Selective Critique for Cost-Aware LLM Agents in Long-Horizon Decision Making

Heewon Park[1], **Somin Im**[2], **Minhae Kwon**[1*]

Abstract

Improving the reliability of large language model (LLM) agents in long-horizon decision-making remains a key challenge. When deployed as autonomous agents interacting with complex environments, early mistakes can propagate through trajectories and cause cascading failures. Recent approaches improve reliability by incorporating external critique or deliberation, but invoking these mechanisms at every step substantially increases token consumption and latency, limiting practical deployment. We propose SAG (Self-improving Agent with Gated critique), a cost-aware framework that formulates critique invocation as a step-wise decision problem during long-horizon interaction. SAG introduces a lightweight, training-free gating mechanism that estimates the utility of critique using action-level ambiguity signals—global entropy and local top-2 margin—computed over admissible actions. From a decision-theoretic perspective, this mechanism approximates the Value of Information (VoI) of critique, enabling the agent to selectively allocate expensive feedback only when its expected benefit justifies the cost. SAG further incorporates online bootstrapped self-improvement, allowing the actor to internalize critic-assisted behaviors and progressively reduce reliance on critique. Across three long-horizon interactive benchmarks and multiple backbone models, SAG substantially improves the performance-cost trade-off compared with both no-critique and always-on critique agents. On ALFWorld, SAG increases task success from 24.6% to 78.3% while maintaining a token budget comparable to ReAct, yielding a $3.1\times$ improvement in normalized token efficiency. Moreover, a 7B actor with a lightweight 3B critic achieves performance comparable to a 14B actor without critique, showing that selective critique can recover most of the reliability benefits of deliberation while dramatically reducing inference cost. Project page: `https://h2w1.github.io/SAG/`

1 Introduction

As decision-making agents, LLMs are increasingly deployed to interact with environments and make sequential decisions, often refining their behavior through interaction experience [1, 2, 3, 4, 5, 6, 7, 8, 9]. Despite recent advances, long-horizon interaction remains challenging: early decision errors can propagate across steps and lead to irreversible failures [10, 11, 12, 13, 14, 15]. One route to improve reliability is to post-train agents through large-scale environment rollouts, which can produce strong task-specific policies but requires substantial training compute and infrastructure. A complementary route, and the focus of this work, is to improve the decision process at deployment time by augmenting the agent with auxiliary feedback—e.g., self-reflection, external evaluation, or reward-model based scoring—to assess candidate actions and provide corrective guidance, often improving task success [16, 17, 18, 19, 20, 21]. In this paper, we refer to such auxiliary mechanisms as *critique*, and instantiate them as a *critic* module queried during interaction.

[1]Department of Electrical and Computer Engineering, Sungkyunkwan University, Republic of Korea. [2]Soongsil University, Republic of Korea. *Correspondence to: Minhae Kwon (`minhae.kwon@skku.edu`).

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

Critique, however, is computationally expensive: it typically requires additional model inference and longer contexts, causing token usage, latency, and monetary cost to increase disproportionately with interaction length [22, 23, 24, 25]. This raises a central question for practical agent deployment.

When should an LLM agent request a critique to balance reliability and token efficiency?

Figure 1 highlights this performance–cost trade-off with representative instantiations (ReAct [1] and LAC [17]). While always-on critique can improve task success, it incurs higher token usage and can reduce token efficiency, especially in long-horizon settings where critique is repeatedly invoked.

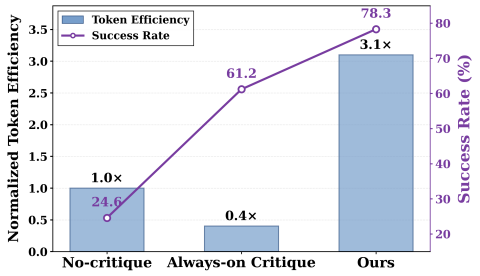


Figure 1: Normalized token efficiency and success rate. Token efficiency measures success rate per token usage and is normalized by ReAct (ReAct = 1.0). Always-on critique (LAC) improves success but can substantially reduce efficiency, whereas SAG achieves both high success and high token efficiency.

Moreover, critique is not free supervision: in realistic settings, a critic is itself an imperfect model [26, 27] whose advice may or may not improve the next action [20, 28]. More broadly, this reflects a missing design principle: many existing methods rely on fixed or heuristic critique schedules, providing limited guidance on *when* critique is warranted given its cost, even as they improve *how* critique is generated or applied.

In this paper, we cast critique invocation as a *cost-aware decision* rather than a fixed procedure, and introduce SAG (***S**elf-improving **A**gent with **G**ated critique*). We view critique invocation through a VoI lens: critique should be sought when its expected improvement outweighs its computational cost. While exact VoI is intractable, SAG uses the actor's action-level uncertainty as a lightweight proxy for critique value, combining global ambiguity (entropy) and local ambiguity (top-2 margin) computed over admissible action candidates. These signals trigger critique via a simple, bounded gating rule. Beyond selective intervention, SAG is self-improving. During online interaction, SAG stores successful trajectories and periodically fine-tunes the actor with lightweight SFT, distilling effective behaviors—including critic-assisted actions—into the policy. As the actor improves, the same VoI-motivated gate naturally reduces critique usage over time, yielding a self-evolving agent that becomes both more capable and more token-efficient. Figure 2 shows the overview of the proposed framework.

In summary, this paper makes the following contributions. 1) **SAG: cost-aware gated critique for LLM agents.** We propose SAG, a framework that formulates critic invocation as a step-wise decision problem during long-horizon interaction. By selectively allocating critique only when it is expected to improve decision quality, SAG enables agents to balance reliability with token efficiency, directly addressing the inference cost and latency bottlenecks of practical LLM agent deployment. 2) **Lightweight ambiguity-based gating.** SAG introduces a simple, training-free gating mechanism that combines global ambiguity (normalized entropy) and local ambiguity (top-2 margin) over admissible action candidates. Despite its simplicity, this mechanism provides an effective and scalable strategy for adaptive critique allocation without additional models or training. 3) **A principled VoI perspective on critique.** We interpret critique invocation through a decision-theoretic lens and show that actor-side ambiguity signals serve as practical proxies for the expected utility of critique, providing a theoretically grounded basis for cost-aware reasoning allocation in LLM agents. 4) **Critique-driven online self-improvement.** SAG leverages selectively refined trajectories as supervision signals for online bootstrapped improvement. As critic-assisted behaviors are distilled into the actor, the agent progressively improves decision quality while naturally reducing reliance on critique over time. 5) **Comprehensive empirical validation of success–cost efficiency.** Across three long-horizon interactive environments, multiple backbone models, and extensive ablations, scaling studies, and matched self-improvement controls, SAG consistently improves the success–cost Pareto frontier over both no-critique and always-on critique baselines. On ALFWorld, SAG increases success from 24.6% to 78.3% while maintaining a token budget comparable to ReAct, yielding a $3.1\times$ improvement in normalized token efficiency. SAG further shows larger gains on harder tasks and narrows the gap between smaller and larger models, with a 7B actor and 3B critic matching the performance of a 14B actor without critique.

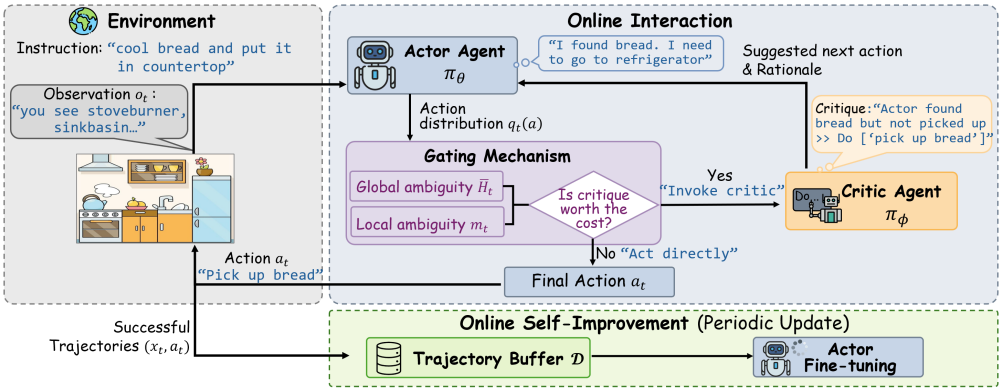


Figure 2: Overview of SAG. At each step, action-level ambiguity decides whether to query a critic; successful trajectories are periodically distilled into the actor via online SFT.

2 Related Work

LLM agents in interactive environments LLMs are increasingly used as agents that reason and act in interactive environments. Existing work improves such agents along two broad directions. The first improves an available agent during execution through prompting, deliberation, feedback, or lightweight adaptation [1, 5, 16, 29, 30, 31, 32, 33, 34]. ReAct [1] interleaves reasoning, actions, and observations for multi-step decision making, and subsequent work improves robustness with self-reflection [16, 29], debate [5, 35, 36], confidence-based aggregation [6, 37], and trajectory reuse or bootstrapping [38, 39, 40]. Inference under partial or imperfect observations has also been studied [41, 42, 43, 34, 44, 45, 46, 47, 48, 49]. The second improves the policy before deployment [50, 9] through preference optimization, environment rollouts, or post-training reinforcement learning (RL) [51, 52, 53, 54], often investing substantial task-specific compute before the agent is deployed. In this paper, we target the former deployment-time regime: given an available actor and a non-oracle critic, we ask when critique is worth its cost during interaction.

Critique and cost-aware improvement Explicit critique and feedback can correct agent behavior and support learning from experience [17, 19, 55, 39, 56, 57, 58]. Prior critic-guided methods use continuous feedback, self-asking, or trajectory-level evaluation to improve agent decisions [17, 39, 59, 19], while self-improvement methods train on successful sampled trajectories [60]. However, these approaches often rely on frequent critique, repeated sampling, or fixed schedules, incurring overhead from extra model calls, longer contexts, or multiple candidate evaluations [28, 61, 62, 63]. Although adaptive-computation methods allocate extra inference using confidence or uncertainty [64], critique in long-horizon agents is still often invoked by fixed heuristics [65, 66, 59, 67]. In contrast, our work frames critique invocation as a cost-aware step-wise decision problem, selectively querying a non-oracle critic using lightweight ambiguity signals that we later justify as proxies for the VoI.

3 Gated Critic-guided Self-improving Agent

We propose SAG, an efficient LLM agent framework that selectively invokes critique via a cost-sensitive gate and internalizes successful critic-guided behavior through online learning. Algorithm 1 summarizes SAG, and Section 4 motivates the ambiguity gate through a VoI analysis. The notations used in this paper are summarized in Appendix B.

3.1 Problem Formulation

We consider an interactive environment in which an LLM-based actor performs a sequence of language actions to solve a task. At time step t, the actor receives an observation o_t and forms a decision context x_t that contains the task instruction, the current observation, and a truncated interaction history. The environment provides an admissible action set $\mathcal{A}_t$ containing valid actions at step t.

Let π_θ denote the actor policy. The actor first produces a reasoning state r_t from x_t and scores each admissible action $a \in \mathcal{A}_t$ by its length-normalized token log-likelihood, i.e., $\ell_t(a) =$

$\frac{1}{|a|}\sum_{j=1}^{|a|}\log p_\theta(a_j \mid x_t, r_t, a_{<j})$. We define the actor-induced distribution over admissible actions as $q_t(a) = \frac{\exp(\ell_t(a))}{\sum_{a'\in\mathcal{A}_t}\exp(\ell_t(a'))}$ for $a \in \mathcal{A}_t$, and compute ambiguity over this action-level distribution.

SAG additionally introduces a binary critic-invocation decision $g_t \in \{0, 1\}$. $g_t = 1$ queries a critic before finalizing an action, while $g_t = 0$ executes the actor decision immediately. Querying the critic incurs an additional cost $c > 0$. From a decision-theoretic viewpoint, critic invocation can be interpreted as a VoI choice: critique is invoked only when its expected improvement exceeds the cost. We formalize this principle in Section 4 and in practice approximate it using ambiguity signals derived from the actor.

3.2 Ambiguity Signals from the Actor

SAG estimates when feedback from the critic is likely to be beneficial using ambiguity signals derived from the actor-induced distribution q_t. These signals capture complementary aspects of decision ambiguity.

Global ambiguity We measure global ambiguity via normalized entropy.

$$\bar{H}_t = -\frac{1}{\log|\mathcal{A}_t|}\sum_{a\in\mathcal{A}_t} q_t(a)\log q_t(a) \quad (1)$$

This is defined for $|\mathcal{A}_t| \geq 2$ and set to 0 otherwise. High entropy indicates many-way ambiguity among admissible actions.

Local ambiguity To capture ambiguity near a decision boundary, let a_1 and a_2 denote the top-1 and top-2 actions under q_t. We define the log-probability margin as

$$m_t = \log q_t(a_1) - \log q_t(a_2). \quad (2)$$

A small margin indicates that two competing actions are nearly indistinguishable, making the decision sensitive to additional information. When $|\mathcal{A}_t| < 2$, we set $m_t = \infty$, so the local ambiguity indicator is inactive.

