Prefix Likelihood-Ratio Control: Tail-Stable Training for Long-Horizon Language Generation

Yangyang Liu
Independent Researcher
Beijing, China
yangyangliu2@acm.org

Abstract

Token-level maximum likelihood and mean negative log-likelihood dominate the training and evaluation of large language models (LLMs), yet long-horizon generation often suffers from late-stage degradation despite improved mean loss. We identify two coupled causes of this mismatch: mean token loss is not scale-free with respect to sequence length, so small per-token divergences can amplify into large sequence-level distribution shifts, and long-horizon failures are spike-dominated, where rare but abrupt prefix deviations trigger errors that are largely invisible to mean objectives. To address these issues, we propose **Prefix Likelihood-Ratio Control** (PLRC), which centers training on the prefix log-likelihood ratio process and its increments to directly capture sequence-level deviation and spikes. As the data likelihood is unknown, PLRC learns a prefix ratio critic via noise-contrastive prefix discrimination and introduces tail-sensitive regularizers on the terminal ratio and its increments, yielding guarantees that bound long-horizon event distortion and the probability of the first spike. PLRC preserves standard teacher-forcing training without architectural changes while explicitly targeting failure modes that mean token loss cannot capture.

1 Introduction

Large language models (LLMs) are almost universally trained and compared under the same paradigm: mean token-level negative log-likelihood (often reported as perplexity) Bengio et al. [2003], LeCun et al. [2015]. Yet the dominant use cases of today's LLMs, including multi-turn dialogue, extended code synthesis, and long-context reasoning, expose a persistent mismatch between what we optimize and what we expect. Some works observed that a model with slightly better perplexity can still generate worse long-horizon outputs Holtzman et al. [2019], Welleck et al. [2019a]: it may remain fluent early on, then degrade later, contradict earlier commitments, violate formatting or tool-call constraints, or collapse into repetition. These failures arise in the very regimes where LLMs are expected to be most useful.

This paper argues that the mismatch is structural with respect to sequence length and abrupt spikes (see in Figure 1). The first issue is that mean token loss is not *scale-free* from sequence length. A small per-token improvement can look meaningful when summarized as an average, yet over long horizons the resulting sequence distribution can still drift Ross et al. [2011] far enough to change the probability of important events, such as maintaining a consistent variable binding in code, adhering to a strict output schema, or preserving a chain of reasoning without contradictions. In other words, the metric we routinely track can remain almost unchanged while the long-horizon behavior of the model changes dramatically. This helps explain why perplexity gains sometimes fail to translate into better long-form generation.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

The second issue is that long-horizon failures are often *spike-dominated* rather than mean-dominated. Many failures are caused by a few steps that abruptly derail the trajectory Holtzman et al. [2019]: a mistaken variable name can cause cascading errors in code, and one incorrect premise can poison a long reasoning trace (we show cases in Appendix H). After such a deviation, later tokens are generated from a prefix that no longer resembles those seen in the data, making recovery unlikely. Mean token loss can hide this mechanism because it averages over all positions; a handful of catastrophic steps can be "washed out" by thousands of easy tokens. This suggests that improving robustness at long horizon requires controlling not only average predictive accuracy, but also the tail behavior of trajectory-level deviations and the frequency of abrupt deviations.

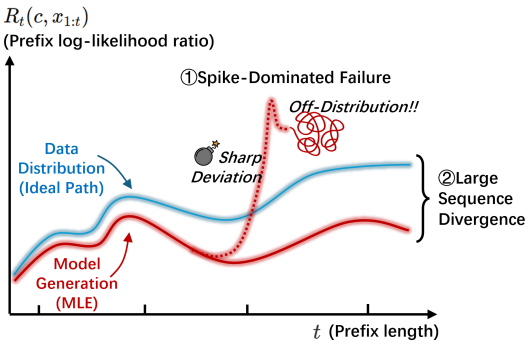

Figure 1: The two failure mechanisms of MLE with respect to prefix log-likelihood ratio in long-horizon generation.

Motivated by these observations, we propose a different training principle: instead of treating long-horizon failures as the byproduct of exposure bias or error accumulation, we explicitly model and control how the model's prefix distribution deviates from the data's prefix distribution along the trajectory. We introduce **Prefix Likelihood-Ratio Control (PLRC)**, which centers training on a trajectory-level deviation signal: the extent to which prefixes are disproportionately likely under the model relative to the data. Because the data prefix likelihood is unknown, PLRC learns a prefix ratio critic using noise-contrastive discrimination between real prefixes and model-generated prefixes. This critic provides an estimate of prefix-level deviation that is directly tied to off-distribution drift. PLRC then augments standard teacher-forcing training with tail-sensitive regularizers that (i) suppress length-amplified sequence-level deviation and (ii) penalize abrupt, spike-like deviations that initiate long-horizon collapse. Importantly, PLRC does not modify the LLM architecture or decoding procedure; it changes the objective to better match the long-horizon reliability requirements.

Beyond proposing an algorithm, we provide a theoretical analysis on why the standard mean-loss view can fail at long horizon. Our analysis leverages change-of-measure arguments to connect trajectory-level deviation to event probability distortions Durrett [2019], Shalizi [2013]. This yields guarantees that the quantities controlled by PLRC upper-bound the probability of entering highly deviated prefixes and the likelihood of spike. The result is a principled framework for long-horizon robustness that targets the failure modes that perplexity cannot directly capture.

Our **key contributions** are as follows:

- **Problem formalization (scale and spikes):** We identify two concrete and coupled mechanisms underlying long-horizon failures in LLMs:
  - Mean token loss is not scale-free with sequence length, so small per-token gaps can correspond to large sequence-level distribution shifts.
  - Failures are often spike-dominated, initiated by abrupt deviations that mean objectives obscure.
- **Method (PLRC):** We propose *Prefix Likelihood-Ratio Control*, which learns a prefix ratio critic via noise-contrastive prefix discrimination and introduces tail-sensitive regularizers that directly control length-amplified trajectory and spike-induced deviations, without changing model architectures or inference.
- **Theory (event distortion guarantees):** We provide change-of-measure–based guarantees linking PLRC's controlled quantities to upper bounds on long-horizon event probability distortions and derailment risks. This offers a principled explanation for why improving perplexity may not improve long-horizon reliability—and how PLRC addresses this gap.

2 Related Works

We introduce several existing paradigms for improving long-horizon LLM or language model behavior.

Loss reweighting and token-/domain-level importance in language modeling. Prior work argues that uniform next-token MLE can under-train rare/hard events and over-fit frequent/easy ones. Unlikelihood training penalizes undesirable tokens (e.g., repetitions) to counter likelihood-induced pathologies Welleck et al. [2019b]; FACE reweights token losses by corpus frequency to reduce over-confidence and improve diversity Jiang et al. [2019]; TLDR emphasizes hard tokens via difficulty-based weights to reduce repetition Jiang et al. [2020]; and MiLe Loss uses entropy-conditioned scaling to mitigate frequent/infrequent imbalance during pretraining Su et al. [2024]. For long-context learning, token-weighting based on comparing long- vs. short-context confidences allocates training signal to positions where long-range context matters Helm et al. [2025]. Beyond token-level schemes, DoReMi reweights data mixtures via group-robust objectives to improve pretraining efficiency and downstream performance Xie et al. [2023].

Instruction tuning, RLHF, and preference-based post-training objectives. Instruction tuning improves zero-shot generalization via supervised finetuning on diverse prompted tasks, exemplified by FLAN-style instruction tuning Wei et al. [2021] and multitask prompted training (e.g., T0 / T-Zero) Sanh et al. [2021]. Related work bootstraps instruction data with minimal human annotation, including Self-Instruct for generating and filtering synthetic instructions Wang et al. [2023] and Alpaca's widely used synthetic-instruction recipe Taori et al. [2023]. For alignment beyond SFT, RLHF (notably InstructGPT) combines supervised finetuning, reward modeling, and RL (e.g., PPO) to optimize human preferences Ouyang et al. [2022]. To scale supervision, RLAIF-style methods leverage AI feedback: Constitutional AI uses principle-based self-critique and AI feedback in supervised and RL phases Bai et al. [2022], UltraFeedback provides large-scale preference data Cui et al. [2023], and empirical studies compare when AI feedback can match RLHF Lee et al. [2023]. More recently, preference optimization methods avoid on-policy RL by optimizing likelihood-ratio objectives on preference pairs, including DPO Rafailov et al. [2023], ORPO Hong et al. [2024], SimPO Meng et al. [2024], KTO Ethayarajh et al. [2024], and IPO/online preference optimization variants Calandriello et al. [2024].

