Missing Old Logits in Asynchronous Agentic RL: Semantic Mismatch and Repair Methods for Off-Policy Correction

Zhong Guan[1,2]*, **Yongjian Guo**[3]*, **Haoran Sun**[4]*, **Wen Huang**[3]*,
Shuai Di[5], **Likang Wu**[1,2], **Xiong Jun Wu**[5]†, **Hongke Zhao**[1,2]‡
[1]Laboratory of Computation and Analytics of Complex Management Systems, Tianjin University
[2]Institute for Embodied Intelligence, Tianjin University
[3]Tsinghua University, [4]Peking University, [5]JDT AI Infra

Abstract

Asynchronous reinforcement learning improves rollout throughput for large language model agents by decoupling sample generation from policy optimization, but it also introduces a critical failure mode for PPO-style off-policy correction. In heterogeneous training systems, the total importance ratio should ideally be decomposed into two semantically distinct factors: a *training–inference discrepancy term* that aligns inference-side and training-side distributions at the same behavior-policy version, and a *policy-staleness term* that constrains the update from the historical policy to the current policy. We show that practical asynchronous pipelines with delayed updates and partial rollouts often lose the required historical training-side logits, or old logits. This missing-old-logit problem entangles discrepancy repair with staleness correction, breaks the intended semantics of decoupled correction, and makes clipping and masking thresholds interact undesirably. To address this issue, we study both exact and approximate correction routes. We propose three exact old-logit acquisition strategies: snapshot-based version tracking, a dedicated old-logit model, and synchronization via partial rollout interruption, and compare their system trade-offs. From the perspective of approximate correction, we focus on preserving the benefits of decoupled cor rection through a more appropriate approximate policy when exact old logits cannot be recovered at low cost, without incurring extra system overhead. Following this analysis, we adopt a revised PPO-EWMA method, which achieves significant gains in both training speed and optimization performance.

1 Introduction

Large-scale reinforcement learning for large language models (LLMs) increasingly relies on distributed rollout and training pipelines. Proximal Policy Optimization (PPO) [17] and its variants [31, 15, 1, 2], including GRPO [18], remain widely used because they provide a simple and stable mechanism for policy improvement: trajectories are generated by a behavior policy, and the current policy is optimized with an importance ratio and a clipped surrogate objective. In ideal on-policy or near-synchronous settings, this ratio has a clear interpretation. It compares the current policy against the policy that generated the sampled tokens, while clipping trades off update magnitude and optimization stability.

*Equal contributions.
†Corresponding author. (xiongjunwu.1@jd.com)
‡Corresponding author. (hongke@tju.edu.cn).
Code: https://github.com/millioniron/fixed_async

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

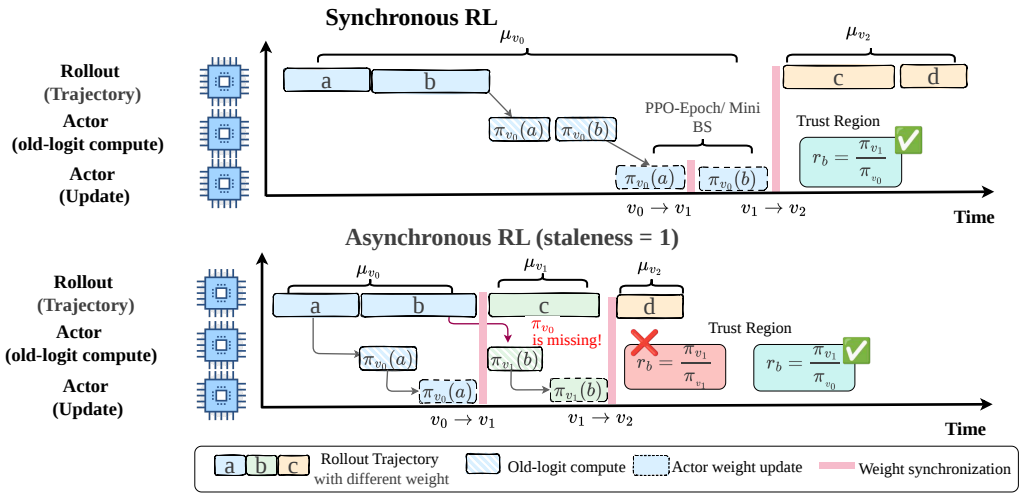


Figure 1: Synchronous versus asynchronous RL. In synchronous RL, the old logits used during training correspond to the same policy version that generated the rollout. In asynchronous RL, delayed updates and partial rollouts can make this version unavailable, which creates an old-logit mismatch and breaks the intended separation between training–inference discrepancy repair and policy-staleness correction.

This interpretation becomes fragile in modern Agentic RL systems. To maximize throughput, rollout and training are often physically separated. Rollouts are produced by optimized inference engines such as vLLM [11] or SGLang [38], whereas gradient updates are performed by training engines such as Megatron-LM or FSDP. Even when the inference and training sides nominally use the same model version, numerical kernels, precision scaling, quantization, tensor parallelism, and routing implementations can lead to different token probabilities. We call this effect *training–inference discrepancy* [29]. At the same time, asynchronous rollouts, large rollout queues, partial trajectories, and multiple actor updates make the behavior policy stale with respect to the current policy. We call this effect *policy staleness*.

A natural choice for correction is to decompose the total ratio into two terms: a discrepancy-repair ratio that compares the training-side and inference-side distributions at the same old version, and a staleness-correction ratio that compares the current training policy with that old training-side policy [28, 23, 32, 25, 24]. Let μ_{old} denote the inference-side rollout policy, and let π_{old} denote the corresponding training-side forward policy. The desired decomposition is

$$r(\theta) = \frac{\pi_\theta(y|x)}{\mu_{\text{old}}(y|x)} = r_d r_s, \quad r_d = \frac{\pi_{\text{old}}(y|x)}{\mu_{\text{old}}(y|x)}, \qquad r_s = \frac{\pi_\theta(y|x)}{\pi_{\text{old}}(y|x)}. \tag{1}$$

Here r_d measures training–inference discrepancy, while r_s measures policy staleness. This decomposition is attractive because the two terms have different meanings and should be controlled differently. Discrepancy repair should filter or down-weight numerically inconsistent tokens. Staleness correction should constrain policy updates with the sign-dependent PPO clipping rule.

However, asynchronous Agentic RL [5, 27, 33] introduces a practical obstacle: the old training-side policy values $\pi_{\text{old}}(y|x)$ may no longer be available when the trajectory reaches the actor. This is especially common under partial rollout collection, where one trajectory can span multiple parameter versions, and the actor may already have advanced beyond the version that generated earlier tokens. Once these old logits are missing, the decomposition in Eq. (1) is no longer semantically valid. Existing decoupled objectives may then mix discrepancy repair and staleness correction into a proxy ratio, causing the clipping and masking mechanisms to interfere with each other. However, current training stacks, including Verl [21], ROLL [26], and SLIME [39], still leave the old-logit mismatch unresolved.

This paper studies the missing-old-logit problem in asynchronous LLM RL. We first give a unified view of existing objectives as imposing two distinct constraints: a discrepancy constraint and a staleness constraint. This view shows why using one ratio or one threshold for both effects is

insufficient. We then analyze interpolation-based proxy policies and show that, under common constructions, they mainly re-parameterize effective clipping boundaries rather than recovering the missing reference policy. Finally, we examine two practical directions: exact acquisition of old logits through system support, and low-cost approximation through an exponentially-weighted moving average PPO (PPO-EWMA) reference policy [9].

Our contributions are summarized as follows.

- **We identify the missing-old-logit problem in asynchronous Agentic RL.** Missing training-side old logits break the intended separation between training–inference discrepancy repair and policy-staleness correction, creating a semantic failure mode in decoupled correction objectives.
- **We provide a unified analysis and practical correction strategies.** We formulate existing PPO-style objectives under a dual-constraint view, clarify the need to decouple discrepancy repair from staleness correction, and show that interpolation-based proxies mainly re-parameterize clipping boundaries. We further study three exact old-logit acquisition routes and a revised PPO-EWMA reference as a low-cost approximation.
- **We evaluate the performance–cost trade-off on dense and MoE LLMs.** Experiments on Agentic benchmarks compare exact recovery, proxy references, and PPO-EWMA across optimization behavior and system overhead.

2 Related Work

2.1 Off-Policy and Asynchronous Reinforcement Learning for LLMs

PPO [17] and GRPO [18] are widely used in LLM reinforcement learning because their clipped objectives stabilize policy updates while remaining straightforward to implement at scale [31]. However, on-policy training can be inefficient for long-horizon Agentic tasks, where rollout generation is expensive and GPU utilization is often limited by synchronization [7]. This has motivated off-policy and asynchronous RL pipelines that reuse stale trajectories and decouple rollout generation from policy optimization.

Several recent methods [4, 22] improve off-policy robustness by modifying the importance-sampling weights. CISPO [4] clips or regularizes importance weights for long-sequence training. GPPO [22] separates gradient propagation from clipping constraints to preserve useful exploratory gradients. M2PO [37] controls the second moment of importance weights to reduce variance under stale data. VESPO [19] and VCPO [10] use effective sample size as a stability signal, while MiniRL [35] and TOPR [16] modify trajectory-level importance weighting through tapered or asymmetric weighting. System-oriented work such as AReaL [6], HybridFlow [21], and related asynchronous frameworks study how to overlap rollout and training clusters at large scale.

These works demonstrate that off-policy and asynchronous training can substantially improve throughput. Our work focuses on a complementary issue: in heterogeneous asynchronous LLM RL, the policy version needed for a clean correction may be missing. This makes the meaning of the importance ratio ambiguous even before variance control or clipping design is considered.

2.2 Training-Inference Mismatch and Reference Policy Correction

Training–inference mismatch arises when the inference engine that produces rollouts and the training engine that computes gradients implement slightly different numerical computations. The mismatch is especially visible in MoE models, where routing decisions can amplify small numerical differences [36]. Existing approaches mitigate this instability through masking, clipping, or routing replay. Masked Importance Sampling (MIS) [13] masks tokens with severe training–inference divergence. IcePop [34] combines bilateral clipping and token masking to reduce the effect of unstable low-probability tokens. Routing-replay methods such as R2 [35] and R3 [14] align expert routing between rollout and training, thereby reducing MoE-specific discrepancy.

A separate line of work builds reference or proximal policies to stabilize stale updates. Decoupled PPO [35] separates importance correction from proximal constraints, while A-3PO [12] approximates the proximal policy via log-space interpolation to reduce overhead. PPO-EWMA-style references

maintain a smoothed policy anchor. These methods motivate our decoupled view. Our key distinction is that we examine whether the reference policy is semantically correct in asynchronous systems. When exact old training-side logits are absent, a proxy reference can help, but remains an approximation rather than true recovery of Eq. (1).