Algorithm 1 SAG

Require: Actor policy π_θ, critic policy π_ϕ, environment $\mathcal{E}$, thresholds (τ, δ)

Observe context x_t and admissible actions $\mathcal{A}_t$
Generate reasoning state r_t from x_t
Compute actor distribution $q_t(a)$ over $\mathcal{A}_t$
Compute actor ambiguity
$\bar{H}_t \leftarrow$ normalized entropy of q_t ▷ global ambiguity
$m_t \leftarrow$ log-prob margin of q_t ▷ local ambiguity
Gated critic decision
$S_t \leftarrow \bar{H}_t + \mathbb{I}[m_t < \delta]$
$g_t \leftarrow \mathbb{I}[S_t > \tau]$ ▷ critic gate
if $g_t = 1$ **then**
    Query critic π_ϕ with $(x_t, r_t, \mathcal{A}_t)$
    $a_t \leftarrow$ critic-suggested action
else
    $a_t \leftarrow \arg\max_{a\in\mathcal{A}_t} q_t(a)$
end if
Execute a_t in environment
Online self-improvement (periodic)
if episode succeeds **then**
    Store successful trajectories in buffer $\mathcal{D}$
end if
if update time **then**
    Update actor via SFT on $\mathcal{D}$
end if

3.3 Gated Critic Invocation

SAG combines global and local ambiguity into a bounded gating score with margin threshold δ.

$$S_t = \bar{H}_t + \mathbb{I}[m_t < \delta], \quad S_t \in [0, 2] \quad (3)$$

We use a binary margin indicator rather than a continuous function to detect near-boundary decisions, since the goal of the margin term is not fine-grained calibration but robust identification of high-risk local ambiguities where small additional information can flip the decision. The critic is invoked when

$$g_t = \mathbb{I}[S_t > \tau], \quad (4)$$

where $\tau \in (0, 2)$ is a cost-calibrated threshold. Intuitively, τ is chosen as a monotone function of the critic cost c: higher costs result in more conservative critic usage.

When $g_t = 1$, SAG queries a non-oracle critic π_ϕ with the decision context, reasoning state, and admissible actions. The critic has no access to ground-truth actions or environment rewards; it returns a diagnostic rationale and a suggested action, which may itself be incorrect. If the critic output cannot be parsed as an admissible action, SAG falls back to the actor's top-1 action under q_t. In our main experiments, the actor and critic use the same backbone with different role prompts, so gains are not attributed to a stronger critic model but to selective use of critique.

3.4 Online Self-Improvement via Bootstrapped Supervision

SAG continuously improves the actor by internalizing successful behaviors observed during online interaction. Episodes are processed sequentially; after a successful episode, we collect context–action pairs (x_t, a_t) into a dataset $\mathcal{D}$, where a_t is the executed admissible action, either selected by the actor or suggested by the critic when critique is invoked. The critic is kept fixed, and every K successful episodes, only the actor is periodically updated using action-level SFT: $\mathcal{L}_{\mathrm{SFT}}(\theta) = \mathbb{E}_{(x,a)\sim\mathcal{D}}\big[-\log \pi_\theta(a \mid x)\big]$. The effect of K is analyzed in Appendix H.10. Since the target is action-only, SAG does not train on mismatched actor reasoning traces paired with critic-corrected actions.

Following prior bootstrapping-based self-training [68], we perform each update by fine-tuning from the same base model whenever new trajectories are accumulated, rather than continuing from the previous updated actor. This avoids compounding errors and preserves distributional stability. This resembles rejection fine-tuning in using successful trajectories, but SAG collects them online under a gated critique policy, allowing critic-assisted corrections to become action-level supervision while explicitly controlling critique cost.

4 Theoretical Analysis

We provide a decision-theoretic justification for SAG. Our goal is to show that (i) critic invocation admits a principled VoI interpretation, and (ii) the actor-side ambiguity signals used in SAG serve as lightweight proxies for VoI.

4.1 Critic as a VoI Decision

At each step t, let $z_t = (x_t, r_t)$ denote the actor's current decision state after reasoning. SAG chooses $g_t \in \{0, 1\}$. Invoking the critic ($g_t = 1$) incurs an additional cost $c > 0$. Let Δ_t denote the step-level improvement attributable to invoking critique under decision state z_t (e.g., an increase in success probability). We define the VoI of critique as

$$\mathrm{VoI}(z_t) := \mathbb{E}[\Delta_t \mid z_t]. \tag{5}$$

Lemma 4.1 (VoI thresholding for critic invocation)**.** *For the binary decision of whether to invoke the critic at step t, the Bayes-optimal rule is*

$$g_t^\star \;=\; \mathbb{I}[\mathrm{VoI}(z_t) > c] \;=\; \mathbb{I}[\mathbb{E}[\Delta_t \mid z_t] > c]\,. \tag{6}$$

Proof for Lemma 4.1 is provided in Appendix C. In practice, $\mathrm{VoI}(z_t)$ is intractable to compute, which motivates using lightweight proxy signals derived from the actor.

4.2 Ambiguity as a Proxy for VoI

We next describe why the actor-side ambiguity signals in SAG provide lightweight proxies for the VoI term $\mathbb{E}[\Delta_t \mid z_t]$ in Eq. (6). The key observation is that critique is valuable mainly when the actor is likely to be wrong. In particular, if critique corrects the actor with probability ρ when the actor is wrong, and a successful correction yields average improvement γ, then the expected step-level gain can be approximated by

$$\mathbb{E}[\Delta_t \mid z_t] \approx \rho\,\gamma\,\Pr(\mathrm{err}_t \mid z_t), \tag{7}$$

where err_t denotes the event that the actor's immediate decision at decision state z_t is incorrect. Combined with Lemma 4.1, this approximation implies that critique should be invoked at high-risk steps. In particular, the VoI condition $\mathbb{E}[\Delta_t \mid z_t] > c$ is equivalent to $\Pr(\mathrm{err}_t \mid z_t) > \frac{c}{\rho\gamma}$, where the same critic cost c is expressed in the error-risk space by scaling with the correction reliability ρ and average gain γ. Since $\Pr(\mathrm{err}_t \mid z_t)$ is not directly observable, we next relate it to ambiguity properties of q_t.

Global ambiguity Let $a_1 = \arg\max_{a\in\mathcal{A}_t} q_t(a)$ and $q_{\max} = q_t(a_1)$. We use normalized entropy $\bar{H}_t$ (Eq. 1) to capture global ambiguity across candidate actions.

Proposition 4.2 (Entropy controls greedy error risk)**.** *Interpreting q_t as the actor's belief over which admissible action is correct at decision state z_t, the step-level error risk satisfies*

$$\Pr(\mathrm{err}_t \mid z_t) \;=\; 1 - q_{\max}. \tag{8}$$

Moreover, with entropy $H(q_t) = -\sum_{a\in\mathcal{A}_t} q_t(a)\log q_t(a)$, we have $q_{\max} \geq \exp(-H(q_t))$. Thus, we get

$$\Pr(\mathrm{err}_t \mid z_t) \leq 1 - \exp\left(-H(q_t)\right) = 1 - |\mathcal{A}_t|^{-\bar{H}_t}. \tag{9}$$

Proof for Proposition 4.2 is provided in Appendix D. Proposition 4.2 formalizes that higher entropy $\bar{H}_t$ rules out the presence of a single dominant action, necessarily increasing an upper bound on the error risk of greedy execution. This corresponds to a regime in which probability mass is spread across many admissible actions, rather than concentrated on a clear best choice. We refer to this regime as global ambiguity, since ambiguity is distributed broadly over the action space. In such decision states, additional information from critique has greater potential value, motivating the use of entropy as a proxy for global decision risk.

Local ambiguity Global ambiguity can under-penalize near-ties between the top two actions (e.g., when q_t concentrates on two candidates). We therefore also use the margin m_t (Eq. 2) to capture *near-boundary* uncertainty.

Proposition 4.3 (Margin controls top-2 error risk)**.** *Assume the correct action lies in* $\{a_1, a_2\}$*. Then selecting* a_1 *has a conditional error probability as follows.*

$$\Pr(\text{err} \mid z_t, \text{top-2}) = \frac{q_t(a_2)}{q_t(a_1) + q_t(a_2)} = \frac{1}{1 + e^{m_t}} \leq e^{-m_t} \tag{10}$$

Proof for Proposition 4.3 is provided in Appendix E. Proposition 4.3 formalizes that small margins correspond to large error risk, making the decision sensitive to small amounts of additional information. Such near-boundary decisions are precisely where critique is most likely to change the outcome, motivating the use of the margin as a proxy for local decision risk.

Therefore, decision states with higher error risk admit larger VoI. Proposition 4.2 and Proposition 4.3 show that (i) high entropy rules out a dominant action (broad ambiguity), and (ii) small margin implies large local error risk. This motivates using $\bar{H}_t$ and m_t as lightweight, computable proxies for VoI in the gating score (Eqs. (3) and (4)).

5 Experimental Setting

In this section, we describe simulation settings for Section 6. Additional simulation details are provided in Appendix G.

Interactive decision-making benchmarks We evaluate SAG on three complementary long-horizon benchmarks: **ALFWorld** [69], which requires grounded action under partial observability; **BabyAI-Text** [70], which tests compositional instruction following and planning; and **WebShop** [71], which involves noisy open-domain search and exploration. Together, they cover state tracking, multi-step reasoning, and cost-aware decision making. Details are in Appendix G.1.

Backbone Models and Training We evaluate all methods using three instruction-tuned LLM backbones: Qwen2.5-7B-Instruct [72], Llama-3.1-8B-Instruct [73], and Mistral-7B-Instruct [74]. For SAG, the actor and critic use the same backbone and model size unless otherwise stated, differing only in their role prompts. Actor–critic capacity scaling is analyzed with Qwen2.5-Instruct 3B/7B/14B pairings and extended to 32B in Appendix H.1. SAG performs online self-improvement via action-level SFT: context–action pairs from successful episodes are collected, and the actor is updated whenever 10 successful trajectories are accumulated. Unless otherwise stated, all baselines share the same backbone models and inference settings to ensure fair comparison. The online self-improvement protocol and hyperparameters are provided in Appendix G.2 and G.3.

Baselines and comparison methods We compare SAG against representative LLM agent frameworks, grouped into three categories. We focus on inference-time and lightweight-adaptation baselines; the scope relative to RL-based agent post-training is discussed in Appendix G.5.

- **Basic single-agent baseline** (ReAct [1]) which interleaves reasoning and environment actions, serving as a lightweight baseline that does not invoke a critic during execution.
- **Deliberation methods** (Reflexion [16], Debate [5], CER [6], Traj-Boost [38]) which improve robustness by increasing computation through multi-pass deliberation or experience reuse (e.g., self-reflection across trials, multi-agent discussion, confidence-based selection/aggregation, or trajectory reuse). However, they do not model critique invocation as a cost-sensitive, step-wise decision during execution.
- **Always-on critic methods** (Retrospex [19], LAC [17]) which utilize a critic throughout execution by re-scoring or querying at essentially every step. While effective for success, their persistent critic usage can substantially increase inference cost and reduce token efficiency.

Table 1: Performance comparison on three benchmarks with three backbones. **Succ**: success rate (%). **#Horizon**: average steps per successful episode. **#Token**: average tokens per episode (k). **Eff**: normalized token efficiency, computed as Succ/#Token and normalized so that ReAct = 1.0 for each backbone and benchmark.

Model	Type	Method	ALFWorld				BabyAI				WebShop			
			Succ↑	#Horizon↓	#Token↓	Eff↑	Succ↑	#Horizon↓	#Token↓	Eff↑	Succ↑	#Horizon↓	#Token↓	Eff↑
GPT-4.1	Basic	ReAct	80.6	11.9	32	1.00	74.0	15.2	27	1.00	33.5	7.9	10	1.00
Qwen2.5-7B-Instruct	Basic	ReAct	24.6	15.8	**55**	1.00	36.0	7.4	55	1.00	11.0	6.3	12	1.00
	Deliberation	Reflexion	40.3	16.3	150	0.60	41.0	**7.0**	162	0.39	11.5	6.8	15	0.81
	Deliberation	Debate	31.3	14.3	389	0.18	52.0	8.9	441	0.18	20.0	**4.7**	90	0.24
	Deliberation	CER	60.5	16.4	210	0.64	45.0	8.0	239	0.29	16.0	5.1	11	1.50
	Deliberation	Traj-Boost	56.0	12.9	272	0.46	44.0	10.0	85	0.80	24.5	5.1	12	2.19
	Critic	Retrospex	70.2	**10.5**	94	1.67	42.0	9.5	**50**	1.30	20.0	5.9	**6**	3.58
	Critic	LAC	61.2	14.2	339	0.40	47.0	10.5	436	0.17	25.0	6.5	82	0.32
	Critic	**Ours (SAG)**	**78.3**	11.4	56	**3.10**	**70.0**	8.1	78	**1.38**	**37.0**	5.8	9	**4.55**
Llama-3.1-8B-Instruct	Basic	ReAct	51.5	15.1	**45**	1.00	32.0	**6.6**	**55**	1.00	26.5	7.1	9	1.00
	Deliberation	Reflexion	61.2	15.7	106	0.51	41.0	8.0	153	0.46	25.0	7.4	13	0.66
	Deliberation	Debate	17.9	**10.1**	569	0.03	35.0	8.8	522	0.12	18.5	6.7	90	0.07
	Deliberation	CER	34.3	15.2	237	0.13	38.0	8.1	255	0.26	4.0	6.3	44	0.03
	Deliberation	Traj-Boost	59.7	15.8	279	0.19	43.0	7.6	80	0.93	11.0	6.0	10	0.38
	Critic	Retrospex	22.4	15.3	121	0.16	11.0	6.8	67	0.28	13.0	7.6	9	0.49
	Critic	LAC	44.0	17.0	381	0.10	**60.0**	10.8	281	0.37	22.0	6.8	87	0.09
	Critic	**Ours (SAG)**	**63.0**	11.0	51	**1.08**	**60.0**	8.0	69	**1.49**	**36.5**	**5.2**	**6**	**2.08**
Mistral-7B-Instruct	Basic	ReAct	30.6	15.4	**56**	1.00	38.0	7.8	**54**	1.00	30.5	7.3	**11**	1.00
	Deliberation	Reflexion	46.3	15.9	145	0.58	36.0	6.7	169	0.30	24.0	7.4	17	0.51
	Deliberation	Debate	20.2	12.9	655	0.06	30.0	9.1	542	0.08	12.0	6.9	125	0.03
	Deliberation	CER	43.3	13.5	247	0.32	47.0	**6.3**	222	0.30	8.0	6.2	18	0.16
	Deliberation	Traj-Boost	30.6	16.2	334	0.17	42.0	7.3	83	0.72	13.5	5.7	16	0.31
	Critic	Retrospex	20.2	11.9	146	0.25	31.0	8.1	58	0.75	7.0	9.3	14	0.28
	Critic	LAC	41.8	17.5	546	0.14	47.0	11.1	293	0.23	19.5	7.1	75	0.09
	Critic	**Ours (SAG)**	**53.2**	**11.7**	61	**1.61**	**48.0**	7.9	58	**1.18**	**38.0**	**5.6**	**11**	**1.26**

- **High-capacity reference** (ReAct using GPT-4.1 [75]) which provides a reference point for how a much stronger model performs without critique.

