REGAIN: INFORMATION-GAIN GUIDED ROLLBACK FOR RETRIEVAL-AUGMENTED REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models can learn to interleave reasoning with search engine calls through reinforcement learning, enabling multi-step retrieval-augmented generation. However, existing approaches either rely on sparse outcome-only rewards with no per-turn signal, or even when providing process-level supervision, retain misleading retrieval content in the trajectory without correction. We propose **ReGain**, a closed-loop search quality control framework that integrates two complementary components: (1) a *per-turn quality signal* based on the change in the model's answer confidence before and after each retrieval, which serves as both a rollback trigger and a step-level process reward; and (2) a *state rollback* mechanism that detects and undoes low-quality searches in real time during RL rollouts, restoring generation state and injecting explicit query reformulation feedback. Experiments on seven QA benchmarks show that ReGain achieves the highest average F1 among existing outcome-reward and process-reward RL methods, with particularly strong gains on multi-hop reasoning tasks.

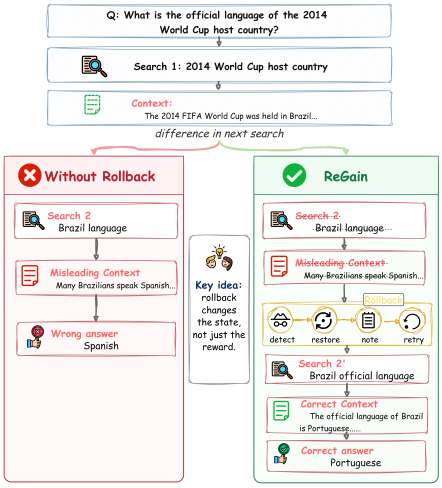


Figure 1: Without Rollback vs. ReGain. Without rollback, low-quality retrieval pollutes context and cascades into a wrong answer. With rollback, the state is restored and a reformulated query yields the correct answer.

1 INTRODUCTION

Large language model (LLM)-based agents are increasingly trained with reinforcement learning (RL) to interact with external environments through tool use (DeepSeek-AI, 2025; Schulman et al., 2017), particularly in search-based settings that require multi-turn reasoning and knowledge acquisition (Jin et al., 2025; Song et al., 2025; Zheng et al., 2025). Training LLMs to *learn when and how to search* via RL has emerged as a dominant paradigm, with methods like Search-R1 (Jin et al., 2025) demonstrating that GRPO-based RL (Shao et al., 2024) can teach models to interleave reasoning with multi-turn retrieval calls.

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

However, existing outcome-reward RL approaches suffer from inherent reward sparsity: supervision is provided exclusively upon generating the final answer. This sparsity is particularly problematic in multi-turn settings, where long trajectories exacerbate three critical issues: (i) Advantage collapse—when all rollouts in a GRPO group receive identical rewards, advantages vanish and provide no gradient signal; (ii) Lack of fine-grained credit assignment—a low-quality search followed by a corrective one receives the same outcome reward as a trajectory where every search was productive; (iii) Poor sample efficiency—each rollout yields only a single terminal signal while intermediate search quality information is wasted.

Recent work has attempted to address these issues through process-level rewards. IGPO (Wang et al., 2026) introduces information gain (IG) as an intrinsic per-turn reward, measuring how each search changes the policy's predicted probability of the correct answer. StepSearch (Wang et al., 2025) assigns token-level rewards based on document similarity to ground-truth evidence. SmartSearch (Wen et al., 2026) uses dual-level credit assessment to evaluate query novelty and usefulness. While these methods provide denser supervision than outcome-only training, most rely solely on negative rewards to penalize low-quality searches—the misleading retrieved content remains in the generation context. SmartSearch introduces query refinement via a separate model, but this adds architectural complexity. In all cases, the downstream effects of low-quality retrieval on subsequent turns remain inadequately addressed.