3 Preliminaries

3.1 PPO-Style Off-Policy Correction

We consider RL fine-tuning of an LLM on prompts $x \sim P$. Given a prompt x, a response $y = (y_1, \ldots, y_T)$ is sampled from an old policy π_{old}. In this standard PPO notation, π_{old} denotes the behavior policy, and we do not yet distinguish the inference-side rollout distribution from the training-side forward distribution. A reward model or environment returns a scalar reward $R(x, y)$, and an advantage estimate A_t is computed for each token or sequence.

For a token-level ratio $r_t(\theta) = \pi_\theta(y_t|x, y_{<t})/\pi_{\text{old}}(y_t|x, y_{<t})$, the PPO clipped surrogate is

$$\ell_{\text{PPO}}(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta) A_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon) A_t \right) \right]. \tag{2}$$

Equivalently, PPO clipping induces an advantage-sign-dependent active region. For $A_t > 0$, ratios above $1 + \epsilon$ are clipped; for $A_t < 0$, ratios below $1 - \epsilon$ are clipped. Therefore, at the per-token level, we rewrite the gradient contribution of Eq. (2) into a masked importance sampling (MIS) form: $\mathbb{E}_t[\text{MIS} \cdot r_t(\theta) A_t \nabla \log \pi_\theta(y_t|x, y_{<t})]$, where the PPO-side active mask is defined as

$$\text{MIS} = \mathbb{I}\{A_t \geq 0\}\mathbb{I}\{r_t(\theta) \leq 1+\epsilon\} + \mathbb{I}\{A_t < 0\}\mathbb{I}\{r_t(\theta) \geq 1-\epsilon\}, \tag{3}$$

where $\mathbb{I}\{\cdot\}$ denotes the indicator function. This advantage-sign-dependent mask is mainly used to enforce the policy-update constraint [17], preventing the policy update from becoming too large.

3.2 Training-Inference Discrepancy, Policy Staleness, and Missing Old Logits

Under modern asynchronous LLM RL systems [6, 20, 21, 26], the same parameter version can induce two distributions: the rollout distribution deployed on the inference engine, such as vLLM or SGLang, and the forward distribution deployed on the training side, such as Megatron or FSDP. Throughout this paper, we use μ to denote the inference-side policy and π to denote the training-side policy. The subscript v denotes the policy version, such as μ_v and π_v. By default, we use θ to denote the current version being optimized on the actor engine, and old to denote the rollout policy version used to generate the token on the inference engine. Therefore, the importance ratio can be naturally decomposed into a staleness ratio r_s and a discrepancy ratio between actor and rollout, r_d, i.e., $r_t(\theta) = r_s \times r_d$, where $r_s = \frac{\pi_\theta(y_t|x,y_{<t})}{\pi_{\text{old}}(y_t|x,y_{<t})}$ and $r_d = \frac{\pi_{\text{old}}(y_t|x,y_{<t})}{\mu_{\text{old}}(y_t|x,y_{<t})}$.

Some works consider controlling these two terms separately, for example by masking the discrepancy ratio to mitigate the impact of numerical discrepancies [30, 14]. More recently, IcePop [34] proposes using a strict masking threshold c for the discrepancy ratio, which can be formulated as an MIS objective:

$$\text{MIS} = M_{[1/c,c]}(r_d) \left(\mathbb{I}\{A_t \geq 0\}\mathbb{I}\{r_s \leq 1+\epsilon\} + \mathbb{I}\{A_t < 0\}\mathbb{I}\{r_s \geq 1-\epsilon\} \right), \tag{4}$$

where we define the masking function $M_{[1/c,c]}(r) := \mathbb{I}\{1/c \leq r \leq c\}$.

Although this strategy has also been verified on various foundation models [28, 23, 32, 25, 24], *there exists a central practical difficulty:* π_{old} *may be missing*. Specifically, as shown in Figure 2, the rollout version old is often outdated with respect to both the behavior model and the actor model, and therefore may have already been discarded. This is particularly common in training systems that involve partial rollout collection and asynchronous model updates [6, 20, 21, 26].

In practice, existing works therefore often replace π_{old} with an approximation, for example by using a linearly interpolated policy [12] or, more generally, a policy version between old and θ as a surrogate. Such a decomposition does not affect the algebraic correctness of the loss function, but the two factors no longer correspond to pure discrepancy repair and pure staleness correction. This semantic entanglement is exactly the old-logit mismatch problem.

4 A Unified Analysis of Decoupled Correction

In this section, we first provide intuition on why discrepancy repair cannot substitute for staleness correction; namely, why the decoupled approach in Eq. (4) cannot be replaced by the standard PPO clip in Eq. (3). We then provide an analysis of existing off-policy corrections, demonstrating how they can be unified into the form of Eq. (4). Furthermore, we explicitly explain how the old-logit mismatch problem can lead to correction failures within the current framework.

4.1 Why Discrepancy Repair Cannot Substitute for Staleness Correction

The intuition for why the dual-side correction in Eq. (4) cannot simply be expressed by the standard PPO correction in Eq. (3) is two-fold. First, PPO primarily prevents overly large update steps by applying an asymmetric filter based on the advantage sign, whereas training-inference discrepancy repair requires a strict, symmetric constraint centered around 1. Second, blending these decomposed terms into a single ratio forces a shared threshold, fundamentally compromising optimization. Because discrepancy repair targets numerical consistency while staleness correction controls update magnitude, they naturally demand different levels of constraint strength. A strict shared constraint stably filters out errors but severely bottlenecks learning, whereas a looser constraint accelerates early training but exposes the policy to noisy, compounded updates that increase the risk of oscillation or collapse.

We further quantify this effect in Section 6.4, where exact-old-logit experiments show how discrepancy masking and PPO-CLIP still interact through the final active-token set.

4.2 A Unified View of Existing Off-Policy Corrections

As shown in Table 1, existing off-policy methods in LLM RL generally decouple the optimization process into a discrepancy ratio r_d and a staleness ratio r_s. In synchronous settings, an accessible and semantically correct old policy π_{old} allows for an exact decomposition of training-inference discrepancy and policy staleness. However, in asynchronous RL, the latency between training and generation engines introduces an unavoidable version mismatch ($\theta \geq$ async $\geq$ old). This breaks the semantic consistency of the reference policy, corrupting the meanings of both r_d and r_s and causing standard decoupled corrections to fail.

Table 1: Summarization of PPO variants under a unified Masked Importance Sampling (MIS) view.

Algorithm	**Discrepancy Ratio** r_d	**Staleness Ratio** r_s	**Proxy Definition**
General Format: $\text{MIS} = M_{[1/c,c]}(r_d)\Big(\mathbb{I}_{A_t\geq 0}\mathbb{I}_{r_s\leq 1+\epsilon} + \mathbb{I}_{A_t<0}\mathbb{I}_{r_s\geq 1-\epsilon}\Big)$			
PPO-clip (standard)	1	$\frac{\pi_\theta}{\pi_{\text{old}}}$	–
PPO-clip & train_infer	1	$\frac{\pi_\theta}{\mu_{\text{old}}}$	–
decoupled PPO & train_infer	$\frac{\pi_{\text{old}}}{\mu_{\text{old}}}$	$\frac{\pi_\theta}{\pi_{\text{old}}}$	–
PPO-EWMA	1	$\frac{\pi_\theta}{\pi_{\text{prox}}}$	$\theta_{\text{prox},t} = \beta\theta_{\text{prox},t-1} + (1-\beta)\theta_t$
PPO-EWMA & train_infer	$\frac{\pi_{\text{prox}}}{\mu_{\text{old}}}$	$\frac{\pi_\theta}{\pi_{\text{prox}}}$	$\theta_{\text{prox},t} = \beta\theta_{\text{prox},t-1} + (1-\beta)\theta_t$
linear_prox	1	$\frac{\pi_\theta}{\pi_{\text{prox}}}$	$\pi_{\text{prox}} = \alpha\mu_{\text{old}} + (1-\alpha)\pi_\theta$
linear_prox & train_infer	$\frac{\pi_{\text{prox}}}{\mu_{\text{old}}}$	$\frac{\pi_\theta}{\pi_{\text{prox}}}$	$\pi_{\text{prox}} = \alpha\mu_{\text{old}} + (1-\alpha)\pi_\theta$
decoupled PPO & Async	$\frac{\pi_{\text{async}}}{\mu_{\text{old}}}$	$\frac{\pi_\theta}{\pi_{\text{async}}}$	$\theta \geq$ async $\geq$ old

To mitigate this missing reference without heavy infrastructure overhead, a common strategy is to construct a probability-space proximal policy π_{prox} through interpolation between the current and behavior policies, such as `linear_prox` or token-wise log-linear interpolation. However, this approach does not genuinely resolve the discrepancy, as stated below:

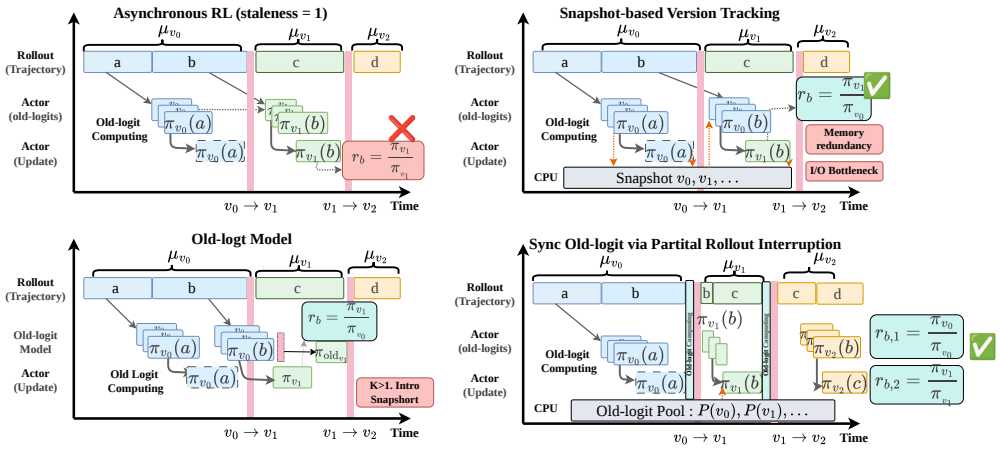


Figure 2: Exact old-logit acquisition in asynchronous RL. The top-left panel shows the original mismatch problem. The remaining panels show snapshot-based version tracking, a dedicated old-logit model, and synchronization via partial rollout interruption.