Evaluation protocol and metrics We evaluate all methods on standard test sets (ALFWorld: 134; BabyAI-Text: 100; WebShop: 200 instances) using a sequential online-evaluation protocol. For SAG, each episode is evaluated once; successful past trajectories may update the actor before later episodes, but no episode is re-evaluated after being used for self-improvement. We report success rate, average steps per successful episode, average tokens per episode, and normalized token efficiency, defined as success rate per token usage and normalized by ReAct for each benchmark and backbone. All methods use pass@1, except Reflexion in its native pass@3 setting. Detailed analyses and ablations in Section 6 are conducted on ALFWorld with Qwen2.5-7B-Instruct, which serves as our primary setting.

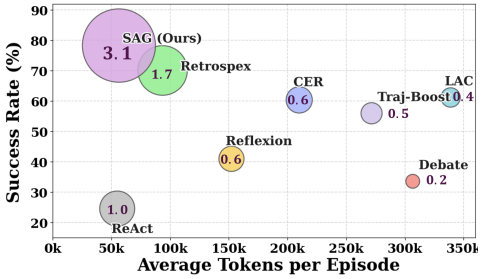


Figure 3: Performance–cost comparison. Bubble area indicates normalized token efficiency (ReAct = 1.0).

6 Experimental Results

6.1 Selective critique achieves strong success at near-ReAct cost

Table 1 summarizes the performance across methods. Overall, deliberation and always-on critique tend to improve success at the cost of substantially increased token usage, whereas SAG achieves a more favorable balance by selectively invoking critique only when needed.

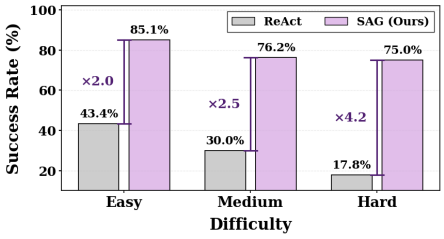
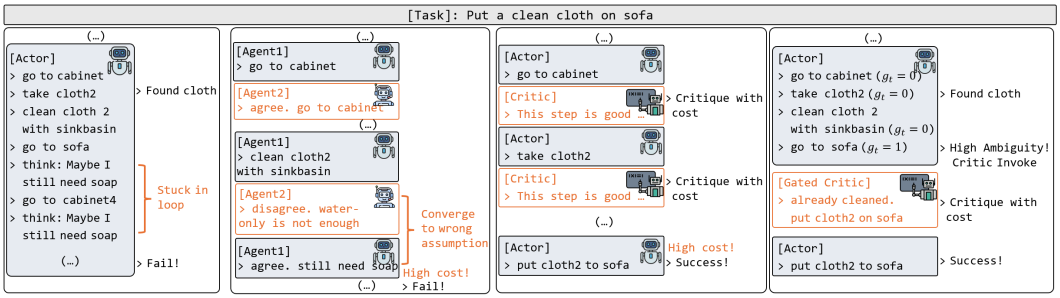


Figure 4: Success rate over task level. Task difficulty is stratified by the trajectory length of a GPT-4.1 ReAct agent.

SAG improves long-horizon success while preserving a ReAct-like token budget Across all backbone–benchmark settings, SAG improves success over ReAct in every case, with an average relative improvement of 91% while keeping token usage close to ReAct on ALFWorld/WebShop and far below always-on critique on BabyAI-Text. This yields higher normalized token efficiency, especially on longer-horizon environments such as ALFWorld. Notably, in several settings,



(a) ReAct (b) Debate (c) LAC (d) SAG (Ours)

Figure 5: Case Study: SAG effectively corrects the trajectory, avoiding both failure and pervasive critique overhead.

SAG with a 7B backbone attains success rates close to those of a stronger GPT-4.1-based ReAct agent, narrowing the gap through improved decision quality rather than increased model capacity.

Always-on critique and deliberation tend to be token-inefficient Methods that invoke critique or deliberation at most steps quickly become cost-dominated, increasing token usage by $3\times$ to over $10\times$ relative to ReAct while yielding only modest success gains. As shown in Figure 3, they shift rightward on the performance–cost plane, reflecting higher cost without proportional reliability gains. Qualitative analysis (Figure 5) shows that always-on critique repeatedly intervenes at low-risk states, accumulating redundant confirmations and growing context length, whereas SAG corrects a few high-risk decisions while avoiding unnecessary feedback.

SAG exhibits increasing advantages on harder episodes Figure 4 analyzes performance as a function of episode difficulty. We stratify test episodes into easy, medium, and hard subsets based on the trajectory length of a GPT-4.1 ReAct agent, with longer trajectories indicating higher difficulty. We compare SAG and ReAct on the same subsets. SAG consistently outperforms ReAct across all difficulty levels, with the performance gap widening as difficulty increases. On easy episodes, SAG improves success from 43.4% to 85.1% (approximately $2\times$ relative improvement), whereas on hard episodes it improves success from 17.8% to 75.0%, corresponding to over a $4\times$ improvement. This trend indicates that selective critique is particularly effective in challenging long-horizon scenarios, where early mistakes are more likely to propagate and dominate outcomes.

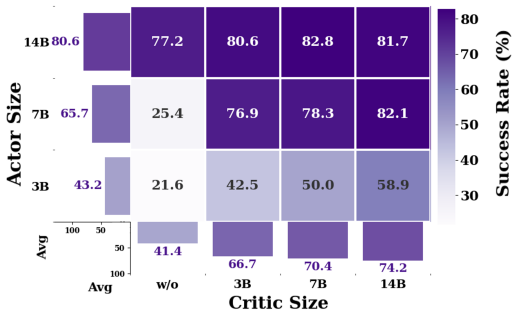


Figure 6: Actor–critic capacity scaling for SAG. Each cell reports success rate for a given actor–critic size pairing. Marginal bars show mean performance across actor and critic sizes.

6.2 Why Gated Critique Works: Capacity Scaling and Ablations

Effect of actor and critic capacity Figure 6 analyzes SAG across actor–critic capacity pairings (3/7/14B) and a no-critic setting. Actor scaling provides the largest gains: without critique, success increases from $21.6\% \rightarrow 25.4\% \rightarrow 77.2\%$ from 3B to 14B, indicating that actor capacity is the primary driver of long-horizon performance. Nevertheless, gated critique consistently improves all actor sizes. Notably, a 7B actor with a 3B critic already matches a 14B actor without critique (76.9% vs. 77.2%), showing that modest critic capacity can close much of the gap to a larger actor. These gains are strongest for smaller actors and saturate as the actor improves, suggesting that selective critique complements actor capacity rather than replacing it. Extended 32B and always-on comparisons are in Appendix H.1.

Table 2: Ablation study on ALFWorld. Global indicates using global ambiguity $\bar{H}_t$; Local indicates using local ambiguity m_t. Always removes the gate and invokes critique at every step; w/o critic disables critic invocation.

Global	Local	Variant	Succ ↑	#Horizon ↓	#Token ↓
✓	✓	**Full (SAG)**	**78.3**	11.4	56
✗	✓	w/o $\bar{H}_t$	77.6	12.1	51
✓	✗	w/o m_t	74.6	11.4	52
✗	✗	Always (no gating)	62.0	17.4	282
✗	✗	w/o critic	33.0	19.3	50

Ablation study Table 2 ablates the two ambiguity signals—global entropy $\bar{H}_t$ and local ambiguity m_t—as well as the gating mechanism in SAG. Removing either signal leads to a small but consistent drop in success, from 78.3% to 77.6% without $\bar{H}_t$ and to 74.6% without m_t, indicating that global and local ambiguity provide complementary cues for identifying high-risk states. In contrast, removing the gate and invoking critique at every step reduces success to 62.0% while increasing token usage to 282k

tokens, showing that indiscriminate critique can be both costly and harmful. Finally, disabling critic invocation lowers success to 33.0%, confirming that SAG's gains come from selectively deploying critique at ambiguous steps rather than from self-improvement alone.

Selective critique is not explained by self-improvement alone Table 3 compares SAG with matched online-SFT controls. ReAct + online SFT uses the same update protocol without critique, but improves success only to 30.2%. LAC + online SFT uses always-on critique and reaches 74.2%, but requires 341k tokens. SAG achieves higher success (78.3%) with only 56k tokens, showing that the gain comes from selectively deploying critique rather than online SFT alone. Full results on all benchmarks are in Appendix H.3.

Table 3: Matched online-SFT controls. All methods use the same online update protocol.

Method	Critique	Succ. ↑	#Token ↓
ReAct + online SFT	None	30.2	57
LAC + online SFT	Always-on	74.2	341
SAG	Gated	**78.3**	**56**

Parameter sensitivity Figure 7 shows that SAG is robust to the gate thresholds. Increasing τ smoothly trades success for lower token cost: over $\tau \in [0.30, 0.45]$, success changes mildly while token cost drops from 62k to 52k. Varying δ has limited impact on success (77.6%–78.3%), though larger values invoke critique more often. Overall, SAG does not require fine-grained threshold tuning. Importantly, the same threshold values are used across all benchmarks and backbone models in Table 1 without task-specific tuning. Detailed threshold sweeps are reported in Appendix H.5.

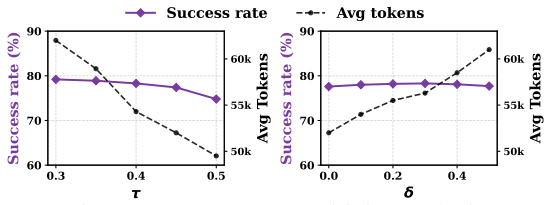
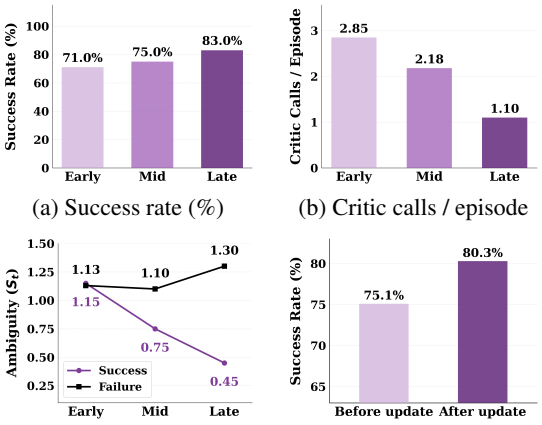


Figure 7: Parameter sensitivity analysis

6.3 Learning Dynamics of Self-Improving Agents

Online self-improvement reduces reliance on critique Figure 8a and Figure 8b show that, as interaction proceeds, SAG's success rate increases (71.0% → 75.0% → 83.0%) while critic calls decrease (2.85 → 2.18 → 1.10). This suggests that successful critic-assisted trajectories are distilled into the actor, making it more confident under the same gate and less reliant on critique.

Learning reduces decision risk rather than increasing computation Figure 8c shows that ambiguity remains well separated between successes and failures as learning progresses. For successful episodes, ambiguity consistently decreases (1.15 → 0.75 → 0.45), whereas failures exhibit persistently higher and increasing ambiguity (1.13 → 1.10 → 1.30), indicating that remaining errors concentrate on harder decision points.



(a) Success rate (%) (b) Critic calls / episode

(c) S_t for success vs. failure (d) Seen: before vs. after

Figure 8: Learning dynamics of SAG under online self-improvement. Metrics are aggregated into Early/Mid/Late phases over 134 evaluation episodes.

As a diagnostic analysis excluded from the main success metric, re-evaluating previously encountered episodes after online updates shows improved performance (Figure 8d), increasing from 75.1% to 80.3%. Together, these results show that critique acts as a transient scaffold: as learning progresses, SAG internalizes effective behaviors and shifts decision-making toward lower-risk states without increasing inference-time computation.

7 Conclusion

We introduced SAG, a cost-aware framework that selectively invokes critique to improve the reliability of LLM agents in long-horizon decision-making. SAG formulates critique invocation as a step-wise decision problem and employs a lightweight, training-free gate based on action-level ambiguity. Across multiple long-horizon benchmarks and backbone models, SAG consistently achieves a superior performance–cost trade-off compared to both no-critique and always-on critique agents. These results suggest that selective critique provides a practical and scalable pathway for improving the reliability and efficiency of LLM agents in complex sequential environments.

Acknowledgments

This work was partly supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant (No. RS-2026-25519475), the National Research Foundation of Korea (NRF) grants (RS-2023-00278812, RS-2025-02214082, RS-2026-25522801) funded by the Korean government (MSIT), and Korea Institute of Police Technology (KIPoT) funded by the Korean National Police Agency & Korea Customs Service (RS-2026-25536784). The computing resources were partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP, AI Computing Support Project for R&D) grant funded by the Korea government(MSIT) (RS-2026-25505492)

References

[1] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2022.

[2] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, 2022.

[3] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, 2023.

[4] Zhihan Liu, Hao Hu, Shenao Zhang, Hongyi Guo, Shuqi Ke, Boyi Liu, and Zhaoran Wang. Reason for future, act for now: A principled framework for autonomous LLM agents with provable sample efficiency. In *International Conference on Machine Learning*, 2023.

[5] Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *International Conference on Machine Learning*, 2024.

[6] Ali Razghandi, Seyed Mohammad Hadi Hosseini, and Mahdieh Soleymani Baghshah. CER: Confidence enhanced reasoning in LLMs. In *Annual Meeting of the Association for Computational Linguistics*, 2025.