We identify this as the *error propagation problem*: low-quality retrieved content persists in context, introducing misleading information that biases subsequent reasoning and query formulation. To address this, we propose **ReGain**, a closed-loop search quality control framework that integrates active state rollback with information gain-based supervision:

- **Rollback-aware SFT cold start** (§3.2): We bootstrap the model's search-reasoning capabilities using constructed trajectories that include rollback scenarios, providing a strong initialization for subsequent RL training.
- **The per-turn quality signal** (§3.3): We measure information gain (IG) in log-probability space as a per-turn reward signal and, crucially, as the trigger for rollback decisions. To enable rollback at inference time without ground truth, we introduce a *self-probe* mechanism that uses the model's own greedy predictions as the probe target.
- **Active rollback intervention** (§3.4): When IG indicates a low-quality search, we restore the generation state to its pre-search snapshot, inject explicit query reformulation feedback, and reactivate the sample for retry—actively preventing error propagation rather than merely penalizing it.

Together, these components form a two-phase training pipeline: SFT teaches the structured protocol including rollback patterns, then RL with a detect-intervene-learn loop refines search quality through active intervention. Advantages are computed via separate normalization of IG and outcome signals to resolve scale mismatch (§3.5). This approach is complementary to existing process reward methods—while prior work improves *what the model learns from* (denser reward signals), ReGain improves *what the model learns on* (cleaner training trajectories free from error propagation). Experiments across seven QA benchmarks demonstrate consistent improvements, particularly on multi-hop tasks where search quality is critical (§5).

2 Related Work

Retrieval-Augmented Generation. RAG systems augment LLMs with external retrieval to mitigate hallucination and improve factuality (Lewis et al., 2020; Borgeaud et al., 2022; Izacard et al., 2023). While early approaches use fixed retrieval pipelines, recent methods enable adaptive retrieval: Self-RAG (Asai et al., 2024) trains models to critique and decide when to retrieve; CRAG (Yan et al., 2024) introduces corrective retrieval actions based on relevance scoring. IRCoT (Trivedi et al., 2023) interleaves chain-of-thought prompting with retrieval. Our work differs fundamentally by training models end-to-end via RL with an explicit mechanism to *detect and physically undo* low-quality retrievals during training rollouts.

RL for Search-Augmented Agents. Reinforcement learning has emerged as the dominant paradigm for training LLMs to interact with search engines. Search-R1 (Jin et al., 2025) demonstrates that GRPO (Shao et al., 2024) with outcome-based rewards and retrieved token masking enables multi-turn retrieval-reasoning, substantially outperforming prompting-based approaches. R1-Searcher

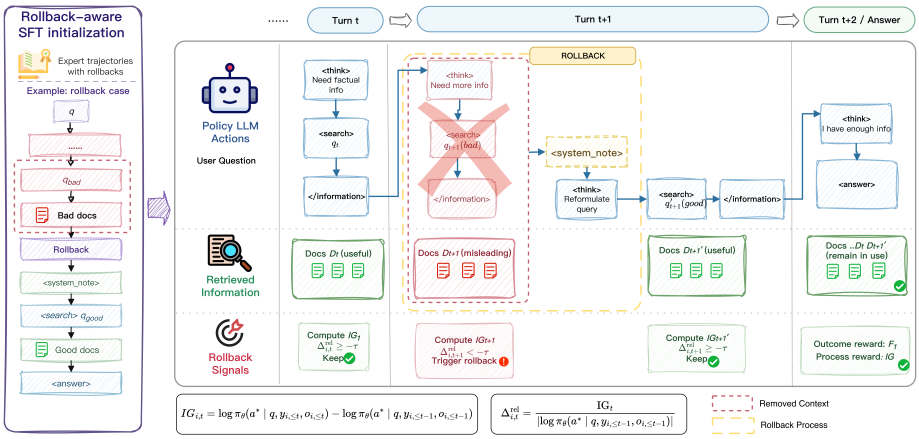


Figure 2: ReGain training pipeline. *Left*: rollback-aware SFT cold start distills expert trajectories (including rollback scenarios) to initialize the policy. *Right*: during RL, the model generates multi-turn search trajectories; after each retrieval, IG is computed to assess search quality. If IG indicates a low-quality search, the state is rolled back, a `<system_note>` listing failed queries is injected, and the model retries with clean context. Positive searches are kept and their IG values serve as per-turn rewards.