Proposition 1. *Let $r(\theta) = \pi_\theta / \mu_{\text{old}}$. If π_{prox} is constructed via arithmetic interpolation or token-wise log-linear interpolation, then clipping and masking on the decoupled ratios merely re-parameterizes the effective constraint boundaries of the single total ratio $r(\theta)$. Full derivations are provided in Appendix A.*

Because interpolation only shifts effective boundaries rather than restoring an exact reference, we explore two distinct directions to resolve the old-logit mismatch in the following sections. The first approach relies on systematic infrastructure support to directly acquire the ground-truth old logits. The second acknowledges limited system overhead and constructs a more reliable, approximate reference using a revised exponential moving average that explicitly accounts for asynchronous delays.

5 Recovering and Approximating Old Logits

5.1 Exact Old-Logit Acquisition

We first consider exact acquisition of $\pi_{\text{old}}(y_t|x, y_{<t})$, the training-side token probability under the rollout version. Figure 2 illustrates three possible strategies.

Snapshot-based version tracking. The most direct solution is to retain historical parameter snapshots and reload the version that generated each token or trajectory. This gives the cleanest estimate of π_{old} and therefore restores the semantic decomposition in Eq. (1). Its drawback is system cost. Snapshot retention requires additional CPU or host memory, and exact recovery may require frequent actor-side version switching. With partial rollouts, a single sample can span multiple versions, which further increases switching and I/O overhead.

Dedicated old-logit model. A second option is to maintain a separate model that computes old logits while the main actor continues training. This can reduce contention on the actor path and allow overlap between old-logit computation and gradient updates. It also decouples old-logit computation from update training, so the overlap can reduce the end-to-end time of the actor stage.

Synchronization via partial rollout interruption. A third option computes old logits before a policy version disappears. Before updating parameters from version v to $v+1$, the system interrupts rollout workers and returns partial trajectories. Since rollout is stopped during this interval, we can use Ray scheduling to release the rollout-side placement and temporarily switch the same resources to actor-side old-logit computation. The still-resident version v is then used to compute exact old

logits for the returned partial trajectories. After the old-logit pass finishes, the system switches the resources back to rollout execution and resumes generation.

This design avoids storing old weights and can provide exact logits, but it introduces synchronization stalls, resource reconfiguration overhead, and disruption to rollout parallelism.

These three methods represent different points in a system trade-off: snapshots are exact but memory- and I/O-heavy; old-logit models enable overlap but require resource partitioning; partial interruption avoids historical storage but adds synchronization overhead.

5.2 Revised PPO-EWMA as a Low-Cost Reference Policy

Exact old-logit acquisition may be too expensive for large asynchronous Agentic RL. We therefore use a (PPO-EWMA) reference policy as a low-cost approximation [9]. The goal is not to claim exact recovery of π_{old}, but to construct a smoother reference that better tracks the center of the asynchronous version window than either the current policy or a static interpolation proxy.

PPO-EWMA maintains π_{prox} as an exponentially averaged reference policy. Given actor parameters $\theta^{(t)}$ after update step t, we use

$$\theta_{\text{prox}}^{(t)} = \frac{\sum_{k=0}^{t} \beta_{\text{prox}}^{t-k} \theta^{(k)}}{\sum_{k=0}^{t} \beta_{\text{prox}}^{t-k}}, \qquad r_s = \frac{\pi_\theta}{\pi_{\text{prox}}}, \qquad r_d = \frac{\pi_{\text{prox}}}{\mu_{\text{old}}}. \tag{5}$$

That is, we replace the unavailable π_{old} with π_{prox} for both staleness correction and discrepancy repair. This is only an approximate reference, not an exact recovery of old logits. The equivalent recursive update is provided in Appendix B.

OOur adjustment is deliberately small. First, we align the EWMA mean age with the observed loss-token-weighted mean staleness $\hat{s}$, giving $\beta_{\text{prox}} = \hat{s}/(1+\hat{s})$. In our runs, $\hat{s} \approx 3$ under a maximum version gap of three, motivating $\beta_{\text{prox}} = 0.75$. Appendix B gives the derivation.

Second, we add an automatic reset to avoid accumulating excessively stale versions in the EWMA reference. As θ_{prox} averages over more historical actor states, it may drift away from the policy version used by recent rollouts. This makes the discrepancy ratio $\pi_{\text{prox}}/\mu_{\text{old}}$ deviate further from one and causes the Train-Infer Mask to reject many tokens. We therefore monitor the Train-Infer Mask value ρ_t, defined as the fraction of tokens that remain active after discrepancy masking and PPO clipping. When $\rho_t < \tau$, where τ is set to 0.9 in our experiments unless otherwise stated, we reset

$$\theta_{\text{prox}}^{(t)} \leftarrow \theta^{(t)}. \tag{6}$$

This clears stale history and re-centers the proxy reference around the current actor. Section 6.5 provides a more detailed analysis of this behavior.

6 Experiments

6.1 Experimental Setup

We evaluate Agentic RL tasks using two representative policy backbones: the dense `Qwen3-4B` model and the MoE `Qwen3-30B-A3B` model. Training data are drawn from an Agentic RL corpus spanning multiple environments. Evaluation covers the retail, airline, and telecom domains of τ^2-Bench [3], together with the in-store and delivery splits of VitaBench [8]. For both benchmarks, we report task-level average success and pass metrics.

To isolate the effect studied in this paper, we adopt an asynchronous RL setup with explicit control over the maximum version gap between rollout workers and the actor, which we cap at three. We eliminate additional staleness from minibatch reuse and multiple PPO epochs where possible, ensuring that the observed staleness is primarily due to rollout–training asynchrony. Detailed hyperparameters, resource configurations, and environment descriptions are provided in Appendix F.

We consider several practical methods: **Decoupled PPO**, which employs the decoupled objective but relies on the available asynchronous reference rather than the true π_{old} policy; **Linear_prox**, a lightweight linear interpolation-based proximal policy; and **PPO-EWMA**, our enhanced EWMA reference with staleness-aware decay and optional reset. Additionally, we include **Snapshot**, which

Table 2: Main results on the held-out Agentic benchmark suite for the dense `Qwen3-4B` model and the `Qwen3-30B-A3B` MoE model. VitaBench is separated into in-store and delivery splits. Snapshot† denotes an idealized setting where exact old logits are available. The best performance among practical methods (first three rows) is **bolded**.

Backbone	Method	τ^2-Bench (Retail)		τ^2-Bench (Airline)		τ^2-Bench (Telecom)		VitaBench (In-store)		VitaBench (Delivery)	
		avg@4	pass@4	avg@4	pass@4	avg@2	pass@2	avg@2	pass@2	avg@2	pass@2
`Qwen3-4B`	Decoupled PPO	63.96	88.60	53.5	72	40	50	19.83	37	19.56	33
`Qwen3-4B`	Linear_prox	64.40	86.84	**54**	72	37.5	50	22.37	40	19.10	28
`Qwen3-4B`	PPO-EWMA	**65.72**	**90.35**	**54**	**74**	**42.5**	**52.5**	**25**	**50**	**25.88**	**39**
`Qwen3-4B`	Snapshot†	66.23	89.47	56	76	42.5	52.5	28.89	47	27.33	42
`Qwen3-30B-A3B`	Decoupled PPO	65.43	89.47	57	76	44.75	55	18.28	32	25.88	39
`Qwen3-30B-A3B`	Linear_prox	65.8	87.7	53.5	74	44	55	31.47	47	20.74	33
`Qwen3-30B-A3B`	PPO-EWMA	**67.82**	**92.1**	**60**	**82**	**45**	**57.5**	**33.41**	**48**	**28.49**	**43**
`Qwen3-30B-A3B`	Snapshot†	69.70	92.1	59	80	45	57.5	34.62	50	30.74	45

Table 3: System-overhead measurement protocols for exact old-logit acquisition. Snapshot recovery mainly adds version-switch latency and CPU storage; a dedicated old-logit model changes the overlapped time allocation.

(a) Snapshot vs PPO-EWMA.

Metric	4B	30B	4B	30B
Switch latency (s)	7	14.2	7	14.2
CPU storage (GB)	40	76.4	7.9	15.2
Extra time (s)	25	150	8	34

(b) Dedicated old-logit model.

Ratio	Old-logit (s)	Update (s)	Single (s)	Overlap (s)	Change (%)
1:2	243	272	305	284	-6.8
1:3	243	206	237	254	+7.17

recovers exact old logits via true version tracking. This is feasible in our setup because we control the maximum version gap.

6.2 Main Results

Table 2 compares exact old-logit recovery, proxy references, and PPO-EWMA on held-out Agentic benchmarks. Snapshot† serves as an idealized reference because it assumes exact old logits are available. PPO-EWMA is the practical method we emphasize: it consistently improves over Decoupled PPO and Linear_prox, and often approaches the Snapshot results.

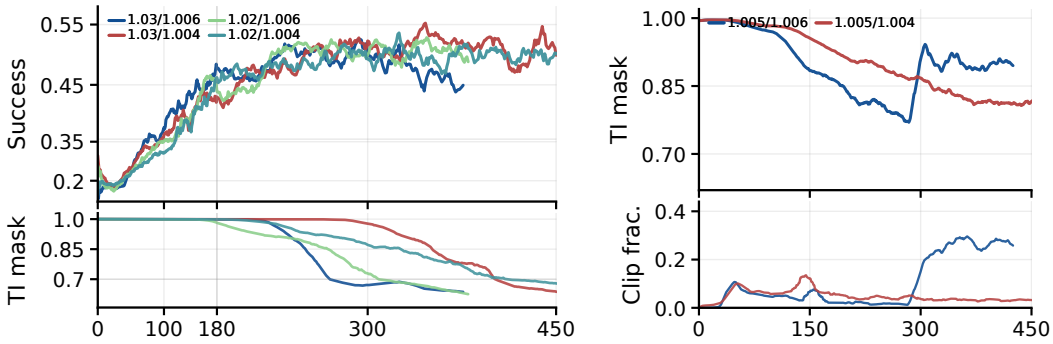
On the dense 4B model, PPO-EWMA obtains the best pass@4 on retail and the best pass@2 on VitaBench in-store, while tying the best telecom scores. On the 30B MoE model, it is strongest on the airline split and ties the best retail pass@4 and telecom avg@2. These results suggest that a maintained EWMA reference can recover much of the benefit of exact correction without requiring exact old-logit recovery.

6.3 System Overhead of Exact Old-Logit Acquisition

Table 3 summarizes the system cost of exact old-logit acquisition. Snapshot is accurate but expensive: it requires version switching, snapshot storage, and extra recovery time, with the cost becoming much larger on the 30B MoE model.