[7] Haotian Sun, Yuchen Zhuang, Lingkai Kong, Bo Dai, and Chao Zhang. AdaPlanner: Adaptive planning from feedback with language models. In *Advances in Neural Information Processing Systems*, 2023.

[8] Dingyang Chen, Qi Zhang, and Yinglun Zhu. Efficient sequential decision making with large language models. In *Conference on Empirical Methods in Natural Language Processing*, 2024.

[9] Kyungjin Im, Miru Kim, Chanin Eom, and Minhae Kwon. Post-hoc merging is not enough: Many-shot model merging with loss-gap balancing. In *International Conference on Machine Learning*, 2026.

[10] Yilun Hao, Yang Zhang, and Chuchu Fan. Planning anything with rigor: General-purpose zero-shot planning with LLM-based formalized programming. In *International Conference on Learning Representations*, 2025.

[11] Lutfi Eren Erdogan, Hiroki Furuta, Sehoon Kim, Nicholas Lee, Suhong Moon, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. Plan-and-Act: Improving planning of agents for long-horizon tasks. In *International Conference on Machine Learning*, 2025.

[12] Jiabao Pan, Yan Zhang, Chen Zhang, Zuozhu Liu, Hongwei Wang, and Haizhou Li. DynaThink: Fast or slow? a dynamic decision-making framework for large language models. In *Conference on Empirical Methods in Natural Language Processing*, 2024.

[13] Yuchen Deng, Shichen Fan, Naibo Wang, Xinkui Zhao, and See-Kiong Ng. AgentPro: Enhancing LLM agents with automated process supervision. In *Conference on Empirical Methods in Natural Language Processing*, 2025.

[14] Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Why do multi-agent LLM systems fail? In *Advances in Neural Information Processing Systems*, 2025.

[15] Sangeun Park and Minhae Kwon. Multi^2: Hierarchical multi-agent decision-making with llm-based agents in interactive environments. In *International Conference on Machine Learning*, 2026.

[16] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

[17] Heng Dong, Kefei Duan, and Chongjie Zhang. Enhancing decision-making of large language models via actor-critic. In *International Conference on Machine Learning*, 2025.

[18] Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *International Conference on Learning Representations*, 2024.

[19] Yufei Xiang, Yiqun Shen, Yeqin Zhang, and Cam-Tu Nguyen. Retrospex: Language agent meets offline reinforcement learning critic. In *Conference on Empirical Methods in Natural Language Processing*, 2024.

[20] Yifei Wang, Yuyang Wu, Zeming Wei, Stefanie Jegelka, and Yisen Wang. A theoretical understanding of self-correction through in-context alignment. In *Advances in Neural Information Processing Systems*, 2024.

[21] Zi-Yi Dou, Cheng-Fu Yang, Xueqing Wu, Kai-Wei Chang, and Nanyun Peng. Re-ReST: Reflection-reinforced self-training for language agents. In *Conference on Empirical Methods in Natural Language Processing*, 2024.

[22] Alexander Havrilla, Sharath Chandra Raparthy, Christoforos Nalmpantis, Jane Dwivedi-Yu, Maksym Zhuravinskyi, Eric Hambro, and Roberta Raileanu. GLoRe: When, where, and how to improve LLM reasoning via global and local refinements. In *International Conference on Machine Learning*, 2024.

[23] Sirui Chen, Mengshi Zhao, Lei Xu, Yuying Zhao, Beier Zhu, Hanwang Zhang, Shengjie Zhao, and Chaochao Lu. DEPO: Dual-efficiency preference optimization for llm agents. In *Proceedings of the AAAI conference on artificial intelligence*, 2026.

[24] Xun Wang, Jing Xu, Franziska Boenisch, Michael Backes, Christopher A. Choquette-Choo, and Adam Dziedzic. Efficient and privacy-preserving soft prompt transfer for LLMs. In *International Conference on Machine Learning*, 2025.

[25] Ollie Liu, Deqing Fu, Dani Yogatama, and Willie Neiswanger. DeLLMa: Decision making under uncertainty with large language models. In *International Conference on Learning Representations*, 2025.

[26] Heewon Park, Miru Kim, and Minhae Kwon. Personalized federated sensing for heterogeneous environment. *IEEE Sensors Letters*, 9(4):1–4, 2025.

[27] Heewon Park, Miru Kim, and Minhae Kwon. Robust partial share federated learning algorithm against model poisoning attack. *Journal of Korean Institute of Communications and Information Sciences*, 48(11):1387–1398, 2023.

[28] Rong Bao, Donglei Yu, Kai Fan, and Minpeng Liao. Fixing distribution shifts of LLM self-critique via on-policy self-play training. In *Annual Meeting of the Association for Computational Linguistics*, 2025.

[29] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-Refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, 2023.

[30] Heewon Park, Mugon Joe, Miru Kim, Kyungjin Im, and Minhae Kwon. Fed-ADE: Adaptive learning rate for federated post-adaptation under distribution shift. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

[31] Heewon Park, Mugon Joe, Miru Kim, and Minhae Kwon. ASAP: Unsupervised post-training with label distribution shift adaptive learning rate. In *ACM International Conference on Information and Knowledge Management (CIKM)*, 2025.

[32] Kyungjin Im, Heewon Park, Miru Kim, Mugon Joe, and Minhae Kwon. Personalized federated post-training for real-time adaptation to label distribution shifts. *Journal of Korean Institute of Communications and Information Sciences*, 50(12):1811–1821, 2025.

[33] Miru Kim, Heewon Park, and Minhae Kwon. Personalized split federated learning with early exit: Pretraining and online learning against label shifts. *IEEE Internet of Things Journal*, 12(22):47069–47082, 2025.

[34] Mugon Joe, Miru Kim, and Minhae Kwon. Fine-tuning anomaly classifier for unbalanced network data. *Journal of Korean Institute of Communications and Information Sciences*, 49(7):911–922, 2024.

[35] Yexiang Liu, Jie Cao, Zekun Li, Ran He, and Tieniu Tan. Breaking mental set to improve reasoning through diverse multi-agent debate. In *International Conference on Learning Representations*, 2025.

[36] Tianyu Hu, Zhen Tan, Song Wang, Huaizhi Qu, and Tianlong Chen. Multi-agent debate for LLM judges with adaptive stability detection. In *Advances in Neural Information Processing Systems*, 2025.

[37] Amir Taubenfeld, Tom Sheffer, Eran Ofek, Amir Feder, Ariel Goldstein, Zorik Gekhman, and Gal Yona. Confidence improves self-consistency in LLMs. In *Findings of the Association for Computational Linguistics*, 2025.

[38] Vishnu Sarukkai, Zhiqiang Xie, and Kayvon Fatahalian. Self-generated in-context examples improve LLM agents for sequential decision-making tasks. In *Advances in Neural Information Processing Systems*, 2025.

[39] Ruihan Yang, Fanghua Ye, Jian Li, Siyu Yuan, Yikai Zhang, Zhaopeng Tu, Xiaolong Li, and Deqing Yang. The lighthouse of language: Enhancing LLM agents via critique-guided improvement. In *Advances in Neural Information Processing Systems*, 2025.

[40] Yifu Guo, Jiaye Lin, Huacan Wang, Yuzhen Han, Sen Hu, Ziyi Ni, Licheng Wang, and Mingguang Chen. SE-Agent: Self-evolution trajectory optimization in multi-step reasoning with LLM-based agents. In *Advances in Neural Information Processing Systems*, 2025.

[41] Ruipeng Wang, Yuxin Chen, Yukai Wang, Chang Wu, Junfeng Fang, Xiaodong Cai, Qi Gu, Hui Su, An Zhang, Xiang Wang, Xunliang Cai, and Tat-Seng Chua. AgentNoiseBench: Benchmarking robustness of tool-using LLM agents under noisy condition. In *International Conference on Machine Learning*, 2026.

[42] Miru Kim, Mugon Joe, and Minhae Kwon. OASIS: Open-world adaptive self-supervised and imbalanced-aware system. In *ACM International Conference on Information and Knowledge Management (CIKM)*, 2025.

[43] Miru Kim, Mugon Joe, and Minhae Kwon. Improving network attack classification on imbalanced real-world intrusion incident datasets. In *International Conference on Mobile Systems, Applications and Services (MobiSys)*, 2025.

[44] Sangeun Park, Chanin Eom, and Minhae Kwon. Inference on driving characteristic based on time-series partial observation of vehicle. *Journal of Korean Institute of Communications and Information Sciences*, 50(6):858–874, 2025.

[45] Sangeun Park, Chanin Eom, and Minhae Kwon. xLSTM-based noise-robust driving characteristic inference network for V2I systems. *Transactions of the Korean Society of Automotive Engineers*, 33(7):487–502, 2025.

[46] Chae Won Kim, Heewon Park, Dohyeon Kim, Yuchang Seong, Minhae Kwon, and Junil Kim. CluVar: Clustering of variants using autoencoder for inferring cancer subclones from single cell RNA sequencing data. *Briefings in Bioinformatics*, 26(6):bbaf603, 2025.

[47] Jungmin Kwon, Sujin Ahn, Hyunggon Park, and Minhae Kwon. Stochastic approximate inference of latent information in epidemic model: A data-driven approach. *Signal Processing*, 232:109919, 2025.

[48] Zhengwei Wu, Minhae Kwon, Saurabh Daptardar, Paul Schrater, and Xaq Pitkow. Rational thoughts in neural codes. *Proceedings of the National Academy of Sciences*, 117(47):29311–29320, 2020.

[49] Minhae Kwon, Saurabh Daptardar, Paul R Schrater, and Xaq Pitkow. Inverse rational control with partially observable continuous nonlinear dynamics. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[50] Jeyeon Eo, Dongsu Lee, and Minhae Kwon. Offline reinforcement learning based uav training for disaster network recovery. *Journal of Korean Institute of Communications and Information Sciences*, 49(1):1–11, 2024.

[51] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, 2023.

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

[53] Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *Advances in Neural Information Processing Systems*, 2025.

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

[55] Yaxi Lu, Shenzhi Yang, Cheng Qian, Guirong Chen, Qinyu Luo, Yesai Wu, Huadong Wang, Xin Cong, Zhong Zhang, Yankai Lin, Weiwen Liu, Yasheng Wang, Zhiyuan Liu, Fangming Liu, and Maosong Sun. Proactive agent: Shifting LLM agents from reactive responses to active assistance. In *International Conference on Learning Representations*, 2025.

[56] Tianqing Fang, Hongming Zhang, Zhisong Zhang, Kaixin Ma, Wenhao Yu, Haitao Mi, and Dong Yu. WebEvolver: Enhancing web agent self-improvement with coevolving world model. In *Conference on Empirical Methods in Natural Language Processing*, 2025.

[57] Chanin Eom, Dongsu Lee, and Minhae Kwon. Autonomous driving strategy for bottleneck traffic with prioritized experience replay. *Journal of Korean Institute of Communications and Information Sciences*, 48(6):690–703, 2023.

[58] Chanin Eom, Dongsu Lee, and Minhae Kwon. Selective imitation for efficient online reinforcement learning with pre-collected data. *ICT Express*, 10(6):1308–1314, 2024.

[59] Byoungjip Kim, Youngsoo Jang, Lajanugen Logeswaran, Geon-Hyeong Kim, Yu Jin Kim, Honglak Lee, and Moontae Lee. Prospector: Improving LLM agents with self-asking and trajectory ranking. In *Findings of the Conference on Empirical Methods in Natural Language Processing*, 2024.

[60] Li-Cheng Lan, Andrew Bai, Minhao Cheng, Ruochen Wang, Cho-Jui Hsieh, and Tianyi Zhou. Exploring expert failures improves LLM agent tuning. *arXiv preprint arXiv:2504.13145*, 2025.

[61] Yingming Wang and Pepa Atanasova. Self-critique and refinement for faithful natural language explanations. In *Conference on Empirical Methods in Natural Language Processing*, 2025.

[62] Junlin Wang, Siddhartha Jain, Dejiao Zhang, Baishakhi Ray, Varun Kumar, and Ben Athiwaratkun. Reasoning in token economies: Budget-aware evaluation of LLM reasoning strategies. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Conference on Empirical Methods in Natural Language Processing*, 2024.

[63] Eray Can Elumar, Cem Tekin, and Osman Yagan. Cost-aware LLM-based online dataset annotation. In *Advances in Neural Information Processing Systems*, 2025.

[64] Peiyuan Feng, Yichen He, Guanhua Huang, Yuan Lin, Hanchong Zhang, Yuchen Zhang, and Hang Li. AGILE: A novel reinforcement learning framework of LLM agents. In *Advances in Neural Information Processing Systems*, 2024.

[65] Xiaohe Bo, Zeyu Zhang, Quanyu Dai, Xueyang Feng, Lei Wang, Rui Li, Xu Chen, and Ji-Rong Wen. Reflective multi-agent collaboration based on large language models. In *Advances in Neural Information Processing Systems*, 2024.

[66] Yurun Yuan and Tengyang Xie. Reinforce LLM reasoning through multi-agent reflection. In *International Conference on Machine Learning*, 2025.

[67] Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Rühle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. Hybrid LLM: Cost-efficient and quality-aware query routing. In *International Conference on Learning Representations*, 2024.

[68] Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*, 2022.

[69] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021.

[70] Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. Grounding large language models in interactive environments with online reinforcement learning. In *International Conference on Machine Lear. ning*, 2023.

[71] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, 2022.

[72] Qwen Team. Qwen2.5: A party of foundation models, September 2024.

[73] AI@Meta. Llama 3 model card, 2024.

[74] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Chaplot, Diego Casas, Hugo Bressand, Gérard Lengyel, et al. Mistral 7B. *arXiv preprint arXiv:2310.06825*, 2023.