(Song et al., 2025) and DeepResearcher (Zheng et al., 2025) extend this to real-world web search environments. ReSearch (Chen et al., 2025) introduces reason-in-documents prompting within an RL loop. These approaches share a common limitation: they rely on sparse outcome-only rewards that provide no per-turn signal about intermediate search quality, leading to advantage collapse and credit assignment problems.

Process Rewards for Multi-Turn Agents. To address reward sparsity, recent work introduces step-level supervision. IGPO (Wang et al., 2026) proposes information gain—the marginal increase in the policy's probability of producing the correct answer—as a turn-level intrinsic reward, achieving significant improvements especially on smaller models where advantage collapse is most severe. StepSearch (Wang et al., 2025) assigns token-level rewards based on cosine similarity between retrieved documents and ground-truth evidence, combined with redundancy penalties. SmartSearch (Wen et al., 2026) introduces dual-level credit assessment (novelty and usefulness) for query quality, with a three-stage curriculum from SFT to DPO to RL. GiGPO (Feng et al., 2025) uses anchor-based grouping to estimate relative advantages for actions from the same state. While these methods all provide denser supervision than outcome-only training, most do not actively remove low-quality content from context. SmartSearch (Wen et al., 2026) introduces query refinement but requires a separate refine model. Our work is the first to introduce *active state rollback* driven by the policy model's own IG signal, requiring no additional model, that physically removes low-quality content during training, complementing process reward signals with trajectory-level intervention.

3 Method

3.1 Problem Formulation

Given a question q and ground-truth answer $a^* = (a_1^*, \ldots, a_L^*)$, the model generates a multi-turn trajectory $\boldsymbol{\mathcal{T}} = (y_1, o_1, y_2, o_2, \ldots, y_T, o_T, a)$ where each y_t consists of a reasoning segment (wrapped in `<think></think>`) followed by a search query (wrapped in `<search></search>`), o_t is the retrieved content (wrapped in `<information></information>`), and a is the final response (contains reasoning segment and its answer wrapped in `<answer></answer>`).

Our method follows a two-phase pipeline: SFT cold start (§3.2) bootstraps the rollback-aware format, then RL (§3.3–§3.6) refines search quality. For the RL phase, we build on Group Relative Policy Optimization (GRPO) (Shao et al., 2024). For each input (q, a^*), GRPO samples a group of G rollouts $\{\boldsymbol{\mathcal{T}}_i\}_{i=1}^{G}$ from the old policy $\pi_{\theta_{\text{old}}}(\cdot \mid q)$. Each rollout receives an outcome reward r_i^{O}: word-level F1 between the predicted answer $\hat{a}_i$ and the ground truth a^*. The group-relative advantage

for the i-th rollout is:

$$\hat{A}_i = \frac{r_i^{\mathrm{O}} - \mu_r}{\sigma_r} \tag{1}$$

where μ_r and σ_r are the mean and standard deviation of rewards within the group. The policy is then updated by maximizing a clipped surrogate objective with KL regularization against a reference policy π_{ref}.

3.2 SFT Cold Start

Our rollback mechanism (§3.4) injects the feedback block wrapped in `<system_note></system_note>` into the generation context. Without prior exposure to this format, the model cannot productively respond to rollback feedback. We therefore apply a supervised fine-tuning (SFT) cold-start phase before RL, distilling rollback-aware trajectories from GPT-5.2 so that the model learns to understand and follow rollback-augmented trajectories.

3.3 Per-Turn Quality Signal

To evaluate the quality of each search action, we measure how much it changes the model's confidence in the correct answer—a quantity named *information gain* (IG). The core idea is to construct a *probe sequence* appended to the current context, then measure the model's log-probability on that probe before and after retrieval. This difference of conditional log-likelihoods is the pointwise conditional mutual information that the retrieval carries about the correct answer under the policy (Church & Hanks, 1990; Cover & Thomas, 2006), so a positive value means the retrieval adds information about a^* and a negative value means it destroys it. We describe two variants: a training-time mode that uses the ground-truth answer, and an inference-time mode that requires no ground truth.