Token importance for attribution, uncertainty, and selective generation. Related work measures which tokens or examples matter for interpretation, debugging, and selective deployment. Influence-based methods identify training instances affecting generations, with RapidIn scaling token-wise influence via gradient compression Lin et al. [2024]. Uncertainty and selective generation methods improve answer calibration through self-evaluation Ren et al. [2023], calibration tuning Kapoor et al. [2024], and conformal prediction for LLM uncertainty quantification Kumar et al. [2023], Xu and Lu [2025]. These works localize *where* risk concentrates, whereas PLRC focuses on *how* to reshape training to reduce tail failures.

3 Problem Analysis: Mean Token Loss Fails at Long Horizon

LLM training and evaluation are dominated by mean token loss (average negative log-likelihood). Concretely, for a continuation length T under context c, the standard objective minimizes:

$$L_{\text{MLE}}(\theta) = \mathbb{E}_{x \sim p^*(\cdot|c)} \left[\sum_{t=1}^{T} -\log p_\theta(x_t \mid c, x_{<t}) \right]. \tag{1}$$

c denotes the context, $x = (x_1, \ldots, x_T)$ a target sequence sampled from the data distribution $p^*(\cdot \mid c)$, and $x_{<t}$ the prefix up to step $t-1$. p_θ is the model with parameters θ. Models are typically compared via the per-token quantity $\frac{1}{T} L_{\text{MLE}}$. While statistically natural, it does not provide scale-free control over long-horizon generation. Two coupled failure modes arise once T becomes large.

1) Length Amplification: Small per-token gaps can imply large sequence-level divergence

Let $p^*(x_{1:T} \mid c)$ be the data distribution and $p_\theta(x_{1:T} \mid c)$ the model distribution. The joint KL divergence decomposes additively:

$$\begin{aligned} &\mathrm{KL}(p^*(x_{1:T} \mid c) \,\|\, p_\theta(x_{1:T} \mid c)) \\ &= \sum_{t=1}^{T} \mathbb{E}_{x_{<t} \sim p^*} \left[\mathrm{KL}(p^*(x_t \mid c, x_{<t}) \,\|\, p_\theta(x_t \mid c, x_{<t})) \right] \end{aligned} \tag{2}$$

Define the token-normalized KL (implicitly tracked by mean token loss):

$$\bar{k}_T(\theta; c) \triangleq \frac{1}{T} \mathrm{KL}(p^*(x_{1:T} \mid c) \,\|\, p_\theta(x_{1:T} \mid c)). \tag{3}$$

Even if $\bar{k}_T$ is small, the total divergence scales as $T\bar{k}_T$, which can still lead to substantial sequence-level mismatch. A consequence of Pinsker's inequality gives:

$$\begin{aligned} &\text{Total Variation}(p^*(x_{1:T} \mid c), p_\theta(x_{1:T} \mid c)) \\ &\triangleq \frac{1}{2} \sum_{x_{1:T}} \left| p^*(x_{1:T} \mid c) - p_\theta(x_{1:T} \mid c) \right| \\ &\leq \sqrt{\frac{1}{2} \mathrm{KL}(p^* \,\|\, p_\theta)} = \sqrt{\frac{T\bar{k}_T(\theta; c)}{2}}. \end{aligned} \tag{4}$$

Thus, holding per-token divergence $\bar{k}_T$ fixed while increasing T leads to a $\sqrt{T}$ growth in the upper bound on total variation. This formalizes a key long-context pathology: per-token metrics can look "nearly identical" while the induced sequence distributions differ enough to change the probability of some rare but critical events (e.g., violating a constraint, breaking a format, contradicting earlier statements). In other words, mean token loss does not directly control the long-horizon behavior that users care about.

2) Mean loss hides spike-dominated failures

Empirically, long-horizon generation failures rarely arise from a uniform degradation of token probabilities. Instead, they are often triggered by localized deviations — a few decisions that abruptly push the prefix into an off-distribution region Pang and He [2020], Bengio et al. [2015].

To formalize this, consider the prefix log-likelihood ratio process:

$$R_t(c, x_{1:t}) \triangleq \log \frac{p_\theta(x_{1:t} \mid c)}{p^*(x_{1:t} \mid c)}. \tag{5}$$

Define its increments:

$$\Delta_t(c, x_{1:t}) \triangleq R_t(c, x_{1:t}) - R_{t-1}(c, x_{1:t-1}), \quad R_0 \equiv 0. \tag{6}$$

A loss spike corresponds to a large positive Δ_t, indicating a step where the model overestimates likelihood relative to the data, which is a sign of prefix deviation. Importantly, such deviations can occur even when average token loss is small: a few large jumps Δ_t can dominate the tail behavior of R_T, while contributing only marginally to the mean.

This yields a second mismatch: mean token loss targets average conditional fit, whereas long-horizon robustness depends on controlling the tail and pathwise maxima of the deviation process, e.g.,

$$\max_{t \leq T} R_t \quad \text{and} \quad \max_{t \leq T} \Delta_t. \tag{7}$$

Objectives that only reweight high token losses Welleck et al. [2019b], Jiang et al. [2019, 2020] remain indirect: they do not explicitly constrain the sequence-level deviation that is amplified by length, nor the spike-like events that initiate long-horizon collapse.

In summary, average token loss can miss long-horizon failures: small per-token gaps can amplify with length, and rare deviation spikes can dominate outcomes. We therefore model and control the prefix likelihood-ratio process $\{R_t\}$ and its increments $\{\Delta_t\}$. Next, we present PLRC, which learns a prefix ratio critic and adds tail-sensitive regularizers based on exponential moments of R_T (sequence distortion) and Δ_t (spike-induced deviations).

4 Method: Prefix Likelihood-Ratio Control (PLRC)

4.1 Setup and Notation

We consider conditional sequence modeling with context c (e.g., prompt or dialogue history). Let $x_{1:T} \in \mathcal{X}^T$ denote a length-T continuation. The data distribution is $p^*(x_{1:T} \mid c)$, and the model is an autoregressive distribution:

$$p_\theta(x_{1:T} \mid c) = \prod_{t=1}^{T} p_\theta(x_t \mid c, x_{<t}). \tag{8}$$

For any prefix length t, define the prefix distributions $p_t^*(x_{1:t} \mid c)$ and $p_{\theta,t}(x_{1:t} \mid c)$ as the marginals induced by p^* and p_θ, respectively.

4.2 Prefix Log-Likelihood Ratio Process

Following Section 3, PLRC operates on the prefix log-likelihood ratio process R_t and its increments Δ_t. Large terminal values R_T capture length-amplified sequence-level deviation, while large positive increments Δ_t capture abrupt spike-like derailments. PLRC therefore controls terminal drift, local jumps, and optionally the pathwise maximum of prefix deviation.

4.3 Estimating the Unknown Data Prefix Probability via Ratio Critic

Since p^* is unknown, direct computation of the prefix log-likelihood ratio $R_t(c, x_{1:t}) = \log \frac{p_{\theta,t}(x_{1:t}|c)}{p_t^*(x_{1:t}|c)}$ is intractable. However, the model prefix log-probability is tractable under a causal LM:

$$\log p_{\theta,t}(x_{1:t} \mid c) = \sum_{i=1}^{t} \log p_\theta(x_i \mid c, x_{<i}). \tag{9}$$

Thus, it suffices to estimate $-\log p_t^*(x_{1:t} \mid c)$ up to an additive constant (which does not affect θ-gradients).

Critic architecture and parameterization. We introduce a *ratio critic* implemented as a small head on top of the LM representations. Given the same tokenized prefix $(c, x_{1:t})$ that the LM conditions on, we run a forward pass of the backbone LM and extract a prefix representation $h_t(c, x_{1:t}) \in \mathbb{R}^d$ from the last layer hidden states of the *prefix tokens* (mean-pooling over the prefix token hidden states). The critic is a lightweight MLP $g_\phi : \mathbb{R}^d \to \mathbb{R}$ that outputs a scalar prefix potential:

$$u_\phi(c, x_{1:t}) \triangleq g_\phi\big(h_t(c, x_{1:t})\big). \tag{10}$$

This design ensures u_ϕ conditions on exactly the same information as the LM at step t, while adding negligible computation compared to the backbone.

