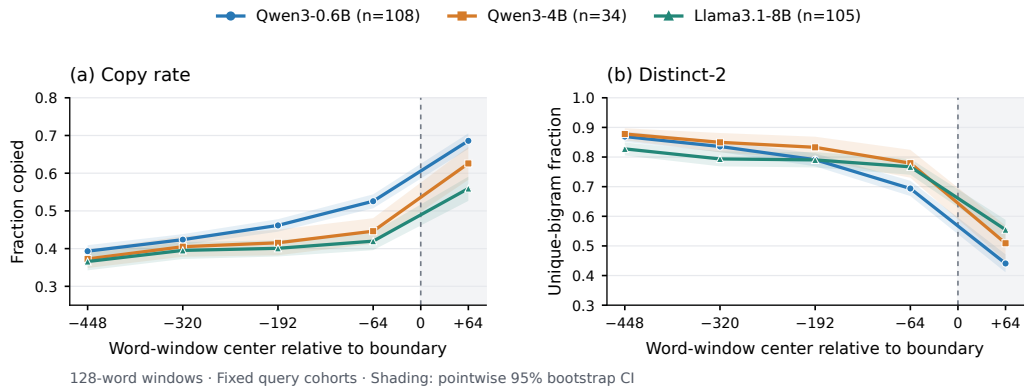


Figure 6: **Local copying and lexical diversity around the trigger-prefix boundary.** Curves show means over a fixed set of queries for each model. Shading shows pointwise 95% confidence intervals from query-level bootstrap resampling. The horizontal axis gives the word offset of each window center from the boundary; the dashed line marks the boundary.

D.2 Hidden-State Analysis Details

We compute hidden-state signals using Equation 1. To reduce the influence of any single token near the boundary, we fit each direction over the calibration window $W = \{-8, -4, -2, -1, 0\}$ tokens.

Cross-fitting splits the data by query, with all responses to a query assigned to the same fold. For each test fold, the direction and normalization scale are estimated from the remaining folds. In the cross-model prefix analysis, Llama calibration queries do not overlap with the test queries that use Qwen-generated prefixes.

The Qwen analyses use cached hidden states. For Llama, we extract the output states of decoder blocks using the evaluation prompts. The sampled layers are listed below. Indices are zero-based, and the final decoder block is excluded.

Model	Sampled layer indices
Qwen3-0.6B	8, 16, 19, 22, 25
Qwen3-4B	10, 21, 25, 29, 32
Llama-3.1-8B	9, 19, 22, 26, 29

Legends identify layers by relative depth $(\ell + 1)/L$, where L is the total number of decoder blocks. Figure 7 provides the hidden-state profiles for both Qwen models, which show stronger repetition-associated signals near the trigger-prefix boundary.

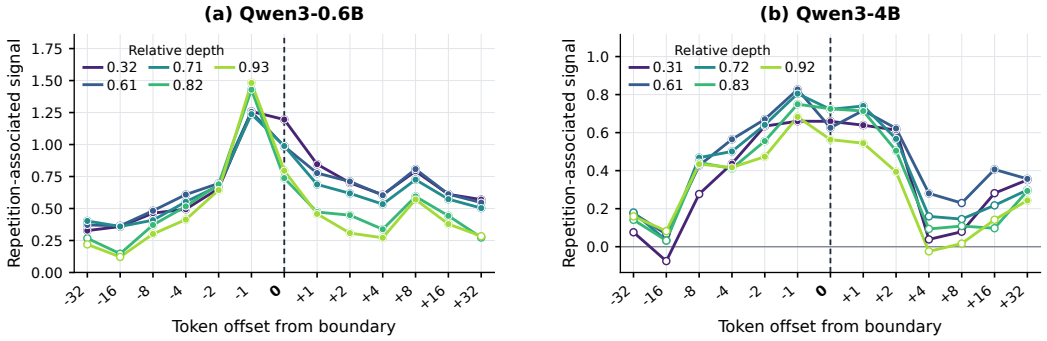


Figure 7: **Hidden-state signals in Qwen models.** Curves show mean normalized signals on held-out queries at different decoder depths. Filled markers indicate pointwise 95% confidence intervals entirely above zero; hollow markers indicate intervals that include zero. The horizontal axis gives token offsets from the trigger-prefix boundary, with sampled positions displayed at equal spacing.

D.3 Boundary-Shift Control

To examine how the signal peak relates to the trigger boundary, we shift the analysis boundary backward and forward by 64 tokens for Llama on the same queries. The paired responses and query folds remain unchanged.