3.3.1 Training: GT-Based IG

During training, the ground-truth answer $a^* = (a_1^*, \ldots, a_L^*)$ is available. The ground-truth answer is wrapped in the same answer schema used during rollouts to ensure format consistency. The probe sequence $\tilde{y}^{\mathrm{gt}}$ is constructed as:

`<think>Now there's enough information to answer</think><answer>`a^*`</answer>`

We append $\tilde{y}^{\mathrm{gt}}$ to the trajectory prefix and measure the average log-probability over the $L = |a^*|$ answer tokens:

$$\log \pi_\theta(a^* \mid q, y_{i,\le t}, o_{i,\le t}) = \frac{1}{L} \sum_{j=1}^{L} \log \pi_\theta(a_j^* \mid q, y_{i,\le t}, o_{i,\le t}, a_{<j}^*) \tag{2}$$

where $y_{i,\le t}, o_{i,\le t}$ denotes the trajectory prefix $(y_{i,1}, o_{i,1}, \ldots, y_{i,t}, o_{i,t})$ of the i-th rollout up to turn t. The information gain for turn t is the marginal increase in log-probability induced by the retrieved content $o_{i,t}$:

$$\mathrm{IG}_{i,t} = \log \pi_\theta(a^* \mid q, y_{i,\le t}, o_{i,\le t}) - \log \pi_\theta(a^* \mid q, y_{i,\le t-1}, o_{i,\le t-1}) \tag{3}$$

where the first term measures confidence *after* turn t (including both $y_{i,t}$ and $o_{i,t}$), and the second measures confidence *before* turn t begins. Positive IG indicates the search improved the model's confidence in a^*; negative IG indicates low-quality retrieval.

3.3.2 Inference: Self-Probe IG (GT-Free)

At inference time, ground truth is unavailable. We propose a *self-probe* mechanism: instead of using a^*, the model generates its own best guess as the probe target.

1. **Generate probe tokens.** After each search, we append the probe prefix `<think>Now there's enough information to answer</think><answer>` to the current context and greedily decode k tokens as $\hat{a}^{\mathrm{self}}$.
2. **Compute log-probability change.** We compute the average log-probability of $\hat{a}^{\mathrm{self}}$ on both pre-search and post-search contexts:
$$\mathrm{IG}_{i,t}^{\mathrm{self}} = \log \pi_\theta(\hat{a}^{\mathrm{self}} \mid q, y_{i,\le t}, o_{i,\le t}) - \log \pi_\theta(\hat{a}^{\mathrm{self}} \mid q, y_{i,\le t-1}, o_{i,\le t-1}) \tag{4}$$
3. **Rollback decision.** The same rollback threshold τ (a hyperparameter controlling intervention sensitivity; default τ=1.0) is applied: if the relative confidence drop exceeds τ, the search is rolled back, following Eq.(5).

3.4 ROLLBACK MECHANISM

Motivation. In multi-turn retrieval-augmented reasoning, low-quality retrieved documents can persist in the context as misleading information, steering subsequent reasoning and search queries toward incorrect directions. While negative rewards penalize poor searches, they cannot eliminate this contamination. Our key insight is that trajectory-level intervention is more effective than reward-level feedback: rather than expecting the model to cope with low-quality content, we remove it directly.

Detection: when to roll back. After each search, we compute the information gain—using GT-based IG during training (§3.3.1) or self-probe IG during inference (§3.3.2)—and the relative confidence change:

$$\Delta_{i,t}^{\text{rel}} = \frac{\text{IG}_{i,t}}{|\log \pi_\theta(a^* \mid q, y_{i,\leq t-1}, o_{i,\leq t-1})|} \tag{5}$$

If $\Delta_{i,t}^{\text{rel}} < -\tau$, the model's confidence in the correct answer has dropped by more than τ relative to its baseline due to this search, and we classify it as low-quality.