We then define the estimated prefix log-likelihood ratio as

$$\hat{R}_t(c, x_{1:t}) \triangleq \log p_{\theta,t}(x_{1:t} \mid c) + u_\phi(c, x_{1:t}). \tag{11}$$

If $u_\phi(c, x_{1:t}) \approx -\log p_t^*(x_{1:t} \mid c) + \text{const}$, then $\hat{R}_t(c, x_{1:t}) \approx R_t(c, x_{1:t}) + \text{const}$, where the constant does not affect training.

Training the critic via noise-contrastive prefix discrimination. We train u_ϕ to discriminate *model prefixes* from *data prefixes* by contrasting $q_t(\cdot \mid c) \equiv p_{\theta,t}(\cdot \mid c)$ against $p_t(\cdot \mid c) \equiv p_t^*(\cdot \mid c)$ under a balanced mixture:

Positive samples: $\tilde{x}_{1:t} \sim q_t(\cdot \mid c)$ (model rollouts),

Negative samples: $x_{1:t} \sim p_t(\cdot \mid c)$ (data prefixes).

Concretely, for each context c we sample a prefix length t, obtain a teacher-forced data prefix $x_{1:t}$ from the dataset, and generate a free-running model prefix $\tilde{x}_{1:t}$ from the current LM. We maximize

the logistic objective

$$\max_{\phi} \mathbb{E}_c\Big[\mathbb{E}_{\tilde{x}_{1:t}\sim q_t}\big[\log\sigma(\hat{R}_t(c,\tilde{x}_{1:t}))\big] \\ +\mathbb{E}_{x_{1:t}\sim p_t}\big[\log(1-\sigma(\hat{R}_t(c,x_{1:t})))\big]\Big]. \tag{12}$$

where $\sigma(z) = \frac{1}{1+e^{-z}}$.

4.4 PLRC Objective: Controlling Length Amplification and Spikes

We now define the full PLRC training objective.

(i) Base MLE:

$$L_{\text{MLE}}(\theta) = \mathbb{E}_{(c,x)\sim D}\left[\sum_{t=1}^{T} -\log p_\theta(x_t \mid c, x_{<t})\right] \tag{13}$$

(ii) Length amplification control via terminal ratio: We penalize the exponential moment of the terminal ratio:

$$R_{\text{len}}(\theta;\phi) \triangleq \log \mathbb{E}_{(c,x)\sim D}\left[\exp\left(\hat{R}_T(c,x_{1:T})\right)\right]. \tag{14}$$

This is tail-sensitive: penalizes large deviations that would otherwise dominate distribution mismatch.

(iii) Spike control via increments: Define estimated increments:

$$\hat{\Delta}_t(c,x_{1:t}) \triangleq \hat{R}_t(c,x_{1:t}) - \hat{R}_{t-1}(c,x_{1:t-1}). \tag{15}$$

We penalize the soft-maximum of positive jumps:

$$R_{\text{spike}}(\theta;\phi) \triangleq \log \sum_{t=1}^{T} \mathbb{E}_{(c,x)\sim D}\left[\exp\left([\hat{\Delta}_t(c,x_{1:t})]_+\right)\right]. \tag{16}$$

(iv) Path control via soft-max of prefix ratios:

$$R_{\text{path}}(\theta;\phi) \triangleq \log \mathbb{E}_{(c,x)\sim D}\left[\sum_{t=1}^{T} \exp\left(\hat{R}_t(c,x_{1:t})\right)\right]. \tag{17}$$

Final PLRC objective:

$$\min_{\theta} L_{\text{PLRC}} = L_{\text{MLE}} + \alpha(R_{\text{len}} + R_{\text{path}}) + \beta R_{\text{spike}}. \tag{18}$$

We reduce hyperparameter complexity by fixing the exponential-tilting temperatures to 1 and tying the weights of R_{len} and R_{path} for they are both exponential tail of the prefix likelihood-ratio process.

Importantly, $\widehat{R}_t(c,x_{1:t}) = \log p_{\theta,t}(x_{1:t} \mid c) + u_\phi(c,x_{1:t})$ contains the model term $\log p_{\theta,t}$, so when updating θ we can treat u_ϕ as fixed and obtain $\nabla_\theta \widehat{R}_t(c,x_{1:t}) = \nabla_\theta \log p_{\theta,t}(x_{1:t} \mid c)$.

In other words, the regularizer backpropagates through the usual log-probability of the model (no gradient needs to pass through the critic), while u_ϕ simply supplies an estimate of the missing $-\log p_t^*(x_{1:t} \mid c)$ term (up to a constant).

4.5 Distinction from GANs and Preference Optimization

Although PLRC uses a discriminator-trained ratio critic, it differs from GANs and preference optimization such as DPO Rafailov et al. [2023] and LongPO Chen et al. [2025]. Unlike preference methods that optimize sequence-level human preferences, PLRC targets statistical prefix-distribution drift that causes long-horizon collapse; its complementarity with LongPO shows that prefix-level stability is distinct from preference alignment. Unlike GAN discriminators, the critic estimates a log-density ratio and is used with stop-gradient, avoiding minimax instability. The MLE term anchors training, while R_{len} and R_{spike} suppress tail deviations and spike events, yielding the guarantees in Appendix G.

Benchmark / Metric	MLE	TokenW	ProLong	LongPO	PLRC	PLRC+ LongPO
LongWriter / Score ↑	74.09±[0.41]	75.03±[0.39]	75.53±[0.38]	76.59±[0.36]	**79.05**±[0.34]	**80.11**±[0.33]
GovReport / ROUGE-L ↑	54.74±[0.29]	55.21±[0.28]	55.81±[0.27]	56.01±[0.27]	**56.83**±[0.26]	**57.29**±[0.26]
GovReport / ROUGE-2 ↑	22.66±[0.21]	22.92±[0.20]	23.31±[0.20]	23.40±[0.20]	**24.17**±[0.19]	**24.42**±[0.19]
PubMed / ROUGE-L ↑	41.23±[0.24]	41.69±[0.24]	41.94±[0.23]	42.03±[0.23]	**42.65**±[0.22]	**42.93**±[0.22]
PubMed / ROUGE-2 ↑	18.25±[0.16]	18.44±[0.16]	18.71±[0.16]	18.84±[0.15]	**19.33**±[0.15]	**19.52**±[0.15]
LongGenBench / Pass@1 ↑	15.02±[0.58]	15.50±[0.57]	16.28±[0.56]	16.91±[0.55]	**18.55**±[0.53]	**19.33**±[0.52]

Table 1: **Main results** across benchmarks (**Qwen2.5-7B-Instruct**). LongGenBench is used *only* for evaluation (no training). TokenW: Token Weighting. Best results are in **bold**. All numbers are reported with bootstrap confidence intervals.

4.6 Optimization Procedure

PLRC alternates between critic and model updates. The critic u_ϕ is updated by prefix discrimination to estimate the model–data log-density ratio. The LM parameters θ are then updated with MLE plus PLRC regularizers computed from $\widehat{R}_t = \log p_{\theta,t} + u_\phi$, while keeping ϕ fixed so gradients flow only through $\log p_{\theta,t}$. Computational cost is reported in Appendix F.

5 Experiments

5.1 Experimental Setup

Model. We use four instruction-tuned LLM backbones: Qwen2.5-7B-Instruct and Qwen2.5-14B-Instruct Yang et al. [2025], Llama 3.1 8B Instruct Dubey et al. [2024], and Phi-3-Medium-128K-Instruct Abdin et al. [2024]. All methods use the same decoding configuration within each benchmark.

Benchmarks. We evaluate on four long-horizon benchmarks: **LongGenBench** Wu et al. [2024], **LongWriter** Bai et al. [2024], **GovReport** Huang et al. [2021], and **PubMed** Cohan et al. [2018]. Training data is strictly split-safe. For **GovReport** and **PubMed** we train only on their train splits (never using dev/test). For **LongWriter** we train only on its train split (never using eval/test). We use **LongGenBench** strictly for evaluation (and do not train on it) to avoid benchmark-specific adaptation and potential test-set contamination, ensuring it remains an unbiased measure of long-form robustness. As an additional safeguard, we perform lightweight near-duplicate filtering across all training corpora using an n-gram hashing deduplication pass to reduce the chance that training data accidentally contains evaluation text.