PPO-EWMA is much cheaper because it only maintains a lightweight proxy reference. In the left subtable, its CPU storage and extra time are far lower than Snapshot for both backbones. The right subtable shows that a dedicated old-logit model can overlap part of the computation, but its benefit depends on the resource partition ratio and still requires additional model-side infrastructure.

Overall, Table 3 supports the same conclusion as Table 2: exact recovery is useful as a strong reference, but PPO-EWMA offers a better performance–cost trade-off for practical asynchronous training.



(a) Threshold trade-off.

(b) Mask–clip interaction.

Figure 3: Exact-old-logit threshold analysis. Each label reports the discrepancy and stale-policy thresholds.

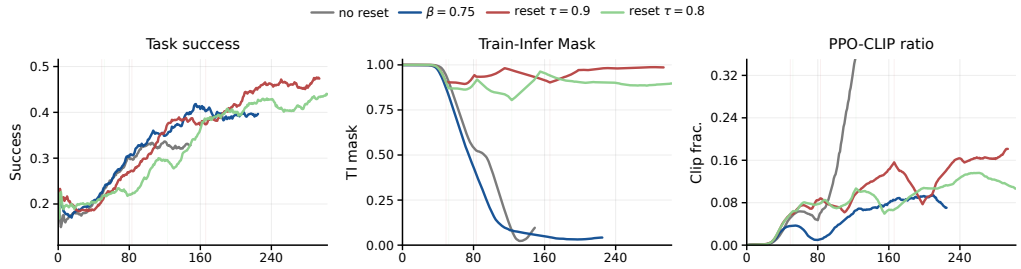


Figure 4: Effect of automatic reset for $\beta = 0.75$. Resetting the EWMA reference when the Train-Infer Mask becomes too low prevents late-stage collapse. Vertical lines mark reset events.

6.4 Threshold Trade-off under Exact Old Logits

Using exact old logits from Snapshot, we study how the discrepancy and stale-policy thresholds affect optimization. All threshold-tradeoff curves in this section are collected with the `Qwen3-4B` backbone. In each run name, the first number denotes the discrepancy threshold and the second denotes the stale-policy threshold; for example, `snap1003_1006` corresponds to 1.003 and 1.006.

Figure 3 summarizes the trade-off. A looser discrepancy threshold keeps more tokens active and improves early learning, but it also admits more biased off-policy tokens and can destabilize later trajectories. A stricter threshold slows the beginning but yields smoother retained-token dynamics and reduces late collapse. Appendix D shows the same pattern for nearby settings.

Panel 3b further shows that discrepancy masking and PPO clipping are coupled. With the same discrepancy threshold, a looser stale-policy threshold lets more problematic tokens survive early, causing the Train-Infer Mask to drop lower; later, stronger PPO clipping caps the remaining updates and helps the mask recover.

6.5 PPO-EWMA Analysis and Ablation

We next study the revised PPO-EWMA through three training-time signals: Task success, Train-Infer Mask, and PPO-CLIP ratio. The core issue is reference-policy lag: a useful EWMA reference must be smooth enough to stabilize updates, but not so stale that it collapses the Train-Infer Mask. We therefore use a staleness-aware decay and add automatic reset to re-center the reference when the mask becomes too low. Detailed decay and threshold ablations are provided in Appendix E.

Figure 4 shows the main result. Without reset, the EWMA reference can drift far enough that Train-Infer Mask collapses. With reset, only a few re-centering events are needed to recover a high Train-Infer Mask value while preserving most of the early Task success gain of $\beta = 0.75$. This supports the practical design: use EWMA as a low-cost approximate reference, but reset it when the proxy becomes too stale.

Table 4: Deviations from Snapshot at staleness three (lower is better). LP denotes token-level log-probability difference.

	Linear_prox		PPO-EWMA	
Step	**KL**	**LP**	**KL**	**LP**
10	0.0053	0.0103	0.0070	0.0104
100	0.0067	0.0124	0.0040	0.0070
300	0.0061	0.0105	0.0030	0.0050
500	0.0052	0.0107	0.0030	0.0050

Table 5: Reference-scoring costs. N_{AF} is the dedicated ActorFwd GPU allocation; CPU storage excludes GPU-resident state.

Method	**Scoring FLOPs ($\times$ batch forward)**	**Additional GPUs**	**Additional CPU storage (GB)**	**Version shards per batch**	**Additional time (s/step)**
PPO-EWMA	1.00	0	15.2	1.0	30
Exact trailing prefill	1.78	N_{AF}	None	2.4	60
Snapshot recovery	1.83	0	76.4	2.4	150

6.6 Reference Fidelity and System Cost

Proxy fidelity. Table 4 compares both proxies against Snapshot-recovered old logits at staleness three. PPO-EWMA has smaller KL and token log-probability deviations from step 100 onward, although not at step 10. This supports its fidelity in the evaluated setting, without implying exact recovery. Unlike the output-space interpolations in Proposition 1, PPO-EWMA averages historical parameters and generally does not satisfy those interpolation identities.

System cost. Table 5 uses Qwen3-30B-A3B with a common batch token budget and asynchronous configuration: maximum version gap three and approximately 2.4 versions per trajectory. Scoring FLOPs include replayed prefixes and are normalized by one additional full-batch forward, excluding shared training computation. CPU storage counts additional persistent model state; time includes scoring, synchronization, switching, and restoration.

PPO-EWMA still requires a forward pass, but avoids scoring repeated prefixes under multiple versions. For segments l_0, l_1, l_2 generated by v_0, v_1, v_2, respectively, let $P_v(x)$ denote training-side prefill of x under version v. Without reusable version-matched prefix state, the workloads are

$$C_{\mathrm{exact}} = P_{v_0}(l_0) + P_{v_1}(l_0 \| l_1) + P_{v_2}(l_0 \| l_1 \| l_2), \tag{7}$$

$$C_{\mathrm{EWMA}} = P_{\mathrm{prox}}(l_0 \| l_1 \| l_2), \tag{8}$$

where $\|$ denotes concatenation. Exact scoring reprocesses l_0 three times and l_1 twice to score each segment under its generating version; EWMA scores the full sequence once. Version-matched caches can reduce replay, but require separate accounting of cache storage and replayed tokens. Trailing prefill also needs ActorFwd resources, while Snapshot adds historical-weight storage and switching. The measured savings concern this scoring configuration, rather than a universal end-to-end speedup.

7 Conclusion

We identified and studied the missing-old-logit problem in asynchronous Agentic RL for LLMs, which undermines decoupled off-policy correction. We address it from two directions: exact old-logit recovery at the infrastructure level, and low-cost approximation with a revised PPO-EWMA reference. Our results show that exact recovery improves correction fidelity, while PPO-EWMA provides a practical alternative when exact recovery is too expensive. A limitation is that PPO-EWMA remains an approximate reference rather than a true reconstruction of the missing old policy, and may become unreliable under highly non-stationary staleness or extreme version gaps.

Acknowledgements and Disclosures

This study was partially funded by the National Natural Science Foundation of China (72471165,62502340), Beijing-Tianjin-Hebei Natural Science Foundation Cooperation Project(25JJJJC0045), Emerging Frontiers Cultivation Program of Tianjin University Interdisciplinary Center and Strategic Leading Special Project of Tianjin University.

References

[1] Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce-style optimization for learning from human feedback in llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12248–12267, 2024.

[2] Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms, 2024.

[3] Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment. *arXiv preprint arXiv:2506.07982*, 2025.

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

[5] Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, et al. Agentic reinforced policy optimization. *arXiv preprint arXiv:2507.19849*, 2025.

[6] Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, WANG JIASHU, et al. Areal: A large-scale asynchronous reinforcement learning system for language reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[7] Zhong Guan, Haoran Sun, Yongjian Guo, Shuai Di, Xiaodong Bai, Jing Long, Tianyun Zhao, Mingxi Luo, Chen Zhou, Yucheng Guo, et al. Rl-vla3: Reinforcement learning vla accelerating via full asynchronism. *arXiv preprint arXiv:2602.05765*, 2026.

[8] Wei He, Yueqing Sun, Hongyan Hao, Xueyuan Hao, Zhikang Xia, Qi Gu, Chengcheng Han, Dengchang Zhao, Hui Su, Kefeng Zhang, et al. Vitabench: Benchmarking llm agents with versatile interactive tasks in real-world applications. *arXiv preprint arXiv:2509.26490*, 2025.

[9] Jacob Hilton, Karl Cobbe, and John Schulman. Batch size-invariance for policy optimization. *Advances in Neural Information Processing Systems*, 35:17086–17098, 2022.

[10] Luke J Huang, Zhuoyang Zhang, Qinghao Hu, Shang Yang, and Song Han. Stable asynchrony: Variance-controlled off-policy rl for llms. *arXiv preprint arXiv:2602.17616*, 2026.

[11] 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*, pages 611–626, 2023.

[12] Xiaocan Li, Shiliang Wu, and Zheng Shen. A-3po: Accelerating asynchronous llm training with staleness-aware proximal policy approximation. *arXiv preprint arXiv:2512.06547*, 2025.

[13] Jiacai Liu, Yingru Li, Yuqian Fu, Jiawei Wang, Qian Liu, and Zhuo Jiang. When speed kills stability: Demystifying RL collapse from the training-inference mismatch. `https://richardli.xyz/rl-collapse`, September 2025. Online article.

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

[15] Penghui Qi, Xiangxin Zhou, Zichen Liu, Tianyu Pang, Chao Du, Min Lin, and Wee Sun Lee. Rethinking the trust region in llm reinforcement learning. *arXiv preprint arXiv:2602.04879*, 2026.

[16] Nicolas Le Roux, Marc G Bellemare, Jonathan Lebensold, Arnaud Bergeron, Joshua Greaves, Alex Fréchette, Carolyne Pelletier, Eric Thibodeau-Laufer, Sándor Toth, and Sam Work. Tapered off-policy reinforce: Stable and efficient reinforcement learning for llms. *arXiv preprint arXiv:2503.14286*, 2025.

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

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

[19] Guobin Shen, Chenxiao Zhao, Xiang Cheng, Lei Huang, and Xing Yu. Vespo: Variational sequence-level soft policy optimization for stable off-policy llm training. *arXiv preprint arXiv:2602.10693*, 2026.

[20] Guangming Sheng, Yuxuan Tong, Borui Wan, Wang Zhang, Chaobo Jia, Xibin Wu, Yuqi Wu, Xiang Li, Chi Zhang, Yanghua Peng, et al. Laminar: A scalable asynchronous rl post-training framework. *arXiv preprint arXiv:2510.12633*, 2025.

[21] 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*, pages 1279–1297, 2025.