[75] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Appendix: Selective Critique for Cost-Aware LLM Agents in Long-Horizon Decision Making

Contents

A Broader Impacts and Limitations **16**

B Notation Table **16**

C Proof of Lemma 4.1 **17**

D Proof of Proposition 4.2 **17**

E Proof of Proposition 4.3 **18**

F Prompts **19**

G Simulation Settings **20**

G.1 Benchmarks for Interactive Environments 20

G.2 Online Self-Improvement Protocol 21

G.3 Hyperparameters 21

G.4 Detailed description of baselines 21

G.5 Scope Relative to RL-based Agent Post-training 22

H Extra Simulation Results **23**

H.1 Extended Model Scaling with Always-on Critique 23

H.2 Effect of Critic Capacity 24

H.3 Matched Online-SFT Controls 24

H.4 Robustness to Trajectory Ordering 25

H.5 Gate Threshold Sensitivity 25

H.6 Simulation results on BabyAI-Text 26

H.7 Simulation results on WebShop 26

H.8 Online SFT vs. One-shot SFT 27

H.9 Failure Modes and Non-oracle Critic Behavior 28

H.10 Sensitivity to Online Update Frequency 28

H.11 Qualitative Analysis of Prompt-Induced Responses 28

H.12 Action-Only versus Reasoning-Consistent Supervision 29

H.13 Empirical Validation of the Gating Score 29

I Example Trajectories **30**

J GenAI Usage Disclosure **35**

A Broader Impacts and Limitations

Broader Impacts This work contributes a practical methodology for improving the reliability and efficiency of LLM-based agents, a capability that is increasingly important as such systems are deployed in complex interactive settings. Because SAG is lightweight, training-free, and compatible with existing agent frameworks, it can be readily integrated into current LLM-agent pipelines used in both academic research and industrial applications. In research contexts, SAG provides a simple yet principled mechanism for studying cost-aware reasoning allocation and selective deliberation in long-horizon decision-making. In applied settings, the framework can reduce unnecessary token usage and latency while preserving reliability, making agent-based systems more economically and computationally sustainable. As with other general-purpose LLM agent techniques, the societal impact ultimately depends on how such systems are deployed; responsible development and application-level safeguards remain important for ensuring beneficial outcomes.

Limitations While SAG demonstrates consistent improvements across multiple environments and backbone models, several aspects remain open for further exploration. First, our current gating mechanism relies on ambiguity signals derived from the actor's action distribution; alternative uncertainty or value-estimation signals may further improve critique allocation. Second, the current study focuses on representative long-horizon interactive benchmarks, and evaluating SAG in additional real-world agent settings would provide further insight into its generality. Finally, the present work uses a fixed critic architecture; future work may explore adaptive or learned critics that further enhance the reliability–efficiency trade-off.

B Notation Table

Notation	Description
t	Time step in an interaction episode
o_t	Environment observation at step t
x_t	Actor context at step t, including task instruction, observation, and truncated history
$\mathcal{A}_t$	Set of admissible actions at step t
a_t	Action executed at step t
r_t	Actor reasoning state generated from x_t at step t
π_θ	Actor language-model policy parameterized by θ
K	Actor policy update interval
$\ell_t(a)$	Length-normalized token log-likelihood score of admissible action a under the actor
$q_t(a)$	Softmax-normalized actor-induced distribution over admissible actions
a_1, a_2	Top-1 and top-2 actions under q_t
$q_{\max}$	Maximum probability mass under q_t, i.e., $q_t(a_1)$
$g_t \in \{0, 1\}$	Binary critic-invocation decision at step t
c	Computational cost of invoking the critic
π_ϕ	Critic policy parameterized by ϕ
$\bar{H}_t$	Normalized entropy of q_t (global ambiguity)
m_t	Log-probability margin between top-1 and top-2 actions (local ambiguity)
S_t	Gating score combining global and local ambiguity
τ	Gating threshold controlling overall critic usage
δ	Margin threshold for detecting near-boundary decisions
Δ_t	Step-level improvement attributable to critique
$z_t = (x_t, r_t)$	Decision state after actor reasoning at step t
$\mathrm{VoI}(z_t)$	Value of information of invoking critique at decision state z_t
err_t	Event that the actor's immediate decision at step t is incorrect
ρ	Probability that the critic corrects an incorrect actor decision
γ	Average gain from a successful critic correction
$\mathcal{D}$	Buffer of successful context–action pairs for online self-improvement
$\mathcal{L}_{\mathrm{SFT}}$	Supervised fine-tuning loss used for online self-improvement

C Proof of Lemma 4.1

We consider the one-step decision problem at decision state $z_t = (x_t, r_t)$, where the agent chooses whether to invoke the critic. Let $g_t \in \{0, 1\}$ denote the invocation decision. The Bayes-optimal decision maximizes the conditional expected net utility given z_t. Invoking the critic ($g_t = 1$) yields a random improvement Δ_t (e.g., increase in success probability or expected return) and incurs a deterministic cost $c > 0$. If the critic is not invoked ($g_t = 0$), the improvement is defined as 0 and no additional cost is incurred.

Conditioned on z_t, the expected net utility of choosing $g_t = 1$ is

$$\mathbb{E}[\Delta_t \mid z_t] - c, \tag{11}$$

whereas the expected net utility of choosing $g_t = 0$ is 0. Therefore, invoking the critic is optimal iff the expected net gain is positive:

$$g_t^\star = \mathbb{I}\left[\mathbb{E}[\Delta_t \mid z_t] - c > 0\right] = \mathbb{I}\left[\mathbb{E}[\Delta_t \mid z_t] > c\right]. \tag{12}$$

Recalling the definition $\mathrm{VoI}(z_t) := \mathbb{E}[\Delta_t \mid z_t]$ completes the proof. □

D Proof of Proposition 4.2

Fix a step t and decision state $z_t = (x_t, r_t)$, and let q_t be the actor-induced distribution over admissible actions $\mathcal{A}_t$. Let $a_1 = \arg\max_{a \in \mathcal{A}_t} q_t(a)$ and $q_{\max} = q_t(a_1)$.

Step 1: Error probability under the belief interpretation. Equivalently, Eq. (8) holds if we define $\Pr(a \text{ is correct} \mid z_t) := q_t(a)$ over $\mathcal{A}_t$. Under this interpretation, the probability that the greedy choice a_1 is incorrect equals the probability mass assigned to all other actions:

$$\Pr(\mathrm{err}_t \mid z_t) = 1 - \Pr(a_1 \text{ is correct} \mid z_t) = 1 - q_{\max}, \tag{13}$$

which proves Eq. (8).

Step 2: $-\log q_{\max}$ lower bounds entropy. Let $H(q_t) = -\sum_{a \in \mathcal{A}_t} q_t(a) \log q_t(a)$ denote the entropy. Since $q_{\max} \geq q_t(a)$ for all $a \in \mathcal{A}_t$, we have $\log q_{\max} \geq \log q_t(a)$ and thus $-\log q_t(a) \geq -\log q_{\max}$ for all a. Multiplying by $q_t(a) \geq 0$ and summing over a yields

$$H(q_t) = \sum_{a \in \mathcal{A}_t} q_t(a)\big(-\log q_t(a)\big) \geq \sum_{a \in \mathcal{A}_t} q_t(a)\big(-\log q_{\max}\big) = -\log q_{\max}. \tag{14}$$

Exponentiating both sides gives $q_{\max} \geq e^{-H(q_t)}$.

Combining with Step 1, we obtain

$$\Pr(\mathrm{err}_t \mid z_t) = 1 - q_{\max} \leq 1 - e^{-H(q_t)}. \tag{15}$$

Finally, using the normalized entropy definition $\bar{H}_t = H(q_t)/\log|\mathcal{A}_t|$ (for $|\mathcal{A}_t| \geq 2$), we have $e^{-H(q_t)} = e^{-\bar{H}_t \log|\mathcal{A}_t|} = |\mathcal{A}_t|^{-\bar{H}_t}$, which yields

$$\Pr(\mathrm{err}_t \mid z_t) \leq 1 - |\mathcal{A}_t|^{-\bar{H}_t}. \tag{16}$$

This completes the proof. □

E Proof of Proposition 4.3

Fix a step t and decision state $z_t = (x_t, r_t)$. Let q_t be the actor-induced distribution over admissible actions $\mathcal{A}_t$. Let a_1 and a_2 denote the top-1 and top-2 actions under q_t, and write $q_1 := q_t(a_1)$ and $q_2 := q_t(a_2)$. Recall the log-probability margin $m_t := \log q_1 - \log q_2 = \log(q_1/q_2)$.

Assume the correct action lies in $\{a_1, a_2\}$. Conditioned on this event (denoted top-2), the induced conditional distribution over $\{a_1, a_2\}$ is

$$\Pr(a_i \text{ is correct} \mid z_t, \text{top-2}) = \frac{q_i}{q_1 + q_2}, \quad i \in \{1, 2\}. \tag{17}$$

Therefore,

$$\Pr(\text{err} \mid z_t, \text{top-2}) = \Pr(a_2 \text{ is correct} \mid z_t, \text{top-2}) = \frac{q_2}{q_1 + q_2}. \tag{18}$$

Rewriting in terms of the margin gives

$$\Pr(\text{err} \mid z_t, \text{top-2}) = \frac{q_2}{q_1 + q_2} = \frac{1}{1 + q_1/q_2} = \frac{1}{1 + e^{m_t}}. \tag{19}$$

Finally, since $e^{m_t} > 0$, we use $1 + e^{m_t} \geq e^{m_t}$ to obtain

$$\frac{1}{1 + e^{m_t}} \leq \frac{1}{e^{m_t}} = e^{-m_t}. \tag{20}$$

This proves Eq. (10). □

F Prompts

Table 4: Prompts used for ALFWorld

[Actor Prompt] You are an agent for household tasks.
Your goal is to complete the task described in natural language by reasoning step-by-step and producing an action.
Always output in the format:
"think: reasoning here"
"action: [chosen action]"
Actions must come from the admissible set provided. Do not invent new actions.
[Critic Prompt] You are a critic for an LLM-based agent.
Your GOAL: detect and correct hallucinations, loops, or inconsistent actions.
INPUTS:
- task: the goal
- history: what happened so far
- actions: valid moves.
INSTRUCTIONS:
1. verify objects: does the object the agent is targeting MATCH the task description?
2. verify history: did the agent already check this empty location or repeat failures?
3. decision:
- If the agent is safe and correct, output "next_action": "continue".
- If the agent is wrong, provide a specific "next_action" to fix it.
4. reasoning:
- Also provide a short explanation in "reasoning".
- The reasoning should explain why the proposed action is appropriate.
- Keep it concise, one or two sentences.

Table 5: Prompts used for BabyAI-Text

[Actor Prompt] You are an agent in the Grid world.
Your goal is to follow the natural language instruction step-by-step.
At each step, reason explicitly before acting. Use the format:
"think: reasoning here"
"action: [chosen action]"
Primitive actions include: go left, go right, go forward, pick up object, drop object, etc.
[Critic Prompt] You are a critic helping an actor agent navigate a small grid world.
Your GOAL:
- Detect when the actor is stuck, uncertain, or moving inefficiently.
- Propose a corrective plan that helps the actor make progress toward the goal.
INPUTS:
- goal: The current task the actor must complete.
- observation: The actor's current state in the grid world.
- valid actions: The list of actions the actor can execute now.
INSTRUCTIONS:
1. check progress:
- Determine whether the actor is making progress toward the goal.
- If the actor seems stuck in a loop, suggest actions that break the loop.
2. check validity:
- Only use actions from the valid actions list.
- Do not invent new actions.
3. proposed plan:
- Propose a SHORT sequence of 1 to 3 actions.
- The plan should be directly executable in the current state.
4. reasoning:
- Provide a brief reason explaining why the plan is helpful.
- Keep the reasoning concise, at most one sentence.

Table 6: Prompts used for WebShop

```
[Actor Prompt] You are a shopping assistant.
Your task is to satisfy the given shopping goal.
At each step, reason aloud and then select one action.
Use search[query] for the initial query; after search results are shown, choose
from the web actions available in the current observation, such as click[item],
click[option], navigation, or click[Buy Now].
Use format:
"think: reasoning here"
"action: [web action]"
[Critic Prompt] You are a critic for a WebShop shopping agent.
GOAL:
- Help the actor choose the best next action for completing the shopping task.
- Detect inefficient actions, loops, or choices that do not satisfy the user
instruction.
INPUTS:
- user instruction: the shopping request, including constraints such as
product type, size, flavor, count, price, rating, or other preferences.
- recent history: The recent actions and observations from the shopping
trajectory.
- candidate actions: The list of actions currently available to the actor.
INSTRUCTIONS:
1. check constraints:
- Identify the important constraints in the user instruction.
- Prefer actions that move toward satisfying all constraints.
2. check progress:
- Avoid repeating unhelpful actions.
- Avoid going back unless it clearly helps recover from a mistake.
- Prefer clicking the most relevant product, option, or navigation action.
3. choose action:
- Choose exactly ONE best next action.
- The action must be one of the candidate actions.
- If no candidate action is clearly useful, choose the safest progress-making
action.
- Use "none" only when no valid progress-making action exists.
4. rationale:
- Provide a very short reason explaining why the chosen action is appropriate.
- Keep the rationale concise, one sentence at most.
```

G Simulation Settings

G.1 Benchmarks for Interactive Environments

This section provides detailed descriptions of the interactive environments used in our experiments, including task configurations, scenario distributions, and evaluation protocols. These details are omitted from the main paper for brevity.

ALFWorld [69] is a text-based household simulation environment built on top of ALFRED, where agents interact with a partially observed home environment through natural language actions. Each task consists of a high-level instruction (e.g., placing, cleaning, or heating objects) that must be decomposed into a long sequence of grounded actions such as navigation, object manipulation, and state changes. We follow the standard evaluation protocol, using the unseen test split to assess generalization to novel layouts and object configurations. Episodes typically require dozens of sequential actions, making performance sensitive to early mistakes and error accumulation.

BabyAI-Text [70] provides a natural-language interface to the BabyAI grid world, where agents must follow compositional instructions involving spatial relations, object attributes, and temporal ordering. Although the environment dynamics are deterministic and the observation space is compact, tasks are generated procedurally to ensure a wide range of unseen instruction combinations at test time. We evaluate on held-out instruction distributions to measure compositional generalization and multi-step planning accuracy independent of perceptual noise.