Intervention: the three-step rollback. When a low-quality search is detected, we execute three actions:

1. **State restoration.** We revert both the generation context and the accumulated training sequence to a pre-search snapshot, physically removing the low-quality query, reasoning, and retrieved documents.
2. **Feedback injection.** We append a `<system_note>` block that explicitly lists all failed queries:

   ```
   <system_note>The following queries are unhelpful and will not return
   useful results: "Query1". Please avoid searching these queries and try
   to search another query.</system_note>
   ```

   This serves two purposes: (a) it tells the model *what not to search*, preventing repetition of the same bad query; (b) it creates a training signal—the model learns to associate `<system_note>` with the need to reformulate. When multiple consecutive rollbacks occur, failed queries *accumulate* in the note.
3. **Reactivation.** The sample re-enters the generation loop. The model now sees the original clean context plus the feedback note, and generates a new reasoning-and-search action.

Consecutive rollbacks per position are capped at $M{=}2$; if exhausted, the final search is accepted. The first search of each trajectory skips rollback, as query quality at turn 0 is inherently limited without prior context. The `<system_note>` remains in the sequence but is masked from loss, ensuring the model conditions on it without learning to generate it.

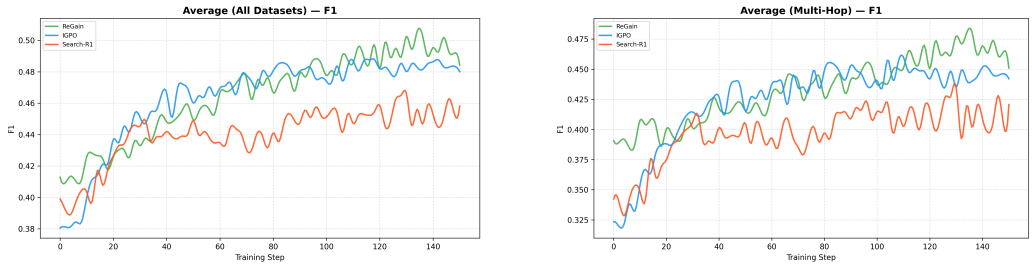
3.5 REWARD DESIGN AND NORMALIZATION

Token-level reward placement. IG rewards are placed at `</information>` token positions (weighted by w_{ig}), while the outcome reward (F1 score) is placed at the last token. All other positions receive zero reward.

Separate normalization. IG rewards and outcome rewards operate on vastly different scales (log-probability differences vs. F1 scores in $[0, 1]$). Joint normalization causes the larger IG values to dominate, effectively drowning out the outcome signal. We perform separate group-wise z-normalization within each GRPO group of G rollouts:

$$\tilde{r}_{i,t} = \begin{cases} w_{\text{ig}} \cdot \dfrac{r_{i,t}^{\text{IG}} - \mu^{\text{IG}}}{\sigma^{\text{IG}} + \varepsilon}, & 1 \leq t < T \\ \dfrac{r_i^{\text{O}} - \mu^{\text{O}}}{\sigma^{\text{O}} + \varepsilon}, & t = T \end{cases} \tag{6}$$

where t indexes the turn within a trajectory of T total turns, w_{ig} is the weight controlling the IG reward strength, $\mu^{\text{IG}}, \sigma^{\text{IG}}$ are the mean and standard deviation of all IG rewards $\{r_{i,t}^{\text{IG}}\}$ within the group, $\mu^{\text{O}}, \sigma^{\text{O}}$ correspond to the outcome rewards $\{r_i^{\text{O}}\}$, and ε is a small constant for numerical stability. This ensures both signal types contribute meaningfully to advantage estimation regardless of their original scales.



(a) Average F1 across all 7 datasets.

(b) Average F1 on multi-hop datasets.

Figure 3: Training curves (F1 score on the test set) over training steps for Qwen2.5-7B-Instruct. ReGain consistently outperforms IGPO and Search-R1, with the advantage most pronounced on multi-hop tasks.

Turn-level cumulative return. After normalization, we compute a turn-level discounted return that captures the impact of all future rewards:

$$\tilde{R}_{i,t} = \sum_{k=t}^{T} \gamma^{k-t} \tilde{r}_{i,k} \tag{7}$$

where γ is the discount factor and $\tilde{R}_{i,t}$ is the cumulative discounted return for turn t of the i-th rollout. During optimization, $\tilde{R}_{i,t}$ is assigned to all tokens produced in turn t of trajectory $\boldsymbol{\mathcal{T}}_i$. Good early searches receive high return because they contribute to all subsequent turns—this directly addresses the credit assignment problem.