[22] Zhenpeng Su, Leiyu Pan, Xue Bai, Dening Liu, Guanting Dong, Jiaming Huang, Minxuan Lv, Wenping Hu, Fuzheng Zhang, Kun Gai, et al. Klear-reasoner: Advancing reasoning capability via gradient-preserving clipping policy optimization. *arXiv preprint arXiv:2508.07629*, 2025.

[23] Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, SH Cai, Yuan Cao, Y Charles, HS Che, Cheng Chen, Guanduo Chen, et al. Kimi k2. 5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*, 2026.

[24] Ling Team, Anqi Shen, Baihui Li, Bin Hu, Bin Jing, Cai Chen, Chao Huang, Chao Zhang, Chaokun Yang, Cheng Lin, et al. Every step evolves: Scaling reinforcement learning for trillion-scale thinking model. *arXiv preprint arXiv:2510.18855*, 2025.

[25] Haifeng Wang, Hua Wu, Tian Wu, Yu Sun, Jing Liu, Dianhai Yu, Yanjun Ma, Jingzhou He, Zhongjun He, Dou Hong, et al. Ernie 5.0 technical report. *arXiv preprint arXiv:2602.04705*, 2026.

[26] Weixun Wang, Shaopan Xiong, Gengru Chen, Wei Gao, Sheng Guo, Yancheng He, Ju Huang, Jiaheng Liu, Zhendong Li, Xiaoyang Li, et al. Reinforcement learning optimization for large-scale learning: An efficient and user-friendly scaling library. *arXiv preprint arXiv:2506.06122*, 2025.

[27] Weixun Wang, XiaoXiao Xu, Wanhe An, Fangwen Dai, Wei Gao, Yancheng He, Ju Huang, Qiang Ji, Hanqi Jin, Xiaoyang Li, et al. Let it flow: Agentic crafting on rock and roll, building the rome model within an open agentic learning ecosystem. *arXiv preprint arXiv:2512.24873*, 2025.

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

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

[30] Feng Yao, Liyuan Liu, Dinghuai Zhang, Chengyu Dong, Jingbo Shang, and Jianfeng Gao. Your efficient rl framework secretly brings you off-policy rl training, august 2025. *URL https://fengyao. notion. site/off-policy-rl*, 2025.

[31] 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.

[32] Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, et al. Glm-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026.

[33] Guibin Zhang, Hejia Geng, Xiaohang Yu, Zhenfei Yin, Zaibin Zhang, Zelin Tan, Heng Zhou, Zhong-Zhi Li, Xiangyuan Xue, Yijiang Li, et al. The landscape of agentic reinforcement learning for llms: A survey. *Transactions on Machine Learning Research*, 2025.

[34] Xin Zhao, Yongkang Liu, Kuan Xu, Jia Guo, Zihao Wang, Yan Sun, Xinyu Kong, Qianggang Cao, Liang Jiang, Zujie Wen, et al. Small leak can sink a great ship–boost rl training on moe with icepop!, 2025.

[35] Chujie Zheng, Kai Dang, Bowen Yu, Mingze Li, Huiqiang Jiang, Junrong Lin, Yuqiong Liu, Hao Lin, Chencan Wu, Feng Hu, et al. Stabilizing reinforcement learning with llms: Formulation and practices. *arXiv preprint arXiv:2512.01374*, 2025.

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

[37] Haizhong Zheng, Jiawei Zhao, and Beidi Chen. Prosperity before collapse: How far can off-policy rl reach with stale data on llms? *arXiv preprint arXiv:2510.01161*, 2025.

[38] Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody H Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. Sglang: Efficient execution of structured language model programs. *Advances in neural information processing systems*, 37:62557–62583, 2024.

[39] Zilin Zhu, Chengxing Xie, Xin Lv, and slime Contributors. slime: An llm post-training framework for rl scaling. `https://github.com/THUDM/slime`, 2025. GitHub repository. Corresponding author: Xin Lv.

A Detailed Derivations for Interpolation-Based Proxies

This appendix gives the derivation behind Proposition 1. We write the total ratio between the current training policy and the behavior-side reference as:

$$r_\theta = \frac{\pi_\theta(y|x)}{\mu_{\text{old}}(y|x)}. \tag{9}$$

For a proximal policy π_{prox}, the decoupled objective applies a discrepancy mask to

$$r_d = \frac{\pi_{\text{prox}}(y|x)}{\mu_{\text{old}}(y|x)} \tag{10}$$

and PPO clipping to

$$r_s = \frac{\pi_\theta(y|x)}{\pi_{\text{prox}}(y|x)}. \tag{11}$$

For simplification, we consider the threshold for masking as ϵ_1 and ϵ_2, instead of c, i.e., the masked importance sampling is:

$$\text{MIS} = M_{[1-\epsilon_1,1+\epsilon_1]}(r_d)\left(\mathbb{I}\{A_t \geq 0\}\mathbb{I}\{r_s \leq 1+\epsilon_2\} + \mathbb{I}\{A_t < 0\}\mathbb{I}\{r_s \geq 1-\epsilon_2\}\right). \tag{12}$$

The key question is whether constructing π_{prox} by interpolation introduces a genuinely new reference, or whether it only changes the effective constraints on r_θ. We use α for the behavior-policy weight, consistent with Table 1.

Log-linear interpolation. Consider the token-wise log-linear interpolation:

$$\log \widetilde{\pi}_{\text{prox}}(y|x) = \alpha \log \mu_{\text{old}}(y|x) + (1-\alpha) \log \pi_\theta(y|x), \quad \alpha \in (0,1). \tag{13}$$

In this case, the discrepancy ratio becomes:

$$r_d = \frac{\widetilde{\pi}_{\text{prox}}}{\mu_{\text{old}}} = \left(\frac{\pi_\theta}{\mu_{\text{old}}}\right)^{1-\alpha} = r_\theta^{1-\alpha}. \tag{14}$$

The mask condition $1-\epsilon_1 \le r_d \le 1+\epsilon_1$ restricts r_θ to the interval:

$$(1-\epsilon_1)^{1/(1-\alpha)} \le r_\theta \le (1+\epsilon_1)^{1/(1-\alpha)}. \tag{15}$$

The PPO-side ratio is $r_s = \pi_\theta / \widetilde{\pi}_{\text{prox}} = r_\theta^\alpha$. We analyze the constraints based on the sign of A_t:

- **Case** $A_t \ge 0$: The clip requires $r_s \le 1+\epsilon_2$, implying $r_\theta \le (1+\epsilon_2)^{1/\alpha}$. Combined with the mask, the active region for r_θ is:

$$(1-\epsilon_1)^{1/(1-\alpha)} \le r_\theta \le \min\left((1+\epsilon_1)^{1/(1-\alpha)}, (1+\epsilon_2)^{1/\alpha}\right). \tag{16}$$

- **Case** $A_t < 0$: The clip requires $r_s \ge 1-\epsilon_2$, implying $r_\theta \ge (1-\epsilon_2)^{1/\alpha}$. Combined with the mask, the active region is:

$$\max\left((1-\epsilon_1)^{1/(1-\alpha)}, (1-\epsilon_2)^{1/\alpha}\right) \le r_\theta \le (1+\epsilon_1)^{1/(1-\alpha)}. \tag{17}$$

First-order expansion shows $\epsilon_{1,\text{eff}} \approx \frac{\epsilon_1}{1-\alpha}$ and $\epsilon_{2,\text{eff}} \approx \frac{\epsilon_2}{\alpha}$. This shows that log-linear interpolation merely re-parameterizes the active region of the original ratio r_θ.

Arithmetic interpolation. Consider the linear proxy:

$$\pi_{\text{prox}}(y|x) = \alpha \mu_{\text{old}}(y|x) + (1-\alpha)\pi_\theta(y|x), \quad \alpha \in (0,1). \tag{18}$$

The discrepancy ratio is $r_d = \alpha + (1-\alpha) r_\theta$, so the mask condition $1-\epsilon_1 \le r_d \le 1+\epsilon_1$ becomes:

$$1 - \frac{\epsilon_1}{1-\alpha} \le r_\theta \le 1 + \frac{\epsilon_1}{1-\alpha}. \tag{19}$$

The PPO-side ratio is $r_s = \frac{r_\theta}{\alpha + (1-\alpha) r_\theta}$. Again, we analyze by the sign of A_t:

- **Case** $A_t \ge 0$: Solving $r_s \le 1+\epsilon_2$ for r_θ gives:

$$r_\theta \le \frac{\alpha(1+\epsilon_2)}{1-(1-\alpha)(1+\epsilon_2)}. \tag{20}$$

  The active region is the intersection of the lower mask bound and the tighter of the two upper bounds.

- **Case** $A_t < 0$: Solving $r_s \ge 1-\epsilon_2$ for r_θ gives:

$$r_\theta \ge \frac{\alpha(1-\epsilon_2)}{1-(1-\alpha)(1-\epsilon_2)}. \tag{21}$$

  The active region is the intersection of the upper mask bound and the tighter of the two lower bounds.

Applying $(1-x)^{-1} \approx 1+x$ reveals an effective radius $\epsilon_{2,\text{eff}} \approx \frac{\epsilon_2}{\alpha}$. Thus, arithmetic interpolation also functions as a set of scaled constraints on the single total ratio r_θ.

Numerical effective thresholds. Table 6 instantiates the above mappings for version gaps $n \in \{1,2,3\}$, where the behavior-policy weight is $\alpha = 1/(n+1)$. We include the thresholds used in our experiments and several nearby settings. The clip intervals are allowed to be asymmetric because our implementation uses separate lower and upper PPO-CLIP thresholds; the symmetric PPO interval is a special case. Across these settings, interpolation changes the effective mask and clip ranges on the same total ratio rather than introducing an independent old-policy reference.