WebShop [71] is a web-based e-commerce environment that simulates realistic online shopping scenarios. Agents interact with dynamically generated product pages using text-based actions such as

search queries, page navigation, and item selection. Each episode requires identifying and purchasing an item that satisfies multiple constraints specified in a natural-language goal. The environment presents noisy, unstructured textual information and numerous distractors, requiring agents to balance exploration cost and task success. For SAG, after the initial search step, $\mathcal{A}_t$ is constructed from the finite set of clickable actions exposed in the current page observation, such as visible products, item options, navigation controls, and `Buy Now`. Thus, the ambiguity gate and critic operate only over this finite candidate set rather than the open-ended space of possible search queries.

Across all environments, tasks require sustained reasoning and adaptive decision making over extended horizons, but differ in the primary source of difficulty: partial observability and error propagation in ALFWorld, compositional generalization in BabyAI-Text, and information-seeking under noise and cost constraints in WebShop. Together, these benchmarks provide a comprehensive testbed for evaluating long-horizon agent behavior under diverse failure modes.

Scope of the paper: Why not math or QA benchmarks We focus on interactive, long-horizon agent benchmarks where agents must act online and where latency and token-budget efficiency materially affect performance. Offline final-answer reasoning benchmarks such as math, code, and QA are out of scope, as they typically allow extended deliberation or backtracking without the compounding interaction delay central to our setting.

G.2 Online Self-Improvement Protocol

SAG updates the actor during a sequential online interaction stream. For each episode, the actor interacts with the environment using the current gate and critic. If the episode succeeds, we store the resulting context–action pairs (x_t, a_t) in the buffer $\mathcal{D}$, where a_t is the executed admissible action, either selected by the actor or suggested by the critic when critique is invoked. Although the actor prompt may elicit reasoning before action selection, online SFT is applied only to the executed action target, not to the generated reasoning trace.

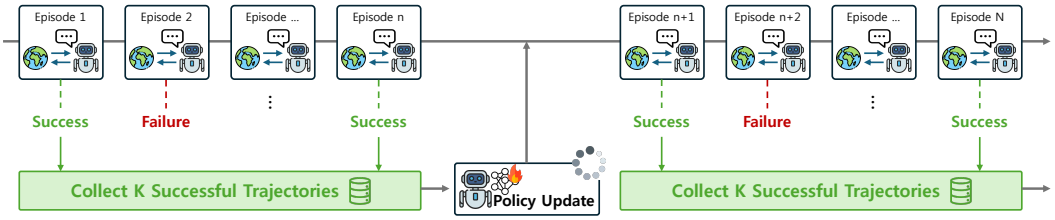


Figure 9: Overall online self-improvement process

Each episode is evaluated only once in the main protocol. If an episode later contributes successful context–action pairs to $\mathcal{D}$, it is not re-evaluated after the actor update; the updated actor is used only for subsequent episodes. After every K successful episodes, we update only the actor with SFT on $\mathcal{D}$ while keeping the critic fixed. The overall online self-improvement process is provided in Figure 9.

At each update, we fine-tune from the same base actor rather than continuing from the previous updated actor. This follows bootstrapping-based self-training [68] and reduces drift from repeated updates. Unless otherwise stated, we use $K = 10$ successful episodes per update. We study the sensitivity to K in Appendix H.10.

G.3 Hyperparameters

All of our experiments were conducted on a single NVIDIA RTX PRO 6000 Blackwell GPU with 96GB VRAM, and the detailed hyperparameters and training configurations for simulation are provided in Table 7.

G.4 Detailed description of baselines

We compare SAG against representative LLM agent frameworks.

- **ReAct** [1]: A reasoning-and-acting framework that interleaves intermediate reasoning with environment actions. ReAct generates thoughts to decompose tasks and guide action selection while interacting with external environments. It serves as a baseline without explicit critique.

Table 7: SAG hyperparameters and online SFT configuration (shared across benchmarks unless otherwise noted).

Component	Value	Component	Value
Uncertainty-based critique gate			
Gate threshold τ	0.4	Margin threshold δ	0.2
Online SFT (self-improvement)			
History window	8 steps	Max obs. length	800 chars
Buffer capacity	50,000	Successes per update	10
Train batch size	2	Grad. accumulation	8
Train steps per update	80	Learning rate	5×10^{-5}
Max sequence length	1024	Optimizer	AdamW
Grad. clipping	1.0	LoRA dropout	0.05
LoRA rank r	8	LoRA scaling α	16

- **Reflexion** [16]: A self-critique approach that generates reflections to guide future trials. Reflections are stored in memory and retrieved to improve subsequent attempts. Critique is applied in a fixed manner without adapting to step-level risk or cost.
- **Debate** [5]: A multi-agent deliberation approach where multiple agents discuss candidate solutions before acting. Structured argumentation helps identify errors and refine reasoning collaboratively. While improving robustness, it incurs high computational cost due to parallel agent executions.
- **CER** [6]: A confidence-based framework that estimates uncertainty and uses it for selection or aggregation. It quantifies model confidence to filter low-quality outputs or combine predictions. While leveraging uncertainty signals, CER does not explicitly model critique as a decision nor invoke an external critic.
- **Traj-Boost** [38]: A trajectory-bootstrapping method that improves performance by reusing successful trajectories. It enhances learning from experience by reusing high-quality execution patterns. However, it does not provide execution-time critique or error correction during task execution.
- **Retrospex** [19]: An always-on critique framework that integrates a learned offline reinforcement learning critic by re-scoring candidate actions at every step. The critic estimates action quality based on historical trajectories, providing lightweight value assessments. Although the critic is lightweight, critique is applied throughout execution without selective gating.
- **LAC** [17]: An always-on critique method that queries an LLM-based critic at every decision step. The critic provides numerical, value-based evaluations of candidate actions to improve action quality. The frequent critic invocation leads to substantial inference cost.
- **Ours (SAG)**: A cost-aware framework that treats critic invocation as a decision and selectively queries a critic based on actor ambiguity. By combining selective critique with online self-improvement, SAG aims to achieve strong performance with significantly improved token efficiency.

G.5 Scope Relative to RL-based Agent Post-training

SAG and RL-based agent post-training address complementary regimes of LLM agent improvement. RL-based methods such as PPO-style optimization, RLOO/REINFORCE-style optimization, GRPO, and GiGPO improve the underlying policy before deployment by collecting environment rollouts, computing rewards or advantages, and updating model parameters through policy optimization [52, 54, 53]. These methods can achieve strong task-specific performance when sufficient rollout budget, reward supervision, and optimization compute are available. SAG instead targets a deployment-time regime: given an available actor and a non-oracle critic, it asks when critique is worth its token and latency cost during interaction.

Different optimization stage RL-based post-training invests compute before deployment to produce a stronger policy. A typical RL pipeline repeatedly samples trajectories from the current policy, evaluates them with task rewards or preference-derived signals, estimates advantages, and performs policy updates, often with KL control against a reference policy. In contrast, SAG does not aim to replace this training stage. It operates during interaction, using a lightweight gate to decide whether an already available actor should query an auxiliary critic at the current step. Thus, RL methods

Table 8: Scope comparison between RL-based agent post-training and SAG.

Dimension	RL-based agent post-training	SAG
Primary goal	Improve the underlying policy before deployment	Decide when to use critique during deployment
Typical methods	PPO, GRPO, GiGPO	Gated critique + online SFT
Training signal	Rewards, preferences, or advantage estimates	Successful trajectories from online interaction
Rollout structure	Multiple sampled trajectories/completions per task	One interaction stream with selective critique
Models during training	Actor, often reference policy, optimizer states	Actor update; critic fixed and queried selectively
Cost focus	Training-time rollout and optimization budget	Inference-time token, critic-call, and latency cost
Hyperparameters	LR, KL, clipping, group size, reward normalization, rollout budget	Gate thresholds τ, δ and update frequency
Comparison regime	Strong task-specific policy after post-training	Cost-aware improvement of an available actor

primarily optimize *what policy is learned*, whereas SAG optimizes *when external critique should be used* under inference-time cost.

Different compute and memory regime The computational structure of RL-based post-training differs substantially from SAG. Group-based RL methods such as GRPO and GiGPO sample multiple trajectories or completions per task, compare them through reward or advantage estimates, and then reprocess these trajectories for policy-gradient updates. Many implementations also maintain a reference policy for KL regularization and store old log-probabilities, rollout buffers, and optimizer states. By contrast, SAG uses a single interaction trajectory at a time, invokes the critic only on selected ambiguous steps, and performs lightweight SFT on successful trajectories. In our implementation, only the actor is updated while the critic remains fixed. This makes SAG closer to inference-time adaptation or lightweight online SFT than to full RL post-training.

Different token-cost accounting RL-based methods may have low inference-time cost after training, but their training phase can consume large numbers of environment interactions and generated tokens due to repeated rollouts. This training-time token budget is not directly comparable to SAG's inference-time critique budget. SAG explicitly studies the cost incurred while the agent is solving tasks: token usage includes actor and critic tokens during interaction, and the method is designed to reduce this cost by avoiding unnecessary critique calls. A fair like-for-like comparison would therefore need to control not only final inference tokens, but also rollout budget, number of environment interactions, training tokens, GPU hours, reward supervision, and task-specific tuning.

Different practical assumptions RL-based post-training often requires task-specific reward design or reliable environment rewards, parallel rollout infrastructure, careful hyperparameter tuning, and repeated optimization over the target environment. For example, policy-gradient methods can be sensitive to learning rate, KL coefficient, clipping range, group size, reward normalization, and rollout budget. SAG assumes a different practical setting: the user already has an instruction-tuned actor and an imperfect critic, and wants to improve reliability during deployment without large-scale task-specific RL training. Its main hyperparameters are the gate threshold τ and margin threshold δ, which we find to be robust across a broad range in Appendix H.5.

Complementarity These differences do not imply that RL-based post-training and SAG are mutually exclusive. A stronger RL-trained agent could still benefit from SAG at deployment time, since even strong policies may encounter ambiguous states where external critique is useful. Conversely, SAG can provide a lightweight alternative when large-scale RL training is impractical or when the goal is to manage token and latency costs during online interaction. We therefore treat RL-based agent post-training as complementary to SAG, and focus our main empirical comparisons on inference-time and lightweight-adaptation baselines that share SAG's deployment-time regime.

H Extra Simulation Results

H.1 Extended Model Scaling with Always-on Critique

We extend the capacity analysis to larger Qwen backbones up to 32B and compare three configurations: **w/o critic**, which uses the actor without critique; **always-on critic**, which removes the gate and invokes the critic at every step; and **SAG**, which uses gated critique. For SAG, the actor and critic use the same backbone and model size. We report success rate, token usage, and critic calls per

episode. Across benchmarks and model sizes, SAG consistently improves over the no-critic baseline and usually matches or exceeds always-on critique while using far fewer critic calls and substantially lower token cost.

Table 9: Extended scaling results with no critic, always-on critic, and SAG. Token usage includes actor and critic tokens.

Benchmark	Backbone	w/o critic			always-on critic			SAG		
		Succ.	Token	Call	Succ.	Token	Call	Succ.	Token	Call
ALFWorld	3B	21.6	57	–	59.0	271	15.1	**69.4**	58	3.2
	7B	25.4	55	–	74.8	273	14.7	**78.3**	56	2.6
	14B	77.2	49	–	80.5	292	13.3	**82.1**	53	2.3
	32B	79.1	48	–	86.2	308	13.2	**87.2**	54	2.1
BabyAI	3B	34.0	59	–	52.3	356	10.3	**61.8**	79	3.3
	7B	38.1	55	–	69.6	377	10.1	**70.0**	78	2.8
	14B	53.3	53	–	76.2	382	8.5	**76.3**	71	2.6
	32B	61.7	53	–	**86.2**	308	8.4	82.8	68	2.3
WebShop	3B	10.9	11	–	25.7	54	6.8	**34.2**	11	2.3
	7B	13.2	11	–	33.2	57	6.5	**37.0**	9	2.2
	14B	18.9	12	–	41.0	61	5.9	**45.7**	9	1.8
	32B	25.3	9	–	43.5	62	5.7	**48.1**	9	1.6

Two trends are consistent across benchmarks. First, selective critique remains useful even for stronger actors: SAG improves over the no-critic baseline at every model scale. Second, always-on critique often improves success but incurs much higher token cost and critic-call frequency. SAG achieves a better performance–cost trade-off by concentrating critique on ambiguous steps rather than invoking it throughout the trajectory.

H.2 Effect of Critic Capacity

We analyze whether SAG benefits from stronger critics by fixing the actor to Qwen2.5-7B-Instruct and varying the critic size on ALFWorld.

Table 10: Effect of critic capacity with a fixed 7B actor on ALFWorld.

Actor / Critic	Succ. ↑
7B / No critic	25.4
7B / 3B critic	76.9
7B / 7B critic	78.3
7B / 14B critic	**82.1**

A stronger critic further improves success, but even a smaller 3B critic provides a large gain over the no-critic baseline. This suggests that SAG does not require an oracle or a substantially stronger model; rather, it benefits from selectively invoking an auxiliary perspective at ambiguous decision points.

H.3 Matched Online-SFT Controls

To isolate the effect of selective critique from online self-improvement alone, we compare SAG with two matched online-SFT controls. **ReAct + online SFT** uses the same successful-trajectory collection and actor-update protocol as SAG, but does not use any critic. **LAC + online SFT** uses the same update protocol with always-on critique, invoking the critic at every step. Thus, all three methods share the same online self-improvement mechanism, but differ in how critique is used.

The results show that online self-improvement alone does not explain SAG's gains. ReAct + online SFT improves only modestly because it can learn only from trajectories generated without critique,

Table 11: Matched online-SFT controls across all benchmarks. All methods use the same online update protocol. Token usage includes both actor and critic tokens.