Baselines. We compare PLRC against four baselines spanning the main paradigms for improving long-horizon behavior: Standard MLE (token-level likelihood training), Token Weighting (token-level reweighting for hard-token emphasis) Helm et al. [2025], ProLong (long-context continued pretraining) Gao et al. [2025], and LongPO (preference optimization for long-context instruction following) Chen et al. [2025]. Together, these cover the most common methods in prior work: unchanged MLE, loss shaping, long-context continued training, and long-context alignment.

Further implementation details are in Appendix A.

5.2 Main Results

Table 1 reports the main results on four long-horizon benchmarks with Qwen2.5-7B-Instruct. We compare PLRC with MLE, Token Weighting, ProLong, and LongPO, covering standard likelihood training, token-level loss shaping, long-context continued training, and preference optimization.

PLRC improves performance across all benchmarks. The gains are largest on long-form generation tasks, where late-stage failures are more common. PLRC outperforms MLE and long-context training baselines on LongWriter, yields consistent ROUGE gains on GovReport and PubMed, and achieves the best Pass@1 on LongGenBench, which is used only for evaluation. Combining PLRC with LongPO further improves results, suggesting that prefix-deviation control is complementary to preference optimization.

Method	SpikeRate(τ) ↓	MaxJump ↓	MaxPrefixDev ↓	FirstFailurePos (tokens) ↑
LongGenBench (32K budget), Qwen2.5-7B-Instruct				
MLE	0.62%	8.91	41.76	6,520
TokenW	0.59%	8.56	40.35	6,980
ProLong	0.57%	8.23	39.64	7,410
LongPO	0.54%	7.90	38.81	7,960
PLRC	0.41%	6.52	32.48	10,980
PLRC+LongPO	**0.36%**	**6.01**	**30.79**	**12,140**
LongWriter-style prompts (targeted long writing), Qwen2.5-7B-Instruct				
MLE	0.48%	7.64	36.93	5,840
TokenW	0.46%	7.30	35.85	6,210
ProLong	0.44%	7.02	35.16	6,660
LongPO	0.42%	6.87	34.48	7,120
PLRC	0.32%	5.79	29.67	9,420
PLRC+LongPO	**0.29%**	**5.37**	**28.35**	**10,260**

Table 2: Spike diagnostics and first-failure localization (**Qwen2.5-7B-Instruct**). SpikeRate uses $\tau = 2.0$ (nats) over the generation horizon. FirstFailurePos is defined by the LongGenBench verifier and by automatic derailment detectors aligned with the LongWriter long-writing protocol.

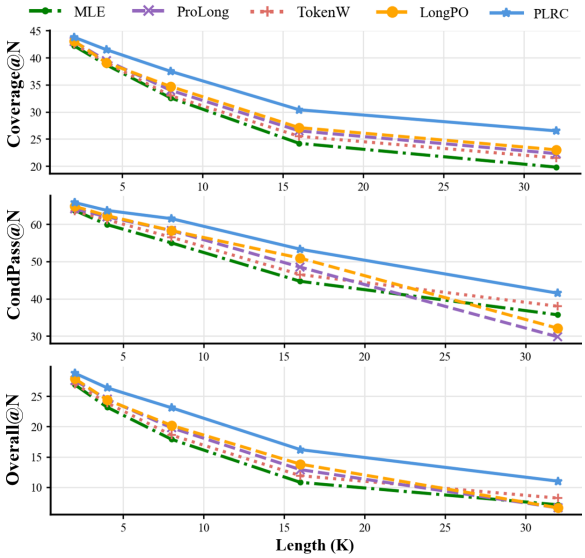

Figure 2: Prefix-bucket curves for visualizing length-scaling trends. Official LongGenBench metrics at 16K/32K are reported separately.

Additional results are provided in the appendix: cross-model robustness in Appendix B, loss-component ablations in Appendix C, human and AI-judge evaluation in Appendix D, and critic-robustness analysis in Appendix E.

5.3 Length-Scaling Robustness

To examine length-scaling robustness on LongGenBench, we evaluate each method under generation budgets $N \in \{2\text{K}, 4\text{K}, 8\text{K}, 16\text{K}, 32\text{K}\}$ and score subtasks whose verification targets appear within the first N tokens. We report **Coverage@N**, **CondPass@N**, and **Overall@N** $=$ Coverage@N $\times$ CondPass@N.

As shown in Figure 2, MLE degrades sharply as the horizon grows, while TokenW, ProLong, and LongPO offer modest gains. PLRC degrades more slowly, maintaining better coverage and conditional pass rate at long budgets and achieving the best Overall@32K. Official LongGenBench metrics are reported separately using the released protocol.

5.4 Spike Analysis

We analyze deviation jumps and first-failure positions to test whether long-horizon failures are spike-dominated. We test whether long-horizon failures are *spike-dominated*, i.e., triggered by a few sharp prefix deviations. We evaluate MLE, TokenW, ProLong, LongPO, PLRC, and PLRC+LongPO on LongGenBench (16K–32K) and LongWriter-style prompts.

Using PLRC's ratio process $\hat{R}^t$ and increments $\hat{\Delta}^t = \hat{R}^t - \hat{R}^{t-1}$, we report **SpikeRate**(τ), the fraction of steps with $\hat{\Delta}^t > \tau$; **MaxJump**, $\max_t \hat{\Delta}^t$; and **MaxPrefixDev**, $\max_t \hat{R}^t$, with $\tau = 2.0$ nats. We also report **FirstFailurePos**, the earliest token where a verifiable instruction is irrecoverably violated on LongGenBench, or where repetition/format detectors trigger on LongWriter-style prompts.

Table 2 shows that PLRC substantially reduces SpikeRate and MaxJump and delays FirstFailurePos compared with MLE and long-context baselines. PLRC+LongPO performs best, indicating that spike control and preference alignment are complementary. Table 3 further shows that MaxJump and SpikeRate are the strongest predictors of failure, supporting the hypothesis that long-horizon breakdowns are governed by rare tail events rather than average drift.

Statistic	**Spearman ρ with Failure**	**AUC (Failure Prediction)**
LongGenBench (32K), Qwen2.5-7B-Instruct		
SpikeRate(τ)	0.53	0.79
MaxJump	0.58	0.82
MaxPrefixDev	0.46	0.75
FirstFailurePos	-0.61	0.84
LongWriter-style prompts, Qwen2.5-7B-Instruct		
SpikeRate(τ)	0.49	0.77
MaxJump	0.55	0.80
MaxPrefixDev	0.42	0.73
FirstFailurePos	-0.58	0.82

Table 3: Predictiveness of spike statistics for pass/fail outcomes. AUC uses each scalar statistic as a failure score (higher implies more likely failure), except FirstFailurePos (lower implies more likely failure).

Table 3 yields three observations. First, failures remain strongly associated with spike events even for strong baselines: MaxJump and SpikeRate are the most predictive statistics, consistent with strict long-horizon tasks where one violated constraint can fail the whole sample. Second, PLRC reduces spike frequency and magnitude and delays FirstFailurePos, supporting its intended mechanism of controlling tail increments rather than only average token likelihood. Third, PLRC+LongPO performs best on spike metrics and first-failure localization, suggesting complementarity between preference alignment and spike control.

Overall, these results explain why mean token loss can fail at long horizon: rare prefix-deviation spikes, rather than average drift alone, often govern failures. By regularizing spike tails, PLRC more directly targets this mechanism and improves long-horizon robustness.

6 Limitations

PLRC relies on a learned prefix ratio critic, whose estimation quality may affect the regularization signal in high-dimensional language spaces. It also adds rollout and critic-training overhead during finetuning. Our experiments focus on long-form generation, long-context instruction following, and long-document summarization, so further validation is needed for broader settings such as agents, tool use, and multilingual generation. Finally, PLRC reduces tail instability but does not eliminate failures caused by factual errors or task-specific reasoning limitations.

7 Conclusion

In this work, we revisited long-horizon failures from a process-level perspective and argued that improving mean token loss is not sufficient to ensure reliable long-form behavior. Two structural

effects drive this mismatch: (i) **length amplification**, where small per-token discrepancies compound into large sequence-level distortions, and (ii) **spike-dominated deviations**, where a handful of localized decisions abruptly push generation into off-distribution regions and trigger cascading breakdowns. To address both, we introduced **Prefix Likelihood-Ratio Control (PLRC)**, which learns a prefix ratio critic and augments standard teacher-forcing with tail-sensitive regularizers that directly constrain the dynamics of prefix deviation, controlling terminal drift, suppressing spike events, and limiting worst-case path excursions. Across long-form generation and long-context benchmarks, PLRC consistently improves robustness at increasing lengths while preserving short-horizon quality, and our diagnostics confirm that the gains are driven by reduced tail risk rather than changes in average likelihood.