Original mask	Original clip	n	Interpolation	Mask on r_d	Clip on r_s
$[0.990, 1.010]$	$[0.997, 1.004]$	1	Linear	$[0.9800, 1.0200]$	$[0.9940, 1.0080]$
$[0.990, 1.010]$	$[0.997, 1.004]$	1	Log-linear	$[0.9801, 1.0201]$	$[0.9940, 1.0080]$
$[0.990, 1.010]$	$[0.997, 1.004]$	2	Linear	$[0.9850, 1.0150]$	$[0.9911, 1.0121]$
$[0.990, 1.010]$	$[0.997, 1.004]$	2	Log-linear	$[0.9850, 1.0150]$	$[0.9910, 1.0120]$
$[0.990, 1.010]$	$[0.997, 1.004]$	3	Linear	$[0.9867, 1.0133]$	$[0.9881, 1.0162]$
$[0.990, 1.010]$	$[0.997, 1.004]$	3	Log-linear	$[0.9867, 1.0134]$	$[0.9881, 1.0161]$
$[0.995, 1.005]$	$[0.997, 1.004]$	1	Linear	$[0.9900, 1.0100]$	$[0.9940, 1.0080]$
$[0.995, 1.005]$	$[0.997, 1.004]$	1	Log-linear	$[0.9900, 1.0100]$	$[0.9940, 1.0080]$
$[0.995, 1.005]$	$[0.997, 1.004]$	2	Linear	$[0.9925, 1.0075]$	$[0.9911, 1.0121]$
$[0.995, 1.005]$	$[0.997, 1.004]$	2	Log-linear	$[0.9925, 1.0075]$	$[0.9910, 1.0120]$
$[0.995, 1.005]$	$[0.997, 1.004]$	3	Linear	$[0.9933, 1.0067]$	$[0.9881, 1.0162]$
$[0.995, 1.005]$	$[0.997, 1.004]$	3	Log-linear	$[0.9933, 1.0067]$	$[0.9881, 1.0161]$
$[0.990, 1.010]$	$[0.996, 1.006]$	1	Linear	$[0.9800, 1.0200]$	$[0.9920, 1.0121]$
$[0.990, 1.010]$	$[0.996, 1.006]$	1	Log-linear	$[0.9801, 1.0201]$	$[0.9920, 1.0120]$
$[0.990, 1.010]$	$[0.996, 1.006]$	2	Linear	$[0.9850, 1.0150]$	$[0.9881, 1.0182]$
$[0.990, 1.010]$	$[0.996, 1.006]$	2	Log-linear	$[0.9850, 1.0150]$	$[0.9880, 1.0181]$
$[0.990, 1.010]$	$[0.996, 1.006]$	3	Linear	$[0.9867, 1.0133]$	$[0.9842, 1.0244]$
$[0.990, 1.010]$	$[0.996, 1.006]$	3	Log-linear	$[0.9867, 1.0134]$	$[0.9841, 1.0242]$
$[0.980, 1.020]$	$[0.997, 1.004]$	1	Linear	$[0.9600, 1.0400]$	$[0.9940, 1.0080]$
$[0.980, 1.020]$	$[0.997, 1.004]$	1	Log-linear	$[0.9604, 1.0404]$	$[0.9940, 1.0080]$
$[0.980, 1.020]$	$[0.997, 1.004]$	2	Linear	$[0.9700, 1.0300]$	$[0.9911, 1.0121]$
$[0.980, 1.020]$	$[0.997, 1.004]$	2	Log-linear	$[0.9702, 1.0301]$	$[0.9910, 1.0120]$
$[0.980, 1.020]$	$[0.997, 1.004]$	3	Linear	$[0.9733, 1.0267]$	$[0.9881, 1.0162]$
$[0.980, 1.020]$	$[0.997, 1.004]$	3	Log-linear	$[0.9734, 1.0268]$	$[0.9881, 1.0161]$

Table 6: Effective thresholds on the total ratio $r_\theta = \pi_\theta/\mu_{\text{old}}$ induced by interpolation-based proxies. The first block corresponds to the thresholds used in our experiment. For version gap n, we set the behavior-policy weight $\alpha = 1/(n+1)$.

Figure 5 provides an empirical check of the same reparameterization view. After removing the interpolation and applying the corresponding reparameterized constraints directly, the training curve is almost identical to the log-linear interpolation run. This supports the interpretation that the interpolation proxy mainly changes the effective ratio boundary rather than introducing an independent correction effect.

These derivations explain why interpolation-based proxies can act as useful stabilizers but should not be interpreted as exact training–inference discrepancy repair. The ratios remain deterministic transformations of $r_\theta = \pi_\theta/\mu_{\text{old}}$; no term reconstructs the missing training-side old policy logits.

B PPO-EWMA Update Details

This appendix records the standard PPO-EWMA parameterization used in Section 5.2. It is included for completeness; the main text focuses on the two asynchronous adjustments, namely staleness-aware decay selection and auto-reset.

Let $\theta^{(t)}$ be the actor parameters after update step t, and let $\beta_{\text{prox}} \in (0, 1)$ be the EWMA decay. We maintain a cumulative normalization weight $w^{(t)}$ and proximal parameters $\theta^{(t)}_{\text{prox}}$ as

$$w^{(t)} = 1 + \beta_{\text{prox}} w^{(t-1)}, \qquad \theta^{(t)}_{\text{prox}} = \frac{1}{w^{(t)}}\theta^{(t)} + \beta_{\text{prox}} \frac{w^{(t-1)}}{w^{(t)}} \theta^{(t-1)}_{\text{prox}}. \tag{22}$$

Unrolling the recursion gives the closed-form expression used in the main text:

$$\theta^{(t)}_{\text{prox}} = \frac{\sum_{k=0}^{t} \beta_{\text{prox}}^{t-k} \theta^{(k)}}{\sum_{k=0}^{t} \beta_{\text{prox}}^{t-k}}. \tag{23}$$

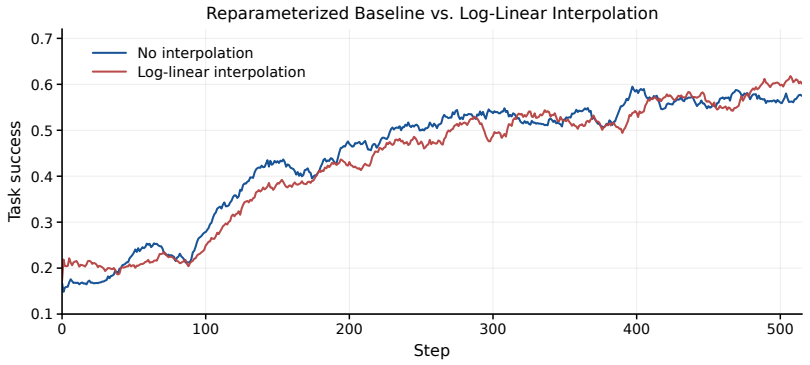


Figure 5: Training curves for the reparameterized no-interpolation baseline and the log-linear interpolation proxy. The two curves are nearly indistinguishable over training, consistent with the derivation that log-linear interpolation mainly changes the ratio parameterization.

Thus, $\theta_{\text{prox}}^{(t)}$ is a normalized exponentially weighted average of historical actor states. A larger β_{prox} keeps a longer memory of previous policies, while a smaller value places more weight on the current actor.

The decay factor also determines the effective look-back depth of the reference. In the stationary limit, the center of mass of the EWMA history is

$$\text{COM}_{\text{prox}} = \frac{\sum_{k=0}^{\infty} k\beta_{\text{prox}}^{k}}{\sum_{k=0}^{\infty} \beta_{\text{prox}}^{k}} = \frac{\beta_{\text{prox}}}{1-\beta_{\text{prox}}}. \tag{24}$$

We align the EWMA mean age with the observed loss-token-weighted mean staleness:

$$s_i = v_{\text{actor}} - v_{\text{behavior},i}, \qquad \hat{s} = \frac{\sum_i m_i s_i}{\sum_i m_i}, \qquad \beta_{\text{prox}} = \frac{\hat{s}}{1+\hat{s}}, \tag{25}$$

where m_i is the loss mask and versions are measured in actor-update units. Only under a uniform age distribution over $[0, W_{\max}]$ does this reduce to $\hat{s} \approx W_{\max}/2$ and $\beta_{\text{prox}} \approx W_{\max}/(W_{\max}+2)$. In our runs, valid tokens concentrate near age three, so $\hat{s} \approx 3$ and $\beta_{\text{prox}} = 0.75$.

C Full PPO Variant Comparison

Table 7 provides the expanded comparison behind Table 1. We distinguish two ratios whenever possible: r_1 is the policy-update or staleness ratio used by PPO-style clipping, while r_2 is the training–inference discrepancy or proxy-reference ratio. For positive advantages, PPO clipping keeps samples before the upper clipping boundary; for negative advantages, it keeps samples after the lower clipping boundary. The discrepancy constraint, when used, is symmetric around one and does not depend on the advantage sign.

The standard PPO-clip objective uses a single ratio and therefore cannot separate policy staleness from training–inference discrepancy. The train_infer variant replaces the denominator with the rollout-side behavior distribution, but still applies PPO clipping to a mixed ratio. Decoupled PPO restores the intended two-factor structure when the training-side old policy π_{old} is available. In asynchronous training, however, this reference may be missing; using π_{async} or a constructed π_{prox} then gives a practical proxy rather than an exact decomposition. This is why proxy methods can stabilize optimization while still leaving the semantic old-logit mismatch unresolved.

D Additional Threshold Examples

We provide additional threshold comparisons to show that the trade-off in Section 6.4 is not specific to one pair of hyperparameters. As in the main text, the first number in each run name is the training–inference discrepancy threshold and the second number is the stale-policy threshold. The upper

Table 7: Expanded comparison of PPO variants and proxy-policy corrections. The columns A_+ and A_- show the active regions for positive and negative advantages, respectively.