Method	ALFWorld		BabyAI-Text		WebShop	
	Succ. ↑	#Token ↓	Succ. ↑	#Token ↓	Succ. ↑	#Token ↓
ReAct + online SFT	30.2	57	40.8	56	15.3	12
LAC + online SFT	74.2	341	67.2	438	32.1	84
SAG	**78.3**	**56**	**70.0**	**78**	**37.0**	**9**

which limits the quality and diversity of successful supervision. LAC + online SFT benefits from critic-assisted trajectories, but its always-on critique incurs substantially higher token cost: 341k on ALFWorld, 438k on BabyAI-Text, and 84k on WebShop. SAG achieves the best success on all three benchmarks while using token budgets close to or lower than ReAct + online SFT and far below LAC + online SFT. This confirms that the improvement comes from the combination of selective critique and online self-improvement, rather than from online SFT alone.

H.4 Robustness to Trajectory Ordering

To verify the robustness of the online self-improvement method, we repeated the ALFWorld online evaluation under different random episode orders. As online SFT updates depend on the accumulation of successful trajectories, episode order could potentially influence when updates occur and how much improvement is observed. We therefore compare the default benchmark order with shuffled orders using different seeds.

Table 12: Robustness of online-SFT across different episode orders

Order Setting	Seed / Order	Success (%)	Token (k)
Default benchmark order	default	78.3	56
Shuffled order	1	78.7	56
Shuffled order	2	78.2	58
Mean $\pm$ std over shuffled orders	default/1/2	78.3 ± 0.3	56.7 ± 1.2

The results show that success rate and token usage remain stable across order settings, and the qualitative conclusion remains unchanged. This suggests that the gains are not driven by one favorable sequence of episodes, but are robust to task order.

H.5 Gate Threshold Sensitivity

SAG uses the same thresholds across benchmarks and backbones unless otherwise specified. We analyze the sensitivity to the entropy threshold τ and margin threshold δ on ALFWorld. The results show that τ primarily controls the cost–success trade-off, while δ has a relatively mild effect on success over a broad range.

Table 13: Sensitivity to τ and δ on ALFWorld.

τ	Succ. ↑	#Token ↓	δ	Succ. ↑	#Token ↓
0.2	79.0	74	0.1	77.1	53
0.3	78.9	63	0.2	**78.3**	56
0.4	**78.3**	56	0.4	78.1	61
0.5	75.1	49	0.5	77.8	64
0.7	66.1	45	1.0	75.3	60

Performance degrades mainly when τ is set too high, causing the agent to under-use critique. In contrast, nonzero values of δ yield similar success, indicating that the local ambiguity term is useful but does not require fine-grained tuning.

H.6 Simulation results on BabyAI-Text

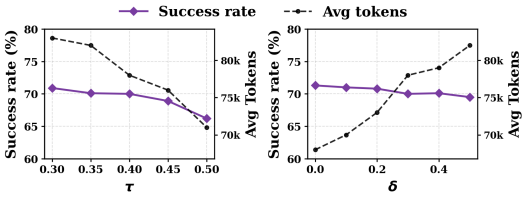
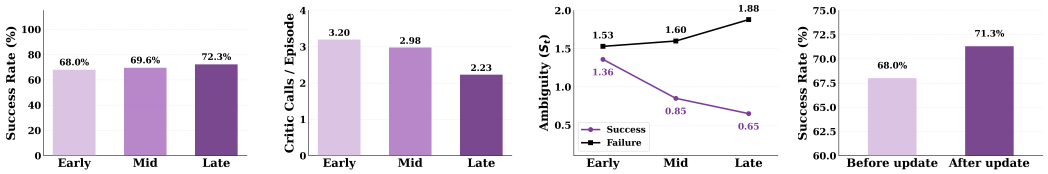


Figure 10: Parameter sensitivity analysis on BabyAI-Text benchmark

Parameter sensitivity Figure 10 reports the sensitivity of SAG to the gate threshold τ and the margin threshold δ on BabyAI-Text. Varying τ yields a smooth cost–performance trade-off: as τ increases from 0.30 to 0.50, success decreases moderately from 70.9% to 66.2%, while the average token usage drops substantially from 83k to 71k tokens. Within the mid-range ($\tau \in [0.35, 0.45]$), success remains relatively stable around 70%, but token cost continues to decrease (82k $\rightarrow$ 76k), indicating that τ can be used as an effective knob for cost calibration. In contrast, SAG is largely insensitive to the margin threshold δ: success stays within a narrow band of 69.5%–71.3% across all tested values, while larger δ primarily increases token usage from 68k to 82k by triggering critique more often. These trends mirror those observed on ALFWorld, suggesting that τ governs the main cost–performance trade-off, whereas δ can be chosen from a broad range without fine-grained tuning.

Learning dynamics under online self-improvement Figure 11 shows the learning dynamics of SAG under online self-improvement on the BabyAI-Text benchmark. As interaction proceeds, the success rate steadily improves from 68.0% (Early) to 69.6% (Mid) and 72.3% (Late; Figure 11a), while the average number of critic calls per episode decreases from 3.20 to 2.98 and finally 2.23 (Figure 11b). This reflects SAG's intended behavior: as successful trajectories are distilled into the actor via online SFT, the actor increasingly resolves decisions on its own under the same gating rule.

Figure 11c further shows that ambiguity remains well separated between successes and failures over time. For successful episodes, the ambiguity score S_t decreases from 1.36 to 0.65, whereas failures maintain higher ambiguity and even become more uncertain (1.53 $\rightarrow$ 1.88), indicating that remaining errors concentrate on harder decision points. Moreover, performance on previously seen episodes improves after online updates, increasing from 68.0% to 71.3% (Figure 11d), suggesting that SAG does not forget past experience and can correct earlier failures. Overall, these results corroborate the findings on ALFWorld: critique acts as a transient scaffold, with SAG gradually internalizing effective behaviors and shifting decisions toward lower-risk states without increasing inference-time computation.



(a) Success rate (%) (b) Critic calls / episode (c) S_t for success vs. failure (d) Seen: before vs. after

Figure 11: Learning dynamics of SAG under online self-improvement. Metrics are aggregated into Early/Mid/Late phases over 100 evaluation episodes on BabyAI-Text benchmark.

H.7 Simulation results on WebShop

Parameter sensitivity Figure 12 analyzes the sensitivity of SAG to τ and δ on the WebShop benchmark. Increasing τ leads to a monotonic reduction in token usage (12k $\rightarrow$ 8k tokens), while success decreases from 37.2% to 33.0%. This reflects the higher sensitivity of WebShop to missed critique in open-domain information-seeking tasks. By contrast, success remains stable across a wide range of δ values (36.9%–37.2%), indicating robustness to the local ambiguity threshold. As in other environments, larger δ increases critique frequency and token usage, confirming that δ primarily affects cost rather than reliability.

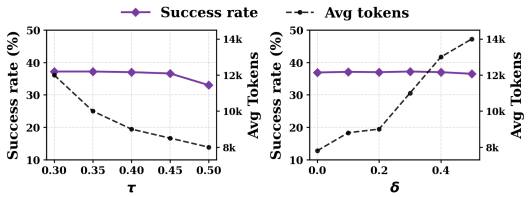
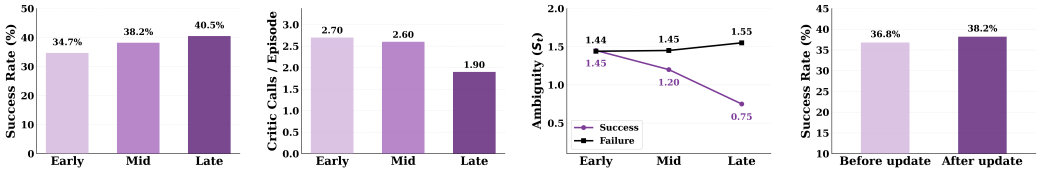


Figure 12: Parameter sensitivity analysis on WebShop benchmark

Learning dynamics under online self-improvement Figure 13 presents the learning dynamics of SAG on WebShop. Success rate steadily improves from 34.7% to 38.2% and 40.5% over the Early/Mid/Late phases (Figure 13a), while the average number of critic calls per episode decreases from 2.7 to 1.9 (Figure 13b). Ambiguity remains well separated between successes and failures throughout learning (Figure 13c): successful trajectories exhibit decreasing ambiguity ($1.45 \rightarrow 0.75$), whereas failures remain consistently more ambiguous. Moreover, performance on previously seen episodes improves after online updates, increasing from 36.8% to 38.2% (Figure 13d). These results align with those observed on ALFWorld and BabyAI-Text, demonstrating that SAG consistently transforms critique from a frequent intervention into a transient scaffold that accelerates learning while preserving inference efficiency.



(a) Success rate (%) (b) Critic calls / episode (c) S_t for success vs. failure (d) Seen: before vs. after

Figure 13: Learning dynamics of SAG under online self-improvement. Metrics are aggregated into Early/Mid/Late phases over 200 evaluation episodes on WebShop benchmark.

H.8 Online SFT vs. One-shot SFT

We compare SAG's online self-improvement with a one-shot SFT variant that first collects successful trajectories and then performs a single actor update. Online-SFT incorporates critic-corrected behaviors earlier, allowing subsequent rollouts to benefit from the updated actor and reducing later reliance on critique.

Table 14: Comparison between one-shot SFT and online-SFT.

Method	**ALFWorld**		**BabyAI-Text**		**WebShop**	
	Succ. ↑	**#Token** ↓	**Succ.** ↑	**#Token** ↓	**Succ.** ↑	**#Token** ↓
One-shot SFT	72.8	63	66.9	82	34.8	10
Online SFT (SAG)	**78.3**	**56**	**70.0**	**78**	**37.0**	**9**

Online SFT consistently improves success while maintaining comparable or lower token usage. This supports the view that early distillation of critic-assisted successful behavior improves subsequent interaction, whereas one-shot SFT misses this compounding benefit.

H.9 Failure Modes and Non-oracle Critic Behavior

SAG assumes that critique is useful but not perfect. We observe three common failure modes.

Incorrect critic suggestion The critic may suggest an action that is plausible but suboptimal under the current environmental state. This can occur when the critic misinterprets the interaction history or overlooks a constraint in the admissible action set.

Missed critique opportunity The gate may fail to invoke critique when the actor is confidently wrong. This false-negative case is difficult for any actor-side uncertainty method, since the actor's distribution does not expose the underlying error.

Unrecoverable early error In long-horizon environments, some early mistakes create states from which later critique cannot recover. This suggests that future work could combine step-level gating with trajectory-level monitoring.

These cases highlight that SAG does not rely on an oracle critic. Its gains come from invoking imperfect critique selectively at high-risk steps, while avoiding the systematic cost of always-on feedback.

H.10 Sensitivity to Online Update Frequency

We vary the number of successful episodes required before each actor update. Table 15 shows that SAG is robust across a moderate range of update frequencies. We use $K = 10$ as the default because it provides a good balance between success, critic usage, and token cost.

Table 15: Sensitivity to the number of successful episodes per update on ALFWorld.

Successes / Update	Succ. ↑	Critic Calls / Ep. ↓	#Token ↓
5	76.9	2.52	58
10	**78.3**	**2.10**	**56**
20	78.1	2.28	57
30	77.7	2.42	58

H.11 Qualitative Analysis of Prompt-Induced Responses

Although the actor and critic share the same backbone models, they can produce different outputs because they operate under different prompts and objectives. This difference in prompt framing, rather than model capacity, is what allows the critic to reason beyond the actor's current step. To examine how prompt composition shapes this behavior, we remove individual components from the full critic prompt and compare the generated rationales at identical decision points.

Table 16: Qualitative examples of how critic prompts affect the generated rationales

Prompt variant	Step	Rationale by critic
Full prompt	5	Search shelves for both bowl and desklamp
w/o expected JSON example	5	Search shelves as likely bowl location
w/o goal	5	Bowl may be on an unchecked shelf
Full prompt	9	Check whether bowl is near the found desklamp
w/o expected JSON example	9	Re-examine shelf for missed items
w/o goal	9	Bowl may be near the desklamp
Full prompt	20	Return to desklamp to use it
w/o expected JSON example	20	Use desklamp to look at bowl
w/o goal	20	Desklamp is needed to illuminate bowl

As shown in Table 16, the full prompt tends to produce rationales that reflect the broader task structure, whereas the ablated prompts shift the rationale toward more local cues, such as likely object locations

or immediate re-examination. These results indicate that even with the same underlying models, the actor and critic can attend to different aspects of the task and produce meaningfully different outputs.

H.12 Action-Only versus Reasoning-Consistent Supervision

We compare action-only SFT with reasoning-consistent SFT, where the critic rewrites the reasoning to support a corrected action before supervision. Following online adaptation on 134 ALFWorld episodes, we freeze each actor and evaluate without further updates. We report ID and OOD results, including evaluation on 140 episode-disjoint held-out examples.

Table 17: Comparison of SFT targets. Actor denotes execution without critique; Calls denotes critic calls per episode.

	ID			OOD		
SFT target	Actor SR	SAG SR	Calls	Actor SR	SAG SR	Calls
No SFT	24.6	72.3	3.00	25.7	70.7	2.85
Matched reasoning	43.2	76.0	1.35	42.9	77.8	1.67
Action-only	**45.7**	**80.3**	**1.18**	**43.4**	**80.9**	**1.41**

Action-only supervision achieves higher success and fewer critic calls than reasoning-consistent supervision in both evaluations. These results support the effectiveness of action-only supervision without requiring imitation of critic-generated reasoning.

H.13 Empirical Validation of the Gating Score

We evaluate a frozen actor on 10 randomly sampled BabyAI test episodes, collecting 178 non-terminal decision states. At each actor-visited state, we record the gating score S_t and whether the actor's top-1 action disagrees with BabyAI's built-in replanning bot. We sort states by S_t into four approximately equal-sized bins, without conditioning on whether the reference action lies among the actor's top-two predictions.

Table 18: Reference-action disagreement across gating-score quartiles, ordered from lowest to highest S_t.