Each condition refits its direction over W around its designated boundary. We normalize using the original-boundary calibration scale for the corresponding fold and layer. Local prominence is the signal at the designated boundary minus the mean signal at neighboring offsets $B = \{-32, -16, +16, +32\}$ tokens.

Within each query, we average both the signal and local prominence across the five sampled Llama layers before aggregating across queries.

Figure 8 shows the signal profiles for the three conditions. Local prominence at the original boundary exceeds the mean of the two shifted conditions by 0.954, with a paired 95% confidence interval of $[0.757, 1.152]$.

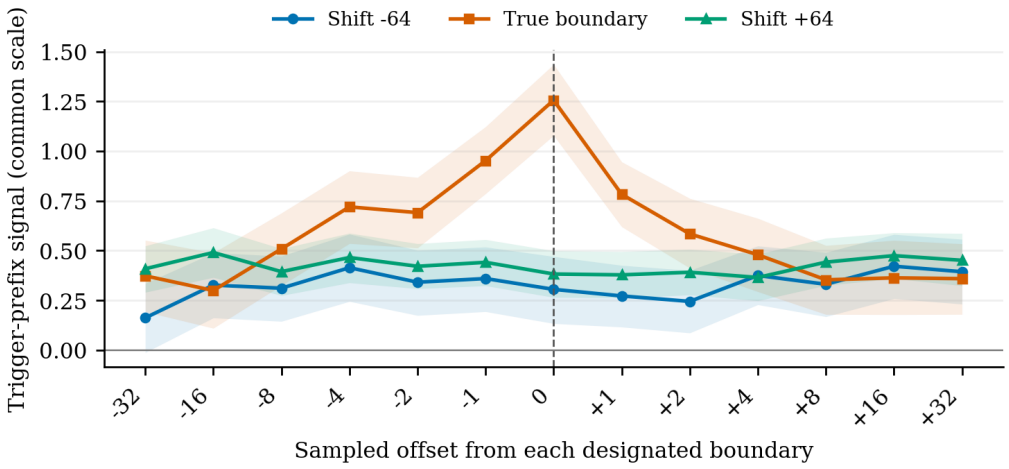


Figure 8: **Boundary-shift control in Llama.** Curves show mean signals on held-out queries; shading shows pointwise 95% confidence intervals from query-level bootstrap resampling. Offsets are relative to the designated boundary in each condition, so the three zero positions correspond to different absolute token positions. Sampled offsets are displayed at equal spacing.

D.4 Detection Thresholds and Boundary Signals

For Llama, we compare the default four-occurrence rule with a three-occurrence rule on the same queries. We hold the paired responses, query folds, and all other detector parameters fixed. The sampled layers, calibration window, and local prominence measure follow Appendix D.3. Both rules use the calibration scale from the default rule.

We compare two settings: refitting the direction around the boundary identified by each rule, and using the fixed direction fitted under the default rule. Table 12 reports the difference in mean local prominence between the two rules. The default rule yields higher local prominence in both settings.

Table 12: Paired differences in mean local prominence between the two detection rules. We define $\Delta = P_{\text{four}} - P_{\text{three}}$, where P denotes mean local prominence.

Direction setting	Δ	95% confidence interval
Separately fitted directions	0.323	$[0.091, 0.562]$
Fixed default direction	0.268	$[0.027, 0.504]$

Statistical methods. We estimate confidence intervals by bootstrap resampling the collected data at the query level. We use 5,000 resamples for behavioral trajectories and control experiments, and 4,000 for the Qwen hidden-state profiles. All intervals are 95% confidence intervals; curve intervals are computed separately at each sampled position. The fitted directions remain fixed during bootstrap resampling for hidden-state analyses.

E EXAMPLES

In this section, we provide examples illustrating different types of repetition in generated texts, along with examples of training inputs for different training methods.

E.1 REPETITION EXAMPLES

Example of *Repetition Response* 1

[Query] Fs according iso26262 with adas radar systems
[Failure Response]
ISO 26262 is a standard that deals with the functional safety of the E/E (Electric and Electronic) components of a road vehicle. It provides requirements to avoid or reduce the risk caused by these systems. The ISO 26262 standard is an automotive-specific interpretation of the basic safety standard IEC 61508 for functional safety of electrical/electronic (E/E) systems. The ISO 26262 standard provides an automotive safety lifecycle and provides a risk-based approach to determine Automotive Safety Integrity Levels (ASIL). ASILs are used to specify the applicable requirements of ISO 26262 in order to avoid unreasonable residual risk. Defined confirmation measures must be performed to ensure that a sufficient and acceptable level of safety has been achieved [1].
The ISO 26262 standard is also used for the development of ADAS systems. It is used to ensure that the systems are safe and reliable. The standard is used to ensure that the systems are safe and reliable. The standard is used to ensure that the systems are safe and reliable. The standard is used to ensure that the systems are safe and reliable [2].
The ISO 26262 standard is used to ensure that the systems are safe and reliable. The standard is used to ensure that the systems are safe and reliable. The standard is used to ensure that the systems are safe and reliable. The standard is used to ensure that the systems are safe and reliable [3].
The ISO 26262 standard is used to ensure that the systems are safe and reliable. The standard is used to ensure that the systems are safe and reliable. The standard is used to ensure that the systems are safe and reliable. The standard is used to ensure that the systems are safe and reliable [4].