Algorithm	**Ratio Formula**	A_+	A_-	**Role**
PPO-clip (standard)	$r_1 = \frac{\pi_\theta(a_t\|s_t)}{\pi_{\text{old}}(a_t\|s_t)}$	$r_1 \in (0, 1+\epsilon_1)$	$r_1 \in (1-\epsilon_1, \infty)$	$M(r_1)$
PPO-clip & train_infer	$r_1 = \frac{\pi_\theta(a_t\|s_t)}{\mu_{\text{old}}(a_t\|s_t)}$	$r_1 \in (0, 1+\epsilon_1)$	$r_1 \in (1-\epsilon_1, \infty)$	$M(r_1)$
decoupled PPO & train_infer	$r_1 = \frac{\pi_\theta(a_t\|s_t)}{\pi_{\text{old}}(a_t\|s_t)}$ $r_2 = \frac{\pi_{\text{old}}(a_t\|s_t)}{\mu_{\text{old}}(a_t\|s_t)}$	$r_1 \in (0, 1+\epsilon_1)$ $r_2 \in (1 \pm \epsilon_2)$	$r_1 \in (1-\epsilon_1, \infty)$ $r_2 \in (1 \pm \epsilon_2)$	$M(r_2) \cdot M(r_1)$
PPO-EWMA	$r_1 = \frac{\pi_\theta(a_t\|s_t)}{\pi_{\text{prox}}(a_t\|s_t)}$ $r_2 = \frac{\pi_{\text{prox}}(a_t\|s_t)}{\mu_{\text{old}}(a_t\|s_t)}$ $\theta_{\text{prox},t} = \beta\theta_{\text{prox},t-1} + (1-\beta)\theta_t$	$r_1 \in (0, 1+\epsilon_1)$	$r_1 \in (1-\epsilon_1, \infty)$	$r_2 \cdot M(r_1)$
PPO-EWMA & train_infer	$r_1 = \frac{\pi_\theta(a_t\|s_t)}{\pi_{\text{prox}}(a_t\|s_t)}$ $r_2 = \frac{\pi_{\text{prox}}(a_t\|s_t)}{\mu_{\text{old}}(a_t\|s_t)}$ $\theta_{\text{prox},t} = \beta\theta_{\text{prox},t-1} + (1-\beta)\theta_t$	$r_1 \in (0, 1+\epsilon_1)$ $r_2 \in (1 \pm \epsilon_2)$	$r_1 \in (1-\epsilon_1, \infty)$ $r_2 \in (1 \pm \epsilon_2)$	$M(r_2) \cdot r_2 \cdot M(r_1)$
linear_prox	$r_1 = \frac{\pi_\theta(a_t\|s_t)}{\pi_{\text{prox}}(a_t\|s_t)}$ $r_2 = \frac{\pi_{\text{prox}}(a_t\|s_t)}{\mu_{\text{old}}(a_t\|s_t)}$ $\pi_{\text{prox}} = \alpha\mu_{\text{old}} + (1-\alpha)\pi_\theta$	$r_1 \in (0, 1+\epsilon_1)$ $r_2 \in (1 \pm \epsilon_2)$	$r_1 \in (1-\epsilon_1, \infty)$ $r_2 \in (1 \pm \epsilon_2)$	$r_2 \cdot M(r_1)$
linear_prox & train_infer	$r_1 = \frac{\pi_\theta(a_t\|s_t)}{\pi_{\text{prox}}(a_t\|s_t)}$ $r_2 = \frac{\pi_{\text{prox}}(a_t\|s_t)}{\mu_{\text{old}}(a_t\|s_t)}$ $\pi_{\text{prox}} = \alpha\mu_{\text{old}} + (1-\alpha)\pi_\theta$	$r_1 \in (0, 1+\epsilon_1)$	$r_1 \in (1-\epsilon_1, \infty)$	$M(r_2) \cdot r_2 \cdot M(r_1)$
decoupled PPO & train_infer & Async	$r_1 = \frac{\pi_\theta(a_t\|s_t)}{\pi_{\text{async}}(a_t\|s_t)}$ $r_2 = \frac{\pi_{\text{async}}(a_t\|s_t)}{\mu_{\text{old}}(a_t\|s_t)}$ $\theta \geq \text{async} \geq \text{old}$	$r_1 \in (0, 1+\epsilon_1)$ $r_2 \in (1 \pm \epsilon_2)$	$r_1 \in (1-\epsilon_1, \infty)$ $r_2 \in (1 \pm \epsilon_2)$	$M(r_2) \cdot M(r_1)$

panel of each success figure reports the task success curve, and the lower panel reports the difference between the two curves. These differences make the early-speed and late-stability trade-off easier to inspect.

Figure 6 isolates the effect of changing the stale-policy threshold under a fixed discrepancy threshold of 1.005. The looser threshold initially retains more update signal and therefore improves early progress. Later, however, the gap narrows and the curve becomes more oscillatory, consistent with the interpretation that early under-filtering creates additional correction pressure in later training.

Figure 7 repeats the comparison with discrepancy threshold 1.003. Since this discrepancy filter is already stricter, both settings discard more biased tokens than the 1.005 pair. The remaining difference is therefore smaller, but the same qualitative pattern remains: the looser stale-policy setting improves early learning speed, while the stricter setting gives a more controlled trajectory.

Figure 8 uses discrepancy threshold 1.002, which filters the most aggressively among the appendix examples. This setting illustrates the other side of the trade-off: stronger filtering reduces the number of biased tokens and stabilizes training, but it also limits the amount of useful off-policy signal available for policy improvement. The result is slower optimization and a lower early success curve.

Figure 9 provides a second example of the coupling between discrepancy repair and PPO-CLIP. The two runs share discrepancy threshold 1.004 and differ in the stale-policy threshold. Changing the stale-policy threshold changes not only the PPO clip fraction, but also the trajectory of the training–inference mask. This confirms that the two mechanisms interact through the active-token set: a looser stale-policy constraint can leave more questionable tokens for discrepancy repair, while stronger discrepancy filtering changes how often PPO-CLIP becomes active on the remaining tokens.

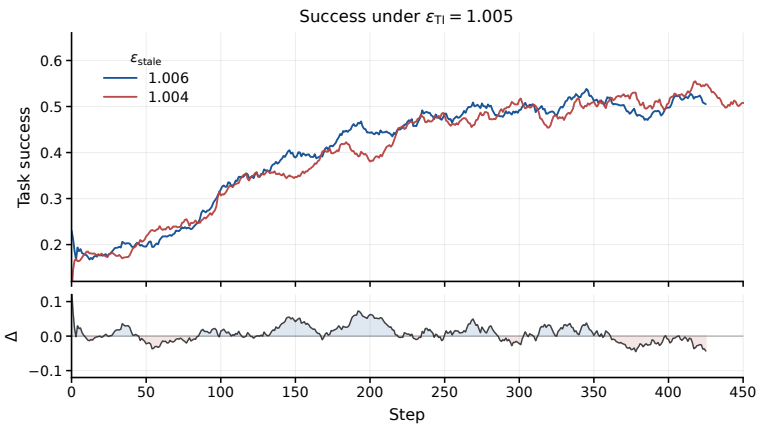


Figure 6: Additional threshold comparison: `snap1005_1006` vs. `snap1005_1004`. The two runs share the same discrepancy threshold and differ only in stale-policy control. The looser stale-policy threshold gives a faster early trajectory, while the stricter setting catches up more smoothly later.

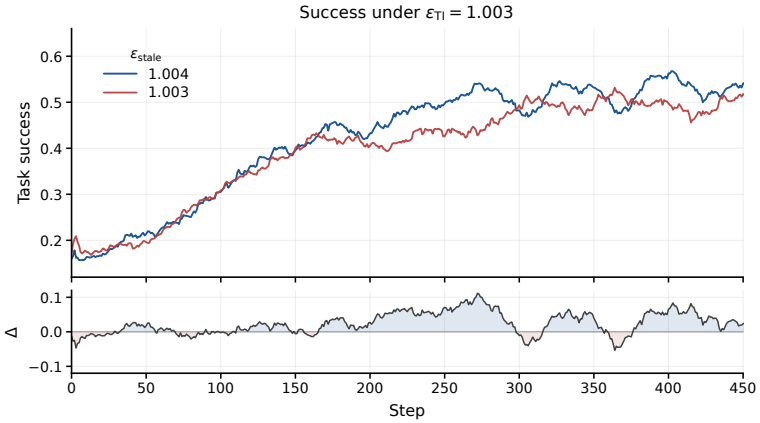


Figure 7: Additional threshold comparison: `snap1003_1004` vs. `snap1003_1003`. The same early-speed and late-stability trade-off appears under a stricter discrepancy threshold.

Thus, exact old-logit recovery restores the correct semantic decomposition, but threshold tuning still has to account for the joint behavior of the two constraints.

E Additional PPO-EWMA Ablations

This appendix provides the detailed PPO-EWMA ablations behind Section 6.5. We use three diagnostic quantities: Task success, Train-Infer Mask, and PPO-CLIP ratio. Task success measures optimization progress, while Train-Infer Mask and PPO-CLIP ratio show whether the EWMA reference remains usable as a proximal policy.

Decay-factor behavior. Figure 10 compares $\beta = 0.9$, $\beta = 0.75$, and $\beta = 0.5$. When $\beta = 0.9$, the EWMA reference has a long memory and remains strongly affected by early actor versions. As training proceeds, π_{prox} becomes increasingly misaligned with recent rollout policies, so the discrepancy ratio $\pi_{\text{prox}}/\mu_{\text{old}}$ drifts away from one. This makes the Train-Infer Mask progressively more aggressive. Tightening either the discrepancy threshold or the PPO-CLIP ratio threshold does not remove this failure mode: both $\beta = 0.9$ runs eventually reach an almost zero Train-Infer Mask value. This supports our interpretation that the issue is not only threshold choice, but also reference-policy lag.

The choice $\beta_{\text{prox}} = 0.75$ gives an asymptotic EWMA mean age of three updates, matching the observed loss-token-weighted mean staleness in our runs. It therefore gives the fastest early Task

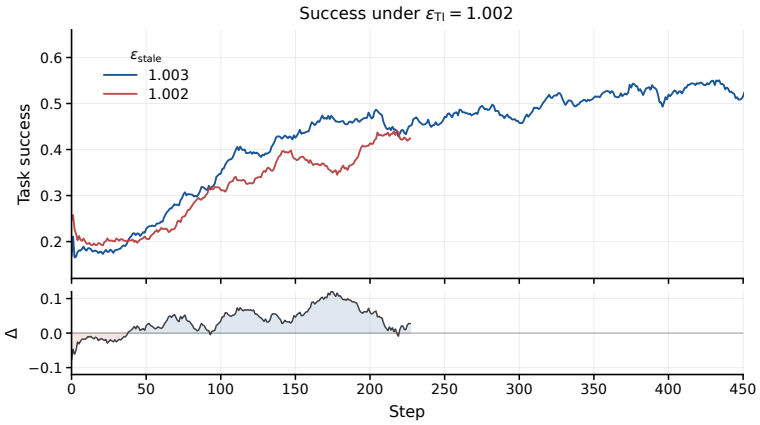


Figure 8: Additional threshold comparison: `snap1002_1003` vs. `snap1002_1002`. Under the strictest discrepancy threshold, the retained signal is smaller and learning is slower, but the training trajectory is more stable.

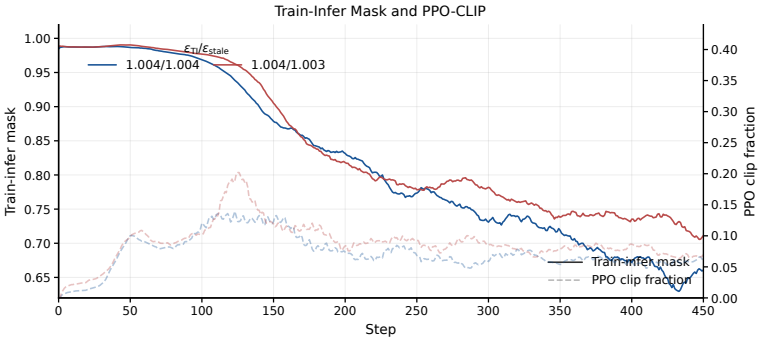


Figure 9: Additional interaction example: `snap1004_1004` vs. `snap1004_1003`. The discrepancy mask and PPO-CLIP activation change together even though the exact old logits are available.

success increase among the compared decays. However, without any reset, this run also suffers a late Train-Infer Mask collapse: the minimum Train-Infer Mask value falls below 2%, and the final Train-Infer Mask value remains only about 3%. In contrast, $\beta = 0.5$ has a shorter memory and keeps many more tokens active, but its Task success curve is less efficient early. These results show the core trade-off of PPO-EWMA: a longer memory provides a stronger stabilizing anchor, but can accumulate stale-policy bias; a shorter memory adapts faster, but weakens the proximal reference.