3.6 State Masking

Tokens within <information></information> and <system_note></system_note> are assigned a loss mask $m_j = 0$; all other tokens have $m_j = 1$. The model only receives gradient updates on its own generated tokens (reasoning, queries, answers), not on retrieved documents or system feedback.

3.7 Training Objective

With the turn-level discounted return $\tilde{R}_{i,t}$ defined above, we optimize the policy using a clipped surrogate objective with KL regularization, following the same structure as GRPO but with finer-grained credit assignment and state masking (§3.6). Formally, our objective is:

$$J(\theta) = \mathbb{E}_{(q,a^*)\sim\mathcal{D},\,\{\boldsymbol{\mathcal{T}}_i\}\sim\pi_{\theta_{\text{old}}}(\cdot|q)} \Bigg[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|\boldsymbol{\mathcal{T}}_i|} \sum_{j=1}^{|\boldsymbol{\mathcal{T}}_i|} m_j \cdot \min\Big(\rho_{i,j}\, \tilde{R}_{i,t(j)},\ \text{clip}(\rho_{i,j}, 1-\epsilon, 1+\epsilon)\, \tilde{R}_{i,t(j)} \Big) \Bigg] - \beta\, D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}) \tag{8}$$

where $\mathcal{D}$ is a training dataset, $x_{i,j}$ denotes the j-th token in the flattened sequence of trajectory $\boldsymbol{\mathcal{T}}_i = (y_1, o_1, \ldots, y_T, o_T, a)$, $\rho_{i,j} = \pi_\theta(x_{i,j} \mid q, x_{i,<j}) / \pi_{\theta_{\text{old}}}(x_{i,j} \mid q, x_{i,<j})$ is the importance ratio, $t(j)$ maps token position j to the turn it belongs to so that $\tilde{R}_{i,t(j)}$ assigns the discounted return of that turn to each of its tokens, m_j is the loss mask (§3.6), ϵ is the clipping threshold, and β controls the KL penalty strength.

4 Experimental Setup

4.1 Datasets

We evaluate on seven QA benchmarks: **Multi-hop**: 2WikiMultiHopQA (Ho et al., 2020), HotpotQA (Yang et al., 2018), MuSiQue (Trivedi et al., 2022), Bamboogle (Press et al., 2023); **Single-hop**: Natural Questions (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023). We use the same training and test sets as IGPO.

Table 1: F1 scores (%) on seven QA benchmarks. Our experiments use identical conditions (E5 retriever, top-3, max 5 turns). **Bold** marks the best among comparable methods. Results for StepSearch, ReasoningRAG, GiGPO are copied from their original papers, following IGPO. Results for R1-Searcher and DeepResearcher are obtained by inference with their released model weights.

	Multi-Hop				Single-Hop				
Method	**2Wiki**	**Bamboogle**	**HotpotQA**	**MuSiQue**	**NQ**	**PopQA**	**TriviaQA**	**MH Avg.**	**Avg.**
Prompt-based methods									
CoT	29.9	41.2	30.5	12.6	24.2	21.8	55.0	28.6	30.7
CoT+RAG	22.1	39.5	42.7	15.1	39.0	45.8	70.3	29.9	39.2
Search-o1	39.7	45.5	46.4	19.9	29.5	35.0	63.2	37.9	39.9
Outcome-reward RL methods									
R1-Searcher	49.9	54.8	55.1	29.5	39.6	43.1	65.6	47.3	48.2
DeepResearcher	42.3	49.7	47.6	24.2	41.1	43.0	69.5	41.0	45.3
Search-R1	44.2	53.2	50.0	26.7	38.7	45.9	67.5	43.5	46.6
Process-reward RL methods									
StepSearch	43.1	53.4	50.2	**31.2**	–	–	–	44.5	44.5
ReasoningRAG	50.4	45.5	48.9	20.6	–	46.2	–	41.4	42.3
GiGPO	43.6	**68.9**	41.6	18.9	**46.4**	46.1	64.7	43.3	47.2
IGPO	49.1	52.0	51.5	26.7	41.9	47.8	**72.5**	44.8	48.8
ReGain	**54.7**	55.8	**53.9**	27.7	39.4	**52.4**	70.0	**48.0**	**50.6**