Example of *Repetition Response* 2

[Query] Activated carbon fiber high surface area conductivity REE concentration separation
[Failure Response]
The literature discusses various methods for separating and purifying rare earth elements (REEs). One notable method is **supported liquid membrane extraction**, which uses different configurations such as hollow fiber supported liquid membrane (HFSLM), renewal liquid membrane (HFRLM), and emulsion pertraction technology. These methods are applied in pilot scale to recover REEs from apatite concentrate, using synthetic feed solutions with compositions based on a process for recovery of REEs from apatite concentrate. The experiments show that the membrane performance is governed by both the kinetics of REE transport and the thermodynamic properties, with the HFRLM showing higher transport rates than HFSLM, while the HFSLM configuration gives higher selectivity for individual REEs. The membrane performance in HFSLM rapidly decays with time, while in HFRLM and EPT configurations, the performance is more stable. Gel formation is identified as a significant factor in the decay of membrane performance, with gel formation observed at organic loading above 46% for Nd, 38% for Y, 46% for Dy, and 65% for Er.
Another method involves the use of **adsorption** and **solvent extraction** to recover REEs. For example, **activated carbon modified with potassium permanganate ** was investigated, and the adsorption capacity of the modified activated carbon was found to be effective for REEs. The **carboxylate-functionalized mesoporous carbon (CMC)** was also used to separate REEs, with results showing that CMC can extract REEs efficiently. Additionally, **solvent extraction** using ionic liquids has been explored to separate REEs from apatite concentrate, with experiments showing that the separation efficiency is improved by using D2EHPA and 2-ethylhexylphosphonic acid (HEH(EHP)) as the organic membrane solution.
The **ion interaction chromatography (IIC)** method using diglycolic acid (ODA) as a complexing agent is also discussed. This method is particularly useful for separating REEs by ion exchange and has been used to determine their concentrations in aqueous solutions. The experiments indicate that the affinity of REEs with complexing agents increases with their oxidation state and that the separation order is complex, with Sc < La < Ce < Lu < Pr < Yb < Nd < Tm < Sm < Eu < Pr < Yb < Nd < Tm < Sm < Eu < Er < Yb < Nd < Tm < Sm < Eu < Er < Yb < Nd < Tm < Sm < Eu < Er

Example of *Repetition Response* 3

[Query] Compare different device structure of SiC FET, including planar, trench and IGBT and explain their performance advantage/disadvantage, technology maturity and manufacturing challenges
[Failure Response]
The knee prosthesis design is described in several references, which provide insights into various aspects of the design and development of knee prosthesis systems. Here are some key points based on the literature:
1. **Design and Experimental Research**: The design and experimental research of a knee joint prosthesis involves a six-bar mechanism, which is a vector illustration of a 6-bar mechanism. The vector model helps in understanding the mechanical behavior of the knee joint. This is detailed in [1].
2. **Control of an Electrically Powered Knee Prosthesis**: A paper discusses the design and control of an electrically powered transfemoral prosthesis. It uses impedance-based control to coordinate the interaction between the prosthesis and the user during level walking. The control methodology is implemented on the prosthesis, and experimental results show effectiveness in level walking. This is detailed in [2].
3. **Synthesis of a Non-Grashof Six-Bar Polycentric Knee Prosthesis**: The synthesis of a non-grashof six-bar polycentric knee prosthesis using an evolutionary optimization algorithm is described. This approach considers both passive and powered designs, aiming to achieve a biomechanically sound prosthesis. The synergistic combination of these two perspectives is necessary for a successful design. This is detailed in [3].
4. **Design of a Pneumatically Actuated Transfemoral Prosthesis**: Another reference discusses the design and control of a pneumatically actuated transfemoral prosthesis, which includes components such as springs and actuators. This is detailed in [10].
5. **Design of a Pneumatically Actuated Transfemoral Prosthesis**: The design and control of a pneumatically actuated transfemoral prosthesis is detailed in [10].
6. **Design of a Pneumatically Actuated Transfemoral Prosthesis**: The design and control of a pneumatically actuated transfemoral prosthesis is detailed in [10].
7. **Design of a Pneumatically Actuated Transfemoral Prosthesis**: The design and control of a pneumatically actuated transfemoral prosthesis is detailed in [10].
8. **Design of a Pneumatically Actuated Transfemoral Prosthesis**: The design and control of a pneumatically actuated transfemoral prosthesis is detailed in [10].
9. **Design of a Pneumatically Actuated Transfemoral Prosthesis**: The design and control of a pneumatically actuated transfemoral prosthesis is detailed in [10].
10. **Design of a Pneumatically Actuated Transfemoral Prosthesis**: The design and control of a pneumatically actuated transfemoral prosthesis is detailed in [10].
11. **Design of a Pneumatically Actuated Transfemoral Prosthesis**: The design and control of a pneumatically actuated transfemoral prosthesis is detailed in [10].
The references highlight the importance of considering biomechanical factors, such as the use of springs, actuators, and control mechanisms, in designing knee prosthesis systems. These designs aim to provide improved stability, natural movement, and user comfort. The literature also emphasizes the need for customization and individualization to address the unique needs of amputees.