Score quartile	Q1	Q2	Q3	Q4
Number of states	45	45	44	44
Disagreements	8	10	14	20
Disagreement rate (%)	17.8	22.2	31.8	45.5

Disagreement increases monotonically from 17.8% to 45.5% across score quartiles, supporting the combined score as a practical indicator of decision ambiguity. The bot is used only as a reference for this diagnostic analysis: disagreement is not a ground-truth error rate and does not extend the conditional guarantee of the margin proposition.

I Example Trajectories

Table 19: An example trajectory from ALFWorld for `pick_two_obj_and_place`. SAG selectively invokes the critic based on ambiguity (global entropy and local margin) and stores successful trajectories for online self-improvement.

You are in the middle of a room. Looking quickly around you, you see an armchair 1, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1, a dresser 1, a garbagecan 1, a safe 1, a shelf 12, a shelf 11, a shelf 10, a shelf 9, a shelf 8, a shelf 7, a shelf 6, a shelf 5, a shelf 4, a shelf 3, a shelf 2, a shelf 1, a sidetable 1, and a sofa 1. Your task is to: put two pillow in sofa.
> [Agent] Act 1: think: To solve the task, I need to find and take the first pillow, then put it in sofa, then find and take the second pillow, then put it in sofa.
OK.
> [Agent] Act 2: think: First I need to find the first pillow. A pillow is more likely to appear in armchair (1), cabinet (1-4), drawer (1-5), dresser (1), sidetable (1), sofa (1). I can check one by one, starting with armchair 1.
OK.
> [Agent] Act 3: go to armchair 1
You arrive at armchair 1. On the armchair 1, you see a cellphone 2, a pillow 2, and a pillow 1.
> [Agent] Act 4: take pillow 1 from armchair 1
You pick up the pillow 1 from the armchair 1.
> [Agent] Act 5: think: Now I find the first pillow (1). Next, I need to put it in sofa 1.
OK.
> [Agent] Act 6: go to sofa 1
You arrive at sofa 1. On the sofa 1, you see a newspaper 3.
> [Gate] global ambiguity $\bar{H}_t = 0.20$, local margin $m_t = 0.06 < \delta$.
> [Gate] score $S_t = \bar{H}_t + \mathbb{I}[m_t < \delta] = 1.20 > \tau$ (VoI proxy $>$ cost) $\Rightarrow$ critic invoked.
> [Critic] Suggested action: move **pillow 1** to **sofa 1**.
> [Critic] Rationale: You are close, but you forgot to put the first pillow in the sofa before searching for the second one. Putting the first pillow in the sofa now reduces the risk of forgetting it later.
> [Agent] Act 7: move pillow 1 to sofa 1
On the sofa 1, you see a newspaper 3, and a pillow 1.
> [Agent] Act 8: go to armchair 1
You arrive at armchair 1. On the armchair 1, you see a cellphone 2, and a pillow 2.
> [Agent] Act 9: take pillow 2 from armchair 1
You pick up the pillow 2 from the armchair 1.
> [Agent] Act 10: think: Now I find the second pillow (2). Next, I need to put it in sofa 1.
OK.
> [Agent] Act 11: go to sofa 1
You arrive at sofa 1. On the sofa 1, you see a newspaper 3, and a pillow 1.
> [Agent] Act 12: move pillow 2 to sofa 1
You move the pillow 2 to the sofa 1.
Outcome: success
> [SAG] Successful episode. Trajectory is added to the self-improvement buffer.

Table 20: Example trajectory on ALFWorld `pick_clean_then_place_in_recep`. SAG invokes critique when ambiguity indicates high error risk, preventing unnecessary detours.

You are in the middle of a room. Looking quickly around you, you see a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a countertop 1, a garbagecan 1, a handtowelholder 2, a handtowelholder 1, a sinkbasin 2, a sinkbasin 1, a toilet 1, a toiletpaperhanger 1, and a towelholder 1. Your task is to: put a clean cloth in countertop.
> [Agent] Act 1: think: To solve the task, I need to find and take a cloth, clean it with a sinkbasin, then place it on the countertop.
OK.
> [Agent] Act 2: think: First I need to find a cloth. I will check the cabinets starting from cabinet 4.
OK.
> [Agent] Act 3: go to cabinet 4
You arrive at cabinet 4. The cabinet 4 is closed.
> [Agent] Act 4: open cabinet 4
You open the cabinet 4. The cabinet 4 is open. In it, you see a soapbottle 2.
> [Agent] Act 5: go to cabinet 3
You arrive at cabinet 3. The cabinet 3 is closed.
> [Agent] Act 6: open cabinet 3
You open the cabinet 3. The cabinet 3 is open. In it, you see a cloth 2.
> [Agent] Act 7: take cloth 2 from cabinet 3
You pick up the cloth 2 from the cabinet 3.
> [Agent] Act 8: go to sinkbasin 1
You arrive at sinkbasin 1. On the sinkbasin 1, you see a soapbar 1.
> [Agent] Act 9: clean cloth 2 with sinkbasin 1
You clean the cloth 2 using the sinkbasin 1.
> [Agent] Act 10: go to countertop 1
You arrive at countertop 1. On the countertop 1, you see a candle 1, a cloth 3, and a spraybottle 1.
> [Agent] Act 11: think: I might need to fetch soap again before placing the cloth. I should go back to cabinet 4.
OK.
> [Gate] global ambiguity $\bar{H}_t = 0.55$, local margin $m_t = 0.78$.
> [Gate] score $S_t = \bar{H}_t + \mathbb{I}[m_t < \delta] = 0.55 > \tau$ (VoI proxy $>$ cost) $\Rightarrow$ critic invoked.
> [Critic] Suggested action: move **cloth 2** to **countertop 1**.
> [Critic] Rationale: The cloth you are holding is already clean. The task only requires placing a clean cloth on the countertop. Fetching soap again is unnecessary and risks delaying completion.
> [Agent] Act 12: move cloth 2 to countertop 1
On the countertop 1, you see a candle 1, a cloth 3, and a spraybottle 1.
> [Agent] Act 13: move cloth 2 to countertop 1
You move the cloth 2 to the countertop 1.
Outcome: success
> [SAG] Successful episode. The trajectory is added to the self-improvement buffer $\mathcal{D}$.

Table 21: Example trajectory on ALFWorld `pick_heat_then_place_in_recep`. SAG corrects inefficient exploration by invoking critique under high global ambiguity.

You are in the middle of a room. Looking quickly around you, you see a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 3, a countertop 2, a countertop 1, a drawer 3, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a shelf 3, a shelf 2, a shelf 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1.
Your task is to: put a hot apple in fridge.
> [Agent] Act 1: think: First I need to find an apple. I will start by checking the fridge.
OK.
> [Agent] Act 2: go to fridge 1
You arrive at fridge 1. The fridge 1 is closed.
> [Agent] Act 3: open fridge 1
You open the fridge 1. In it, you see a bowl 1, a cup 2, a cup 1, a lettuce 1, a mug 2, a potato 1, and a tomato 1.
> [Agent] Act 4: think: No apple here. I might need to search cabinets one by one.
OK.
> [Gate] global ambiguity $\bar{H}_t = 0.21$, local margin $m_t = 0.09 < \delta$.
> [Gate] score $S_t = \bar{H}_t + \mathbb{I}[m_t < \delta] = 1.21 > \tau$ (VoI proxy $>$ cost) $\Rightarrow$ critic invoked.
> [Critic] Suggested action: go to **countertop 1**.
> [Critic] Rationale: Apples are more likely to be placed on open surfaces such as countertops. Searching cabinets first is less efficient.
> [Agent] Act 5: go to countertop 1
You arrive at countertop 1. On the countertop 1, you see an apple 1, an apple 2, and a mug 3.
> [Agent] Act 6: take apple 1 from countertop 1
You pick up the apple 1 from the countertop 1.
> [Agent] Act 7: go to microwave 1
You arrive at microwave 1. The microwave 1 is closed.
> [Agent] Act 8: heat apple 1 with microwave 1
You heat the apple 1 using the microwave 1.
> [Agent] Act 9: think: The apple is now hot. I should place it in the fridge.
OK.
> [Agent] Act 10: go to fridge 1
You arrive at fridge 1. The fridge 1 is open.
> [Agent] Act 11: move apple 1 to fridge 1
You move the apple 1 to the fridge 1.
Outcome: success
> [SAG] Successful trajectory is added to the self-improvement buffer $\mathcal{D}$.

Table 22: Example trajectory on BabyAI-Text `pick_up_green_key`. SAG intervenes selectively to prevent looping behavior.

Your mission: pick up the green key. You start in a corridor with a blue key on the floor and a closed door in front.
> [Agent] Act 1: think: The green key is not visible. I should move toward the door. OK.
> [Agent] Act 2: go forward
You are now standing directly in front of the door.
> [Gate] global ambiguity $\bar{H}_t = 0.28$, local margin $m_t = 0.08 < \delta$.
> [Gate] score $S_t = 1.28 > \tau \Rightarrow$ critic invoked.
> [Critic] Suggested action: open the door.
> [Critic] Rationale: Progress requires entering the next room; waiting in front of the door adds no information.
> [Agent] Act 3: open door
The door opens. Inside you see a red ball in front, a green key on the right, and a yellow box.
> [Agent] Act 4: go forward
> [Agent] Act 5: turn left
> [Agent] Act 6: turn right
> [Gate] global ambiguity $\bar{H}_t = 0.34$, local margin $m_t = 0.07 < \delta$.
> [Gate] score $S_t = 1.34 > \tau \Rightarrow$ critic invoked.
> [Critic] Suggested action: turn right toward the green key.
> [Critic] Rationale: The goal object is already identified; reorientation loops are unnecessary.
> [Agent] Act 7: turn right
> [Agent] Act 8: go forward
> [Agent] Act 9: pick up green key
> [Agent] Act 10: stop
Outcome: success
> [SAG] Successful trajectory with two gated interventions is added to the self-improvement buffer $\mathcal{D}$.

Table 23: Example trajectory on WebShop `buy_red_tshirt_under_20`. SAG resolves constraint ambiguity via targeted critique.

Your task: purchase a red T-shirt under $20. The homepage lists categories: shirts, shoes, bags.
> [Agent] Act 1: click "shirts"
The page shows 5 items: blue T-shirt ($15), red T-shirt ($25), red T-shirt ($18), green hoodie ($22), white T-shirt ($10).
> [Agent] Act 2: think: There are two red T-shirts. I might choose the $25 one.
> [Gate] global ambiguity $\bar{H}_t = 0.33$, local margin $m_t = 0.04 < \delta$.
> [Gate] score $S_t = 1.33 > \tau \Rightarrow$ critic invoked.
> [Critic] Suggested action: add the $18 red T-shirt to the cart.
> [Critic] Rationale: The $25 option violates the budget constraint.
> [Agent] Act 3: add red T-shirt ($18) to cart
> [Agent] Act 4: buy
Outcome: Reward 1.0, success
> [SAG] Successful trajectory is added to self-improvement buffer.

Table 24: Example trajectory on WebShop `buy_black_backpack`. SAG intervenes twice to enforce constraints and finalize execution.

Your task: purchase a cheap black backpack.
> [Agent] Act 1: click "bags"
The bags page shows: black backpack ($55), black backpack ($25), red backpack ($20).
> [Agent] Act 2: think: The $55 one might be better quality.
> [Gate] global ambiguity $\bar{H}_t = 0.24$, local margin $m_t = 0.07 < \delta$.
> [Gate] score $S_t = 1.24 > \tau \Rightarrow$ critic invoked.
> [Critic] Suggested action: add the $25 black backpack to the cart.
> [Critic] Rationale: The task requires a cheap black backpack; $55 violates this constraint.
> [Agent] Act 3: add black backpack ($25) to cart
> [Agent] Act 4: view cart
> [Agent] Act 5: think: The cart satisfies color and price.
> [Agent] Act 6: go to checkout
> [Gate] global ambiguity $\bar{H}_t = 0.31$, local margin $m_t = 0.06 < \delta$.
> [Gate] score $S_t = 1.31 > \tau \Rightarrow$ critic invoked.
> [Critic] Suggested action: place the order.
> [Critic] Rationale: No further browsing is required once all constraints are satisfied.
> [Agent] Act 7: buy
Outcome: Reward 1.0, success

J GenAI Usage Disclosure

We acknowledge the use of large language model (LLM) assistants in the preparation of this paper as follows:

- **Writing assistance:** ChatGPT (OpenAI) was used to improve grammar, rephrase sentences, and refine the clarity of certain passages. All conceptual content, experimental design, analysis, and claims were authored solely by the listed authors.
- **Code generation:** No LLM-based tools were used for writing or generating code in this study.
- **Data processing or analysis:** No LLM-based tools were used for data preprocessing, analysis, or the generation of results.

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: We have claimed our scope, and contribution in the section of Introduction.

   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: We discuss the limitations in Appendix A.

   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 assumptions and theoretical results are provided in Section 4.2. Complete mathematical proofs are provided in Appendix C, D, and E.

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: We provide the main experimental settings and hyperparameters needed to reproduce the results in Section 5 and Appendix F and G.

   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: We provide the source code for our proposed method and scripts to reproduce the experimental results.

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: Dataset descriptions and experimental settings are provided in Section 5, and detailed hyperparameters are provided in Appendix F and G.

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 report the mean and one standard deviation to indicate the reliability of the results, as shown in Appendix H.4.

Guidelines:

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

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Computational resources are reported in Appendix G.3.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

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

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: Broader impacts are reported in Appendix A.

    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: [N/A]

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: In our experiments, we used datasets and models that are all open-source and widely utilized.

    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: [Yes]

    Justification: We provide the implementation with documentation and reproduction instructions.

    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: [N/A]

    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: [N/A]

    Guidelines:

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

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

16. **Declaration of LLM usage**

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

    Answer: [N/A]

    Justification: LLMs were used only for writing assistance, which does not affect the core methodology. We disclose this in Appendix J.

    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.