Table 2: F1 scores (%) across model scales. SFT denotes the supervised fine-tuned model before RL training. SFT$^-$ denotes SFT model inference without rollback. ReGain (w/o SFT) applies our RL method (IG reward + rollback) directly from the Instruct model without SFT cold start. ReGain$^-$ applies rollback only during training (no inference-time self-probe rollback).

	Multi-Hop				Single-Hop				
Method	**2Wiki**	**Bamboogle**	**HotpotQA**	**MuSiQue**	**NQ**	**PopQA**	**TriviaQA**	**MH Avg.**	**Avg.**
Qwen2.5-3B-Instruct									
Search-o1	30.8	38.8	34.9	12.4	27.0	35.8	58.4	29.2	34.0
SFT$^-$	39.8	38.3	39.3	15.4	29.7	35.7	54.2	33.2	36.1
SFT	40.3	38.8	39.9	15.4	30.0	35.6	54.8	33.6	36.4
Search-R1	37.2	38.8	44.2	16.6	38.6	42.7	62.5	34.2	40.1
ReGain (w/o SFT)	36.1	37.4	45.0	18.8	39.2	45.0	62.8	34.3	40.6
IGPO	40.5	44.9	40.4	18.4	**40.3**	41.6	62.4	36.1	41.2
ReGain$^-$	41.2	45.8	47.2	20.1	39.2	45.8	63.1	38.6	43.2
ReGain	**42.3**	**47.4**	**51.2**	**21.7**	**40.3**	**49.2**	**65.7**	**40.7**	**44.8**
Qwen2.5-7B-Instruct									
Search-o1	39.7	45.5	46.4	19.9	29.5	35.0	63.2	37.9	39.9
SFT$^-$	47.2	44.2	44.0	20.0	32.7	39.0	61.3	38.9	41.2
SFT	47.9	44.5	44.1	19.8	32.2	39.1	61.4	39.1	41.3
Search-R1	44.2	53.2	50.0	26.7	38.7	45.9	67.5	43.5	46.6
ReGain (w/o SFT)	45.0	52.8	50.5	26.4	39.5	46.2	69.3	43.7	47.1
IGPO	49.1	52.0	51.5	26.7	41.9	47.8	**72.5**	44.8	48.8
ReGain$^-$	51.0	53.0	**53.9**	26.6	**42.1**	48.7	70.6	46.1	49.4
ReGain	**54.7**	**55.8**	**53.9**	**27.7**	39.4	**52.4**	70.0	**48.0**	**50.6**

4.2 Implementation Details

All methods use Qwen2.5-7B-Instruct as the base model and the same retrieval infrastructure (E5-based dense retrieval, top-3 documents); ReGain additionally applies SFT cold start before RL. We train with GRPO on 8 H800 GPUs using max_turns=5, learning rate 1×10^{-6}, KL coefficient β=0.001, clip ratio ϵ=0.2. ReGain-specific parameters: IG threshold τ=1.0, max retries M=2, IG

Table 3: F1 scores (%) with different policy optimization algorithms on Qwen2.5-7B-Instruct. All methods use the same training data, and retrieval setup (E5 retriever, top-3, max 5 turns).