E.2 Training Input Examples Of The Same Query

Prompt only GRPO input example (No-Prefix)

$<$|**im_start**|$>$**user**
Please answer the question: **Compare different device structure of SiC FET, including planar, trench and IGBT and explain their performance advantage/disadvantage, technology maturity and manufacturing challenges** by referencing the provided literature:
...
$\{"text" : ..., "source" : ...\}$
$\{"text" : ..., "source" : ...\}$
...
Requirements:
Your output must be detailed. You must answer based only on the facts presented in the reference literature; do not fabricate any information. You must include citations to the corresponding reference numbers in your answer, e.g., [1].
$<$|**im_end**|$>$
$<$|**im_start**|$>$**assistant**

Trigger Prefix GRPO input example (PGR)

$<$|**im_start**|$>$**user**
Please answer the question: **Compare different device structure of SiC FET, including planar, trench and IGBT and explain their performance advantage/disadvantage, technology maturity and manufacturing challenges** by referencing the provided literature:
...

{"$text$" : ..., "$source$" : ...}
{"$text$" : ..., "$source$" : ...}
...
Requirements:
Your output must be detailed. You must answer based only on the facts presented in the reference literature; do not fabricate any information. You must include citations to the corresponding reference numbers in your answer, e.g., [1].
< |**im_end**| >
< |**im_start**| >**assistant**
The knee prosthesis design is described in several references, which provide insights into various aspects of the design and development of knee prosthesis systems. Here are some key points based on the literature:
1. **Design and Experimental Research**: The design and experimental research of a knee joint prosthesis involves a six-bar mechanism, which is a vector illustration of a 6-bar mechanism. The vector model helps in understanding the mechanical behavior of the knee joint. This is detailed in [1].
2. **Control of an Electrically Powered Knee Prosthesis**: A paper discusses the design and control of an electrically powered transfemoral prosthesis. It uses impedance-based control to coordinate the interaction between the prosthesis and the user during level walking. The control methodology is implemented on the prosthesis, and experimental results show effectiveness in level walking. This is detailed in [2].
3. **Synthesis of a Non-Grashof Six-Bar Polycentric Knee Prosthesis**: The synthesis of a non-grashof six-bar polycentric knee prosthesis using an evolutionary optimization algorithm is described. This approach considers both passive and powered designs, aiming to achieve a biomechanically sound prosthesis. The synergistic combination of these two perspectives is necessary for a successful design. This is detailed in [3].
4. **Design of a Pneumatically Actuated Transfemoral Prosthesis**: Another reference discusses the design and control of a pneumatically actuated transfemoral prosthesis, which includes components such as springs and actuators. This is