Acknowledgments and Disclosure of Funding

The author received no third-party funding or support for this work during the 36 months prior to submission. The author declares no competing interests or relevant financial relationships during that period.

References

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Yushi Bai, Jiajie Zhang, Xin Lv, Linzhi Zheng, Siqi Zhu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. Longwriter: Unleashing 10,000+ word generation from long context llms. *arXiv preprint arXiv:2408.07055*, 2024.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

Daniele Calandriello, Daniel Guo, Remi Munos, Mark Rowland, Yunhao Tang, Bernardo Avila Pires, Pierre Harvey Richemond, Charline Le Lan, Michal Valko, Tianqi Liu, et al. Human alignment of large language models through online preference optimisation. *arXiv preprint arXiv:2403.08635*, 2024.

Guanzheng Chen, Xin Li, Michael Qizhe Shieh, and Lidong Bing. Longpo: Long context self-evolution of large language models through short-to-long preference optimization. *arXiv preprint arXiv:2502.13922*, 2025.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. A discourse-aware attention model for abstractive summarization of long documents. *arXiv preprint arXiv:1804.05685*, 2018.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, et al. Ultrafeedback: Boosting language models with scaled ai feedback. *arXiv preprint arXiv:2310.01377*, 2023.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407, 2024.

Rick Durrett. *Probability: theory and examples*, volume 49. Cambridge university press, 2019.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*, 2024.

Tianyu Gao, Alexander Wettig, Howard Yen, and Danqi Chen. How to train long-context language models (effectively). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7376–7399, 2025.

Falko Helm, Nico Daheim, and Iryna Gurevych. Token weighting for long-range language modeling. 2025.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

Jiwoo Hong, Noah Lee, and James Thorne. Orpo: Monolithic preference optimization without reference model. 2024.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. Efficient attentions for long document summarization. *arXiv preprint arXiv:2104.02112*, 2021.

Shaojie Jiang, Pengjie Ren, Christof Monz, and Maarten de Rijke. Improving neural response diversity with frequency-aware cross-entropy loss. In *The world wide web conference*, pages 2879–2885, 2019.

Shaojie Jiang, Thomas Wolf, Christof Monz, and Maarten de Rijke. Tldr: token loss dynamic reweighting for reducing repetitive utterance generation. *arXiv preprint arXiv:2003.11963*, 2020.

Sanyam Kapoor, Nate Gruver, Manley Roberts, Arka Pal, Samuel Dooley, Micah Goldblum, and Andrew Wilson. Calibration-tuning: Teaching large language models to know what they don't know. In *Proceedings of the 1st Workshop on Uncertainty-Aware NLP (UncertaiNLP 2024)*, pages 1–14, 2024.

Bhawesh Kumar, Charlie Lu, Gauri Gupta, Anil Palepu, David Bellamy, Ramesh Raskar, and Andrew Beam. Conformal prediction with large language models for multi-choice question answering. *arXiv preprint arXiv:2305.18404*, 2023.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. *Deep learning*, volume 521. Nature Publishing Group UK London, 2015.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, et al. Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*, 2023.

Huawei Lin, Jikai Long, Zhaozhuo Xu, and Weijie Zhao. Token-wise influential training data retrieval for large language models. 2024.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. volume 35, pages 27730–27744, 2022.

Richard Yuanzhe Pang and He He. Text generation by learning from demonstrations. *arXiv preprint arXiv:2009.07839*, 2020.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. volume 36, pages 53728–53741, 2023.

Jie Ren, Yao Zhao, Tu Vu, Peter J Liu, and Balaji Lakshminarayanan. Self-evaluation improves selective generation in large language models. In *Proceedings on*, pages 49–64. PMLR, 2023.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*, 2021.

Cosma Shalizi. Advanced data analysis from an elementary point of view. 2013.

Zhenpeng Su, Zijia Lin, Baixue Baixue, Hui Chen, Songlin Hu, Wei Zhou, Guiguang Ding, et al. Mile loss: a new loss for mitigating the bias of learning difficulties in generative language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 250–262, 2024.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Alpaca: A strong, replicable instruction-following model. Technical Report 6, 2023.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 13484–13508, 2023.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. 2021.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319*, 2019a.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319*, 2019b.

Yuhao Wu, Ming Shan Hee, Zhiqing Hu, and Roy Ka-Wei Lee. Longgenbench: Benchmarking long-form generation in long context llms. *arXiv preprint arXiv:2409.02076*, 2024.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy S Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. Doremi: Optimizing data mixtures speeds up language model pretraining. volume 36, pages 69798–69818, 2023.

Beining Xu and Yongming Lu. Tecp: Token-entropy conformal prediction for llms. *Mathematics*, 13 (20):3351, 2025.

An Yang, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoyan Huang, Jiandong Jiang, Jianhong Tu, Jianwei Zhang, Jingren Zhou, et al. Qwen2. 5-1m technical report. *arXiv preprint arXiv:2501.15383*, 2025.

Appendix

Contents

- A Implementation Details
- B Experiments across models
- C Ablation Study
- D Human & AI-Judge Pairwise Preference
- E Critic Bias Analysis: Robustness to Misspecification
- F Computational Cost of PLRC
- G Theoretical Guarantees
- H Case Studies: What Gets Marked as Spike

A Implementation Details

A key sensitivity of PLRC is how the ratio critic is trained: the rollout policy, the prefix-length distribution, and whether negatives are stabilized with replay all affect whether u_ϕ learns a meaningful density-ratio signal or merely artifacts of a particular decoding scheme. We therefore specify these choices explicitly.

Critic objective and supervision construction. We train the prefix ratio critic $u_\phi(c, x_{1:t})$ by *noise-contrastive prefix discrimination* between (i) model-generated prefixes and (ii) teacher-forced data prefixes under a balanced mixture. For each context c, we sample a prefix length t, take the dataset prefix $x_{1:t} \sim p_t(\cdot \mid c)$ as a negative, and generate a model prefix $\tilde{x}_{1:t} \sim q_t(\cdot \mid c) \equiv p_{\theta,t}(\cdot \mid c)$ as a positive. The critic is trained with the logistic objective in Eq 12 using $\widehat{R}_t(c, x_{1:t}) = \log p_{\theta,t}(x_{1:t} \mid c) + u_\phi(c, x_{1:t})$ as the logit.

Rollout decoding policy (positives). To reduce "decoding-bias" signals, we generate critic positives using *ancestral sampling* from the current model distribution (no beam search; no heuristic re-ranking). Concretely, we sample with a fixed temperature $\tau_{\text{roll}} = 1.0$ and *no truncation* (top-$p = 1.0$, top-$k = \infty$), so that $\tilde{x}_{1:t}$ matches the theoretical q_t in Sec. 4.3 as closely as possible. We keep this rollout policy fixed across all backbones and all training runs; greedy decoding is reserved for evaluation-only determinism (and an auxiliary robustness check), not for critic positives.

Prefix-length sampling $\pi(t)$. We sample t from a length grid that spans the full training horizon to ensure the critic sees both early and late prefixes. Specifically, let $\mathcal{T} = \{512, 1\text{K}, 2\text{K}, 4\text{K}, 8\text{K}, 16\text{K}, 32\text{K}\}$ truncated to the backbone context window; we sample t *uniformly over* $\mathcal{T}$ and then uniformly within the selected bucket (integer length within that range). For very long-context backbones (e.g., 128K), we additionally sample an *offset* s uniformly from $\{0, \ldots, T_{\max} - t\}$ and roll out a *windowed prefix* $(c, \tilde{x}_{s+1:s+t})$ so that the critic is trained on late-stage behavior without requiring prohibitively long rollouts each step.

Replay buffer for stable positives. We maintain a FIFO replay buffer $\mathcal{B}$ of recent model prefixes to reduce variance as p_θ changes. Each stored item contains $(c, \tilde{x}_{1:t}, t)$ (and offset metadata when windowed). At each critic step, we draw a mini-batch of positives as a mixture of *fresh* rollouts and *replayed* rollouts: 50% fresh and 50% replay by default. We refresh the buffer continuously by enqueueing all fresh rollouts each iteration and dropping the oldest samples when capacity is reached. In all experiments we use a small buffer (2–4K prefixes) since it is sufficient to stabilize discrimination while keeping memory overhead negligible.