Figure 11 shows that threshold choice remains important even when the EWMA decay is small enough to avoid the severe collapse observed for $\beta = 0.9$. For $\beta = 0.4$ and $\beta = 0.5$, looser Train-Infer Mask or PPO-CLIP ratio thresholds retain more update signal early and can produce faster initial gains. However, these looser settings also allow more mismatched tokens to participate in optimization, which can cause a visible mid-training success drop. The subsequent recovery is consistent with the dual-constraint interpretation: discrepancy masking removes tokens whose Train-Infer ratio becomes too unreliable, while PPO-CLIP caps the update magnitude on the remaining tokens.

Automatic reset behavior. Figure 4 shows that automatic reset resolves the late-collapse failure while preserving most of the early efficiency of $\beta = 0.75$. The reset mechanism is conservative: in the two runs with thresholds $\tau = 0.9$ and $\tau = 0.8$, it is triggered only three and two times, respectively. Nevertheless, these few resets are sufficient to clear the stale history accumulated in θ_{prox} and recover a high Train-Infer Mask value. This indicates that the most important effect of reset is early re-centering of the proxy reference; after the reference is corrected, later training usually remains in a healthier region without frequent interventions.

The reset indicator further confirms that reset is not acting as a continuously active stabilizer. Most later steps have zero consecutive low-ratio streaks and no reset trigger. Therefore, the improvement

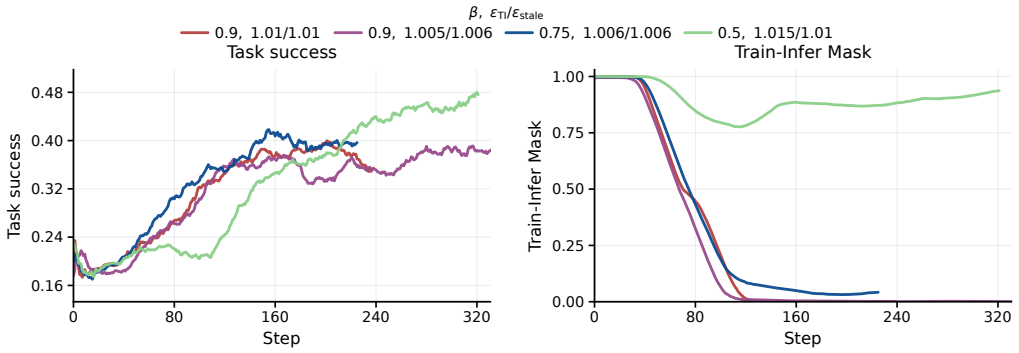


Figure 10: PPO-EWMA decay comparison. A large decay can accumulate stale reference history and collapse the Train-Infer Mask, while the staleness-aware decay gives faster early Task success.

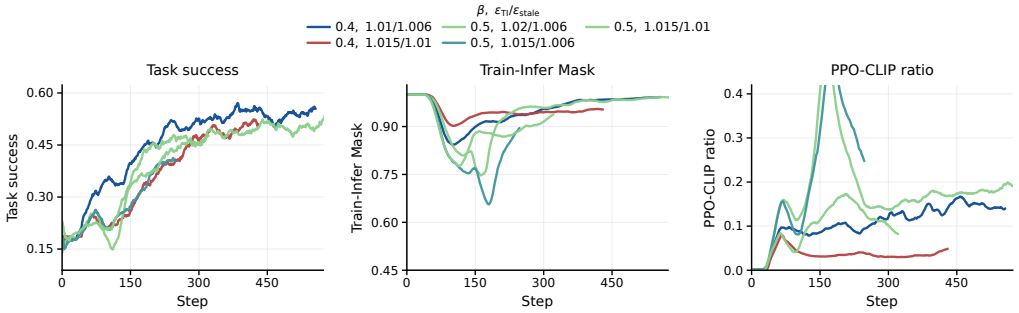


Figure 11: PPO-EWMA threshold interaction for $\beta = 0.4$ and $\beta = 0.5$. Looser Train-Infer Mask or PPO-CLIP ratio thresholds can improve early progress, but they also admit more noisy off-policy updates and may cause a mid-training success drop. The coupled mask and clip dynamics can later recover part of the trajectory by filtering or capping the problematic updates.

should be interpreted as a small number of re-centering events that remove stale EWMA history after the reference has drifted too far, rather than as frequent intervention throughout training.

F Experimental Details

Evaluation splits. For τ^2-Bench, we use all available `base` samples for the retail and airline domains. For the telecom domain, we use the `test` split.

Implementation framework. We use ROLL as the base training framework for our experiments. ROLL provides a well-structured agent abstraction.

G Additional System Measurements

G.1 Snapshot Old-Logit Overhead

Table 8 reports the detailed overhead introduced by enabling snapshot-based old-logit computation. Both runs use `old_log_probs_source=snapshot_old` with asynchronous snapshot recovery enabled. The 4B run contains 77 measured training steps before termination, while the 30B-A3B run contains 11 measured steps. These numbers should therefore be interpreted as system-cost measurements from the observed run windows rather than final training results.

The main additional cost is not snapshot saving itself, but the combination of loading or restoring historical versions and running extra forward passes to compute old logits. In the 4B run, snapshot-old computation adds roughly 95 seconds per measured step on average. In the 30B-A3B run, the same

Metric	Qwen3-4B	Qwen3-30B-A3B
Measured steps	77	11
Snapshot size (GB)	8.04	15.29
Max resident snapshots	5	5
Max resident snapshot storage (GB)	40.22	76.43
Snapshot save time / step (s)	3.95	6.07
Snapshot load time (s)	7.11	14.22
Snapshot restore time (s)	3.01	5.66
Historical forward time (s)	45.05	147.18
Total snapshot-old stage time / step (s)	95.20	178.22
Version shards / batch	1.81	1.09
Historical version shards / batch	1.74	1.00
Max staleness gap	3	3

Table 8: Detailed overhead from snapshot-based old-logit computation. The dominant cost is the additional historical-model forward pass used to reconstruct old logits. Snapshot retention also adds substantial resident storage pressure, especially for the 30B-A3B model.

stage adds roughly 178 seconds per measured step. The 30B-A3B setting also keeps up to 76.43GB of resident snapshot state when five snapshots are retained.

Both runs eventually terminate with Ray actor unavailability after the Raylet exits unexpectedly. The logs do not prove that snapshot recovery is the sole cause, but the measured resident snapshot storage and additional forward time show that exact old-logit recovery materially increases memory pressure and actor-side compute load.

H Limitations

This work has several limitations. First, our experiments are conducted on representative dense and MoE backbones, but we have not yet validated the proposed correction methods on models at the several-hundred-billion-parameter scale. At that scale, memory pressure, communication overhead, expert routing behavior, and rollout–training scheduling can become qualitatively different from the settings studied here. As a result, the empirical trade-offs observed in our current experiments may not fully capture the behavior of extremely large industrial training runs.

Second, our infrastructure analysis focuses on the main sources of overhead for exact old-logit acquisition, including snapshot retention, version switching, historical forward computation, and partial rollout interruption. However, we do not provide a fine-grained end-to-end systems study of all infrastructure components. In particular, we have not exhaustively analyzed scheduler behavior, network communication, placement-group fragmentation, memory swapping, rollout worker idleness, or failure recovery under sustained large-scale asynchronous training.

Third, the throughput measurements in this paper should be interpreted as indicative rather than comprehensive. They are sufficient to show that exact old-logit recovery introduces non-trivial system cost, but we have not performed a large-scale throughput sweep across many cluster sizes, model scales, sequence lengths, rollout lengths, and staleness windows. A more complete systems evaluation would be needed to precisely characterize the throughput frontier and to identify the best engineering design for each deployment regime.

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 abstract and introduction state the missing-old-logit problem, the analysis of decoupled correction, exact old-logit recovery, and the PPO-EWMA approximation. The scope is further supported by the unified analysis in Section 3, the recovery methods in Section 4, and the experiments in Section 6.

   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 limitations in the conclusion and in Appendix H. These include model scale, incomplete end-to-end systems coverage, and the limited breadth of throughput measurements.

   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 theoretical analysis states the relevant ratio definitions and proxy-policy assumptions in Section 3 and Appendix A. The detailed derivations for the interpolation-based proxy equivalences are provided in the appendix.

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 the benchmark suite, model backbones, main metrics, and the asynchronous setup.

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: [Yes]

   Justification: The current submission provide an anonymized public code or data release with commands sufficient to reproduce the main experiments.

   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 specifies the evaluated backbones, benchmark domains, metrics, and maximum staleness setting, and Appendix F records implementation framework information. However, the current manuscript does not yet include the full set of training and evaluation hyperparameters.

   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: We conduct multiple repeated experimental runs and report the averaged aggregate task metrics in the main tables. Error bars, confidence intervals and statistical significance tests are not included, as large-scale asynchronous RL experiments incur substantial computational cost, and our current focus lies in analyzing the trade-offs between different methods and system designs.

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: [No]

   Justification: Section 6.3 and Appendix G report several system measurements for exact old-logit acquisition. The paper does not yet provide complete compute-resource accounting for every training and evaluation run.

   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 work studies reinforcement-learning algorithms and system mechanisms for LLM agents using benchmark environments, and we are not aware of deviations from the NeurIPS Code of Ethics. The submission is prepared under the anonymous-review setting.

   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: [No]

    Justification: The paper is primarily methodological and systems-focused, and it does not currently include a dedicated broader-impact discussion. Potential impacts are indirect through more efficient and stable training of LLM agents.

    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 a new pretrained model, scraped dataset, or other high-risk asset requiring release safeguards. The work evaluates training methods and system designs on existing benchmark-style environments.

    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: [No]

Justification: The paper cites the main prior methods, frameworks, and benchmarks, but the current manuscript does not explicitly enumerate licenses and terms of use for all existing assets. This should be completed before final submission.

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 introduce or release a new dataset, model, or software asset as part of the submission. It proposes and evaluates correction methods within existing training and benchmark setups.

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 experiments or research with human subjects. The evaluations use benchmark environments and automated agent interactions.

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 work does not involve human-subject experiments, participant recruitment, or data collection from human subjects. Therefore IRB approval is not applicable.

    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: [Yes]

    Justification: Yes.

    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.