SFT input example

< |**im_start**| >**user**
Please answer the question: **Compare different device structure of SiC FET, including planar, trench and IGBT and explain their performance advantage/disadvantage, technology maturity and manufacturing challenges** by referencing the provided literature:
...
{"$text$" : ..., "$source$" : ...}
{"$text$" : ..., "$source$" : ...}
...
Requirements:
Your output must be detailed. You must answer based only on the facts presented in the reference literature; do not fabricate any information. You must include citations to the corresponding reference numbers in your answer, e.g., [1].
< |**im_end**| >
< |**im_start**| >**assistant**
The device structures of silicon carbide (SiC) power devices, including planar, trench, and IGBT (Insulated Gate Bipolar Transistor) structures, each have distinct characteristics that influence their performance, reliability, and manufacturing challenges. Below is a detailed comparison of these structures, based on the provided references:
1. **Planar Gate Structure** - **Performance Advantages**: The planar gate structure is simpler in design and has been widely used in SiC power devices. It offers good on-state voltage drop (VON) and is suitable for applications where high voltage blocking and low on-state losses are required. The planar structure is also beneficial for reducing the gate oxide electric field in off-state, which can improve reliability [2]. - **Performance Disadvantages**: The planar gate structure has a lower channel density compared to trench gate structures, leading to higher on-resistance and lower current density. Additionally, the planar structure is prone to higher gate oxide electric field in off-state, which can cause reliability issues [9]. - **Technology Maturity**: The planar gate structure has been well-established and is widely used in SiC power devices, with a long history of research and commercial application [1]. - **Manufacturing Challenges**: The planar gate structure is easier to manufacture, but it may not be as efficient in terms of power density and switching speed compared to trench gate structures. Additionally, the planar structure may have higher manufacturing costs due to the need for precise gate oxide layer processing [12].
2. **Trench Gate Structure** - **Performance Advantages**: The trench gate structure offers a higher channel density, which results in lower on-resistance and higher current density. This makes it suitable for high power applications where high current and low resistance are required. The trench gate structure also allows for better conductivity modulation, which reduces conduction losses [9]. - **Performance Disadvantages**: The trench gate structure is more complex and has a higher risk of gate oxide electric field concentration at the trench corners, which can lead to premature breakdown. Additionally, the trench gate structure has higher manufacturing complexity and may have lower reliability due to the increased stress on the gate oxide [9].

- **Technology Maturity**: The trench gate structure is a more recent development in SiC power devices, with significant research and commercial applications. However, the technology is still evolving, and there are ongoing efforts to improve its reliability and performance [12]. - **Manufacturing Challenges**: The trench gate structure requires more complex manufacturing processes, including deep trench etching and gate oxide layer deposition. The trench gate structure also has higher manufacturing costs and lower yields due to the complexity of the process [12].
3. **IGBT Structure** - **Performance Advantages**: The IGBT structure combines the advantages of both bipolar and MOSFET structures, offering high conductivity and low on-state resistance. The IGBT structure is suitable for high voltage and high current applications, and it has a wide safe operating area (SOA) that allows for higher reliability [1]. - **Performance Disadvantages**: The IGBT structure has higher switching losses compared to MOSFET structures, and it is more complex to manufacture. Additionally, the IGBT structure is more sensitive to gate oxide electric field concentration, which can lead to reliability issues [9]. - **Technology Maturity**: The IGBT structure is well-established and has been widely used in power electronics applications. However, the IGBT structure is less efficient in terms of switching speed and power density compared to MOSFET structures [1]. - **Manufacturing Challenges**: The IGBT structure requires more complex manufacturing processes, including the formation of the p-body and the gate oxide layer. The IGBT structure also has higher manufacturing costs and lower yields due to the complexity of the process [12].
4. **Comparison of Performance, Technology Maturity, and Manufacturing Challenges** - **Performance**: The trench gate structure offers the best performance in terms of on-resistance and current density, but it has higher switching losses and reliability issues. The planar gate structure offers good on-state voltage drop and reliability, but it has lower current density and higher on-resistance. The IGBT structure offers a balance between performance and reliability, but it has higher switching losses and is more complex to manufacture [9].
- **Technology Maturity**: The planar gate structure is the most mature, with a long history of research and commercial application. The trench gate structure is less mature, but it has significant research and commercial applications. The IGBT structure is well-established, but it is less efficient in terms of switching speed and power density [12]. - **Manufacturing Challenges**: The trench gate structure has the highest manufacturing challenges, including complex etching processes and gate oxide layer deposition. The planar gate structure is easier to manufacture, but it may not be as efficient in terms of power density. The IGBT structure has higher manufacturing costs and lower yields due to the complexity of the process [12].
5. **Conclusion** The choice of device structure for SiC power devices depends on the specific application requirements. The trench gate structure offers the best performance in terms of on-resistance and current density, but it has higher switching losses and reliability issues. The planar gate structure offers good on-state voltage drop and reliability, but it has lower current density and higher on-resistance. The IGBT structure offers a balance between performance and reliability, but it has higher switching losses and is more complex to manufacture. The technology maturity and manufacturing challenges vary for each structure, with the planar gate structure being the most mature and the trench gate structure being the most complex [9].