Update schedule and learning rates. We alternate one critic update and one model update per iteration (1:1). The critic uses AdamW with learning rate $2\times$ the model learning rate, and we apply gradient clipping to both updates for stability. During the θ-update, we stop gradients through u_ϕ so that regularizers backpropagate only through $\log p_{\theta,t}$ (Sec. 4.4), preventing a moving-target minimax dynamic.

Practical stabilizers (used throughout). We use (i) a short critic warm-up (first 200–500 steps) where θ is updated normally but rollouts are generated with the same policy and the critic learns to separate prefixes before strong regularization takes effect; and (ii) logit clamping on $\widehat{R}_t$ to avoid numerical issues in the exponential-moment regularizers (implemented as a smooth clamp, e.g., $\widehat{R}_t \leftarrow \mathrm{clip}(\widehat{R}_t, [-C, C])$ with $C = 20$). These stabilizers do not change the formal objective; they only prevent pathological critic saturation early in training.

B Experiments across models

Table 4 summarizes representative results across backbones by selecting the most diagnostic benchmark/metric for each model (LongGenBench for long-form robustness, LongWriter for long-form writing stability, and PubMed ROUGE-L for long-document summarization). PLRC yields consistent gains over MLE and strong long-context training baselines (Token Weighting and ProLong) across all model families. The improvements are most pronounced on LongGenBench and LongWriter, aligning with our central hypothesis that long-horizon failures are driven by length-amplified sequence deviation and spike-like derailments; controlling the prefix deviation process therefore provides benefits that persist across architectures and context-window regimes. For summarization (PubMed), PLRC provides smaller but steady improvements, suggesting that suppressing prefix drift also helps long-document fidelity without task-specific architectural changes.

Backbone	**Benchmark / Metric**	**MLE**	**TokenW**	**ProLong**	**PLRC**
Qwen2.5-14B-Instruct	LongGenBench / Pass@1 ↑	17.54	18.11	18.86	**21.03**
Qwen2.5-14B-Instruct	PubMed / ROUGE-L ↑	42.02	42.30	42.65	**43.27**
Llama 3.1 8B Instruct	LongWriter / Score ↑	73.35	73.96	74.48	**77.14**
Phi-3-Medium-128K-Instruct	LongGenBench / Pass@1 ↑	18.28	18.89	19.35	**22.03**

Table 4: Per-backbone results on representative metrics.

C Ablation Study

Our method is motivated by two distinct but coupled failure mechanisms in long-horizon generation: (i) *length amplification*, where small per-token gaps can accumulate into large sequence-level distribution shifts, and (ii) *spike-dominated derailment*, where rare but sharp prefix deviations trigger cascading failures. PLRC targets these via two regularizers: a terminal (length) tail control term on the prefix likelihood-ratio process, and an increment (spike) tail control term on its jumps. This ablation isolates the contribution of each component and tests whether they are complementary in improving long-horizon robustness.

We evaluate the following variants under the same backbones, data splits, decoding settings, and training budget as the main experiments. **MLE** is standard teacher-forcing training without PLRC. **PLRC (len-only)** enables only terminal ratio tail control. **PLRC (spike-only)** enables only increment spike control. **PLRC (full)** enables both terms (and the same optional path control setting as the main method). Unless stated otherwise, hyperparameters follow the default setting used throughout: $\alpha = \beta = 0.1$ (unused weights set to zero for ablations).

We report end-to-end performance on LongWriter, GovReport, PubMed, and LongGenBench, using the same metrics as Table 1. LongGenBench is used strictly for evaluation and never used for training. We additionally report spike diagnostics on LongGenBench to verify the intended mechanism (reduction of derailment jumps and delayed first failures).

C.1 End-to-end ablation results

Table 5 shows that both components contribute, but in different ways. Terminal tail control (len-only) yields consistent improvements on summarization (GovReport/PubMed) and modest gains on long-form robustness, consistent with mitigating length-amplified sequence deviation. Spike control (spike-only) yields larger gains on LongGenBench and LongWriter-style long generation, consistent with suppressing rare derailment jumps that dominate long-horizon failures. The full PLRC achieves the best overall performance, indicating that length and spike controls are complementary rather than redundant.

Variant	LongWriter / Score ↑	GovReport / ROUGE-L ↑	PubMed / ROUGE-L ↑	LongGenBench / Pass@1 ↑
MLE (no PLRC)	74.09	54.67	41.26	15.02
PLRC (len-only)	76.14	56.12	42.15	16.63
PLRC (spike-only)	77.46	55.40	41.87	17.74
PLRC (full)	**79.05**	**56.81**	**42.68**	**18.55**

Table 5: Ablation results on **Qwen2.5-7B-Instruct**. "len-only" enables terminal ratio tail control; "spike-only" enables increment spike control; "full" enables both (and optional path control).

To confirm that improvements are not driven by a single ROUGE variant, Table 6 reports ROUGE-2 on GovReport and PubMed. The same pattern holds: len-only primarily improves summarization consistency over long inputs, while spike-only yields smaller ROUGE gains but stronger long-horizon generation robustness (Table 5).

Variant	GovReport / ROUGE-2 ↑	PubMed / ROUGE-2 ↑
MLE (no PLRC)	22.63	18.22
PLRC (len-only)	23.71	18.84
PLRC (spike-only)	23.13	18.64
PLRC (full)	**24.18**	**19.30**

Table 6: ROUGE-2 ablation results on **Qwen2.5-7B-Instruct**.

C.2 Mechanistic ablation: spikes, max deviation, and first failure

Table 7 validates the intended mechanism. Spike-only sharply reduces SpikeRate and MaxJump and substantially delays FirstFailurePos, while len-only provides weaker spike suppression but still reduces MaxPrefixDev, consistent with controlling length-amplified terminal deviation. The full PLRC combines both: it matches spike-only in spike suppression while also achieving lower MaxPrefixDev, explaining why it yields the best Pass@1 and overall robustness.

Variant	SpikeRate(τ)↓	MaxJump↓	MaxPrefixDev↓	FirstFailurePos (tokens)↑
MLE (no PLRC)	0.62%	8.91	41.72	6,520
PLRC (len-only)	0.56%	8.14	36.84	7,880
PLRC (spike-only)	0.43%	6.65	34.10	10,420
PLRC (full)	**0.41%**	**6.59**	**32.43**	**10,980**

Table 7: Mechanistic ablation on **LongGenBench (32K budget)** with **Qwen2.5-7B-Instruct**. SpikeRate uses $\tau = 2.0$ (nats). FirstFailurePos is defined by the benchmark verifier as in §5.4.

Analysis. The ablation supports three conclusions. First, the two PLRC components address distinct aspects of the long-horizon failure mechanism: terminal tail control improves long-range consistency and summarization quality (higher ROUGE on GovReport/PubMed), while spike control targets catastrophic derailments that dominate strict long-form success metrics (higher Pass@1 on LongGenBench and higher LongWriter score). Second, spike control is the primary driver of improvements on strict long-horizon pass criteria, consistent with our claim that failures are governed by rare jump events rather than mean token loss. Third, combining the two yields the best overall outcomes, indicating complementarity: spike suppression prevents early derailment, while terminal/path deviation control reduces length-amplified drift that accumulates even without large local jumps.

These results strengthen the causal interpretation of PLRC: the gains are not an artifact of generic reweighting, but arise from explicitly controlling (i) length-amplified sequence deviation and (ii) spike-like derailment jumps, which together determine long-horizon robustness.

D Human & AI-Judge Pairwise Preference

Protocol. We randomly sample 100 prompts from the evaluation set and generate two anonymized outputs per prompt (**PLRC** vs **Baseline**) using the same decoding configuration. We run (i) **human** pairwise preference with three axes (*coherence*, *instruction adherence*, *factuality*), and (ii) a **GPT5-based judge** with the same rubric. For each axis, the rater selects `PLRC win` / `tie` / `Baseline win`. Table 8 reports counts out of 100.

	Human (100)			GPT Judge (100)		
Axis	**Win**	**Tie**	**Lose**	**Win**	**Tie**	**Lose**
Overall preference	53	24	23	70	12	18
Coherence	68	12	20	68	19	13
Instruction adherence	59	20	21	69	13	18
Factuality	56	18	26	62	16	22

Table 8: **Pairwise preference on 100 prompts** (**PLRC** vs **Baseline**). Counts are #prompts where PLRC is preferred (**Win**), judged indistinguishable (**Tie**), or baseline is preferred (**Lose**).

E Critic Bias Analysis: Robustness to Misspecification

PLRC relies on a prefix ratio critic u_ϕ to approximate the missing $-\log p_t^*(x_{1:t} \mid c)$ term in the prefix log-likelihood ratio estimator $\widehat{R}_t$. A natural concern is that this dependency could constitute a strong assumption: if the critic is biased or uninformative, PLRC might over-regularize and degrade generation quality rather than improving long-horizon robustness. Here we explicitly stress-test this failure mode and ask whether PLRC *degrades gracefully* when the critic becomes unreliable.

We keep the LM training recipe fixed and replace only the critic with progressively more misspecified variants: (i) a **Learned critic** trained normally for prefix discrimination; (ii) a **Noisy critic** trained with 30% label flips (systematic corruption of the supervision signal); and (iii) a **Random critic** that outputs i.i.d. logits independent of the prefix (AUC ≈ 0.50). We then evaluate both token-level fit (**PPL**) and long-horizon robustness on 32K generation (**Coverage** and **CondPass**; see Sec. 5.3 for metric definitions).

Table 9 shows three consistent trends. **(i) Gains track critic quality.** With a learned critic, PLRC improves long-horizon performance substantially: 32K Coverage increases from 19.8 to 26.1 and 32K CondPass increases from 35.7 to 46.5, while PPL slightly improves ($7.92 \rightarrow 7.88$). As we inject bias (30% label flip), the improvements shrink smoothly but remain positive (Coverage 22.4, CondPass 40.8), indicating that PLRC tolerates moderate critic corruption.

(ii) When the critic is uninformative, PLRC becomes a near-no-op rather than collapsing. With a random critic (AUC ≈ 0.50), long-horizon metrics revert to the MLE baseline: Coverage is 19.7 vs. 19.8 for MLE, and CondPass is 35.9 vs. 35.7. PPL also stays close (7.95 vs. 7.92). This pattern supports an important safety property: an uninformative critic does *not* trigger catastrophic over-regularization; instead, the PLRC regularizers contribute little useful signal and the training behavior largely reduces to standard MLE.

(iii) Short-horizon fit is preserved across all variants. Across learned/noisy/random critics, PPL changes are minimal, consistent with PLRC being an *augmentation* on top of teacher forcing: the MLE term anchors token-level fit even when the critic quality deteriorates.

This analysis addresses the concern that PLRC critically depends on a perfectly accurate critic. In practice, performance gains scale with critic quality, but when the critic fails (AUC ≈ 0.50), PLRC degrades gracefully and reverts to near-baseline behavior rather than exhibiting severe collapse. This empirical "graceful degradation" is essential for deploying PLRC in high-dimensional language settings where ratio estimation is inevitably imperfect.

Training Variant	**PPL** $\downarrow$	**32K Coverage (%)** $\uparrow$	**32K CondPass (%)** $\uparrow$
MLE (no PLRC)	7.92	19.81	35.76
PLRC + **Learned critic**	**7.88**	**26.13**	**46.55**
PLRC + **Noisy critic** (30% label flip)	7.90	22.46	40.83
PLRC + **Random critic** (AUC$\approx$0.50)	7.95	19.75	35.93

Table 9: Critic bias analysis on short and long context.

F Computational Cost of PLRC

From token-forward equivalents to practical overhead. We can get PLRC overhead in *token-forward equivalents* (TFE): one Transformer forward pass on one token is 1 TFE, and a forward+backward step is approximated as $\approx$ 3 TFE per token. The dominant additional cost in PLRC comes from forward-only model rollouts used to train the ratio critic, yielding the relative overhead

$$\text{Overhead} \approx \frac{\text{Cost}_{\text{rollout}}}{\text{Cost}_{\text{MLE}}} = \frac{BM\bar{t}}{3BT} = \frac{M\bar{t}}{3T}, \tag{19}$$

We denote by B the number of sequences per optimization step, by T the teacher-forcing length used for the MLE update, by t the sampled prefix length used to train the ratio critic (with mean $\bar{t} = \mathbb{E}[t]$), and by M the number of model rollouts generated per step for critic/regularizer computation. The overhead is independent of batch size B and grows linearly with the average rollout length $\bar{t}$ and rollout multiplicity M. Rollouts are generated under `no_grad`, so they do not store activations for backprop; memory overhead is primarily transient KV-cache during sampling plus storage of sampled token ids.

Measurement protocol. To aid practical adoption, we additionally report concrete wall-clock latency, throughput, and peak memory on NVIDIA A100-80GB and H100-80GB (8 GPUs), using BF16, FSDP/ZeRO-style sharding, FlashAttention, and gradient checkpointing. Each entry reports: (i) **step time** (seconds/optimizer step), (ii) **throughput** (global tokens/s = total tokens processed across all GPUs divided by step time), (iii) **peak memory** (max allocated GB per GPU), and (iv) a breakdown of **critic overhead** beyond rollout generation.

The TFE analysis counts rollout generation but treats critic compute as negligible. In wall-clock, we explicitly separate: *(a)* rollout generation time (LM forward-only decoding), and *(b)* critic forward/backward + optimizer steps (a small MLP/Transformer head). The reported "critic-only" overhead measures (b) relative to the MLE baseline step.

Backbone	GPU	Variant	Step (s)↓	Throughput (k tok/s)↑	Peak Mem (GB)↓	Total Ovhd	Rollout Ovhd	Critic Ovhd
Qwen2.5-7B	A100-80G	MLE	1.100	119.2	52.0	–	–	–
		PLRC (Light)	1.161	113.1	53.0	5.5%	4.2%	1.3%
		PLRC (Default)	1.203	109.0	54.2	9.4%	8.6%	0.8%
	H100-80G	MLE	0.720	182.0	49.0	–	–	–
		PLRC (Light)	0.750	174.7	50.0	4.2%	3.6%	0.6%
		PLRC (Default)	0.780	168.1	51.0	8.3%	7.6%	0.7%
Llama-3.1-8B	A100-80G	MLE	1.260	104.0	56.0	–	–	–
		PLRC (Light)	1.329	98.6	57.2	5.5%	4.1%	1.4%
		PLRC (Default)	1.382	94.8	58.5	9.7%	8.5%	1.2%
	H100-80G	MLE	0.820	160.0	53.0	–	–	–
		PLRC (Light)	0.857	153.1	54.0	4.5%	3.2%	1.3%
		PLRC (Default)	0.891	147.3	55.0	8.6%	7.7%	0.9%

Table 10: **Wall-clock and memory cost of PLRC (8×GPU).** We report optimizer step time, global throughput, and peak per-GPU memory on A100-80GB and H100-80GB. **Total Ovhd** is measured vs. MLE at the same (B, T); **Rollout Ovhd** corresponds to forward-only rollout generation; **Critic Ovhd** is the additional cost of critic updates beyond rollouts. LIGHT (B=32, T=4096, $\bar{t}$=512, M=1), DEFAULT (B=32, T=4096, $\bar{t}$=1024, M=1).

Across both backbones, LIGHT adds $\approx 4\%$ and DEFAULT adds $\approx 8\%$ compute, while critic-only overhead remains $\leq 1.4\%$ in typical settings. Peak memory increases modestly ($\approx$ 1–2 GB/GPU), consistent with rollouts not storing backward activations and incurring mainly transient KV-cache.

G Theoretical Guarantees

This appendix summarizes why the PLRC regularizers in Eq 18 provide direct control over long-horizon behavior that mean token loss does not. The key object is the prefix log-likelihood ratio process $R_t(c, x_{1:t}) = \log \frac{p_{\theta,t}(x_{1:t}|c)}{p_t^*(x_{1:t}|c)}$ and its increments $\Delta_t = R_t - R_{t-1}$ (Eq 6 in the main text). Since p_t^* is unknown, PLRC uses the estimated ratio $\widehat{R}_t(c, x_{1:t}) = \log p_{\theta,t}(x_{1:t} \mid c) + u_\phi(c, x_{1:t})$, where the critic u_ϕ is trained by prefix discrimination and is treated as stop-gradient during θ updates.

Prefix change-of-measure identity. For any prefix event $E_t \subseteq \mathcal{X}^t$, the model probability can be written as a data expectation reweighted by the likelihood ratio:

$$p_{\theta,t}(E_t \mid c) = \mathbb{E}_{x_{1:t}\sim p_t^*(\cdot|c)}\big[\exp(R_t(c, x_{1:t}))\,\mathbf{1}\{x_{1:t} \in E_t\}\big]. \tag{20}$$

This identity makes explicit that long-horizon failures are governed by the *tail behavior* of $\exp(R_t)$ on rare sets E_t rather than by mean token loss.

Guarantee 1: event distortion is controlled by exponential tails of R_t (length amplification). A standard Hölder/Markov argument implies that for any $\lambda > 1$,

$$p_{\theta,t}(E_t \mid c) \;\lesssim\; \Big(\mathbb{E}_{p_t^*}[\exp(\lambda R_t)]\Big)^{1/\lambda} \cdot \big(p_t^*(E_t \mid c)\big)^{1-1/\lambda}. \tag{21}$$

Thus, to keep the model from assigning disproportionately large probability to rare "failure" events (format breaks, constraint violations, contradictions), it is sufficient to control the exponential moments of R_t (or R_T) under the data. PLRC implements this principle using tail-sensitive penalties on the estimated ratio process:

$$\begin{aligned} R_{\text{len}}(\theta;\phi) &= \log \mathbb{E}_{(c,x)\sim\mathcal{D}}\Big[\exp(\widehat{R}_T(c, x_{1:T}))\Big], \\ R_{\text{path}}(\theta;\phi) &= \log \mathbb{E}_{(c,x)\sim\mathcal{D}}\Big[\sum_{t=1}^{T}\exp(\widehat{R}_t(c, x_{1:t}))\Big]. \end{aligned} \tag{22}$$

which are exactly the two terms grouped under the single weight α in Eq 18. Intuitively, R_{len} targets *terminal* length-amplified drift, while R_{path} prevents *any-time* excursions into highly deviated prefixes.

Guarantee 2: first-spike (derailment jump) probability is controlled by exponential tails of Δ_t. Let a "spike" be a large positive increment $\Delta_t \geq \tau$. Define the first-spike event $\{\exists t \leq T : \Delta_t \geq \tau\}$. Using (i) the change-of-measure identity, (ii) Markov's inequality, and (iii) a union bound over time, one obtains a bound of the form

$$\mathbb{P}_{p_\theta}\Big(\max_{t\leq T}\Delta_t \geq \tau \,\Big|\, c\Big) \;\lesssim\; \exp(-\tau)\sum_{t=1}^{T}\mathbb{E}_{x_{1:t}\sim p_t^*(\cdot|c)}\Big[\exp([\Delta_t]_+)\Big]. \tag{23}$$

PLRC directly penalizes the corresponding empirical proxy built from the estimated increments $\widehat{\Delta}_t = \widehat{R}_t - \widehat{R}_{t-1}$:

$$R_{\text{spike}}(\theta;\phi) = \log \sum_{t=1}^{T}\mathbb{E}_{(c,x)\sim\mathcal{D}}\Big[\exp([\widehat{\Delta}_t(c, x_{1:t})]_+)\Big], \tag{24}$$

which is the sole term weighted by β in Eq. (18). Therefore, decreasing R_{spike} tightens an exponential upper bound on the probability of an early derailment jump, matching the "spike-dominated failure" mechanism.

Summary (what Eq 18 controls). With only two hyperparameters, PLRC enforces two complementary controls: *(i) accumulated deviation control* via $\alpha\,[R_{\text{len}}+R_{\text{path}}]$, which suppresses length-amplified sequence-level drift and any-time prefix excursions; and *(ii) spike control* via $\beta\,R_{\text{spike}}$, which suppresses rare but catastrophic ratio jumps that trigger long-horizon collapse. These controls act on the trajectory-level deviation process itself, rather than on averaged token losses, and thus better match the failure modes observed in long-horizon generation.

H Case Studies: What Gets Marked as Spike

To make the spike notion concrete, we include case studies. In each case, we show a short excerpt around the first flagged spike position. We mark spike-triggering text spans with **[SPIKE]** ... **[/SPIKE]** and give a brief hallucination tag. These cases illustrate that spikes often correspond to abrupt jumps into low-data-likelihood regions (e.g., fabricated citations, invented entities/APIs, or sudden numerical claims).

Case 1 (Fabricated restaurant / address).

Prompt (excerpt). "Recommend a quiet cafe near King's Cross and include the address."
Model output (excerpt). A nice quiet option is to walk toward Granary Square and find a spot with seating by the canal. **[SPIKE]**I recommend *Willow & Hearth Cafe*, 17 Crandon Street, London N1C 4AX, open until 10pm daily.**[/SPIKE]** It has plenty of outlets and usually low noise in the afternoons.

Case 2 (Invented historical fact).

Prompt (excerpt). "Why is the Eiffel Tower important? Keep it factual."
Model output (excerpt). The Eiffel Tower became a symbol of modern engineering and was originally built for the 1889 World's Fair. **[SPIKE]**It was briefly dismantled in 1902 and reconstructed with a stronger steel frame in 1904.**[/SPIKE]** Its radio antenna later helped early broadcast experiments in France.

Case 3 (Non-existent medical guideline).

Prompt (excerpt). "Can I take ibuprofen with coffee?"
Model output (excerpt). In many people, coffee doesn't directly interact with ibuprofen, but it can irritate the stomach. **[SPIKE]**The WHO recommends waiting at least 90 minutes after caffeine before taking any NSAID.**[/SPIKE]** If you have reflux or ulcers, consider food and water with the dose.

Case 4 (Fabricated programming flag / option).

Prompt (excerpt). "How do I list all tables in a PostgreSQL database?"
Model output (excerpt). In `psql`, you can use meta-commands like `\dt`. **[SPIKE]**Alternatively, run `psql -list-tables -all-schemas` for a full table listing.**[/SPIKE]** You can also query `information_schema.tables`.

Case 5 (Invented quote attribution).

Prompt (excerpt). "Give me a short quote about perseverance from a famous author."
Model output (excerpt). A common theme is continuing despite setbacks. **[SPIKE]**As Jane Austen wrote, "Perseverance is the quiet engine of every triumph."**[/SPIKE]** You could also paraphrase the idea if you want something more modern.

Case 6 (Invented product spec).

Prompt (excerpt). "Does the iPhone 13 support a 120Hz display?"
Model output (excerpt). Some models in the iPhone lineup have high refresh rate displays. **[SPIKE]**The standard iPhone 13 supports 120Hz ProMotion starting from iOS 16.2.**[/SPIKE]** If you want consistently smoother scrolling, enable the high refresh rate in Display settings.

Case 7 (Fabricated policy / airline rule).

Prompt (excerpt). "Can I bring a power bank on a flight?"
Model output (excerpt). Rules vary slightly by airline, but power banks are typically restricted to carry-on baggage. **[SPIKE]**Most airlines allow up to 5 power banks per person as long as each is under 50,000 mAh.**[/SPIKE]** If your battery is labeled in Wh, check the airline's thresholds.

Case 8 (Sudden confident but wrong math claim).

Prompt (excerpt). "What's the monthly payment on a $300,000 mortgage at 6% for 30 years?"
Model output (excerpt). Mortgage payments depend on interest rate and term length. **[SPIKE]**At 6% for 30 years, the principal-and-interest payment is about $1,050 per month.**[/SPIKE]** Property taxes and insurance would add additional monthly costs.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims in the abstract and introduction are aligned with the paper's contributions, assumptions, and empirical/theoretical evidence, without overstating generalization beyond the evaluated settings.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses the main limitations.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The required assumptions are stated for each theoretical result.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper reports sufficient experimental details to reproduce the main results and conclusions.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: The code and data will be released upon publication; the paper provides sufficient implementation and experimental details for reproducibility.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper reports the necessary training, testing, and implementation details to understand the experimental results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The main experimental results include appropriately defined error bars or significance information.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper summarizes the main hardware configuration, runtime, and total compute required for reproducing the reported experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: The work is foundational methodological research and is not tied to a specific deployment scenario or direct societal application. We do not identify a direct path to societal harm beyond generic uses of machine learning methods.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release data or models with high misuse risk.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper properly credits existing assets and reports their licenses and terms of use.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not used as an important, original, or non-standard component of the proposed method. An LLM-based judge is used only for auxiliary evaluation and is described in Appendix D.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.