	Multi-Hop				Single-Hop				
Algorithm	**2Wiki**	**Bamboogle**	**HotpotQA**	**MuSiQue**	**NQ**	**PopQA**	**TriviaQA**	**MH Avg.**	**Avg.**
REINFORCE++	42.3	48.6	49.7	24.3	39.1	45.9	67.1	41.2	45.3
RLOO	38.8	54.7	49.4	23.9	40.9	45.2	67.9	41.7	45.8
GRPO	44.2	53.2	50.0	26.7	38.7	45.9	67.5	43.5	46.6
GSPO	42.9	53.0	50.4	23.4	41.6	47.1	70.1	42.4	46.9
PPO	43.0	53.5	50.7	26.8	**41.7**	44.8	**70.2**	43.5	47.2
ReGain (Ours)	**54.7**	**55.8**	**53.9**	**27.7**	39.4	**52.4**	70.0	**48.0**	**50.6**

reward weight w_{ig}=1.0, discount γ=1.0. All runs train for up to 150 steps with evaluation every 2 steps.

5 Results

5.1 Main Results

Table 1 presents F1 scores across all seven benchmarks under controlled conditions (E5 retriever, top-3 documents, max 5 search turns). ReGain achieves the highest average F1 (50.6%), outperforming both outcome-reward methods (Search-R1 46.6%, R1-Searcher 48.2%, DeepResearcher 45.3%) and process-reward methods (IGPO 48.8%, GiGPO 47.2%, StepSearch 44.5%).

Multi-hop advantages. ReGain's gains are most pronounced on multi-hop benchmarks, where each retrieval step must build on previous evidence—a single low-quality search can derail the entire reasoning chain. As shown in Table 1, ReGain achieves +3.2 MH avg F1 over IGPO on the 7B scale, confirming that rollback effectively prevents error propagation across hops where process rewards alone cannot. Table 2 further shows this advantage generalizes across model scales: on 3B, ReGain achieves 40.7% MH avg versus IGPO's 36.1% (+4.6), demonstrating that the rollback mechanism benefits both smaller and larger models.

5.2 Training Dynamics

Figure 3 shows the training curves over 150 steps. All three methods start from similar performance, but diverge after $\sim$20 steps.

On the overall average (Figure 3a), ReGain shows faster improvement and a higher plateau (peak 50.6%), while IGPO (48.8%) improves over Search-R1 (46.6%) but plateaus lower. The gap between IGPO and ReGain directly measures the benefit of active rollback intervention beyond reward-level IG feedback.

The advantage of ReGain is even more pronounced on multi-hop datasets (Figure 3b), where ReGain achieves 48.0% versus Search-R1's 43.5%. This confirms that search quality control is most impactful when questions require chaining evidence across multiple retrieval steps.

6 Analysis

6.1 Contribution of Each Component

To understand the contribution of each component, we note that SFT serves as a *format prerequisite* for rollback rather than an independent performance source—Table 2 shows the SFT checkpoint alone (41.3%) underperforms Search-R1 (46.6%) and only marginally improves over the prompt-based Search-o1 (39.9%), confirming that SFT's value lies in teaching the rollback format rather than directly improving search quality. The "ReGain (w/o SFT)" row further confirms this: applying rollback directly from the Instruct model yields only 47.1%, a marginal +0.5% over Search-R1, because the model cannot properly interpret `<system_note>` feedback without SFT exposure.

Given this prerequisite, the contribution chain is:

- **IG as process reward** (Search-R1 → IGPO): +2.2 F1 avg. Per-turn IG signals improve credit assignment over outcome-only training.
- **Active rollback** (IGPO → ReGain): +1.8 F1 avg. Rollback physically removes low-quality content from context, preventing cascading errors that negative rewards alone cannot address.

The largest single-dataset improvement is on 2WikiMultiHopQA (+10.5 F1 over Search-R1, +5.6 F1 over IGPO), where two-hop questions are especially vulnerable to one bad search poisoning the second hop. The mechanism behind this improvement—physically removing low-quality content rather than relying on the model to implicitly ignore it—is detailed in §3.4.

6.2 Rollback Threshold Sensitivity

We examine the effect of the rollback threshold τ (full results in Table 5, §A.2). Too aggressive (τ=0.5) discards partially useful retrievals, degrading performance to 48.82%; too conservative (τ=2.0) allows low-quality content to persist (49.76%). The default τ=1.0 achieves the best balance (50.56%).

6.3 Dataset-Specific Observations

Single-hop datasets (TriviaQA, NQ) show smaller gains—one good search typically suffices,leaving less room for rollback. MuSiQue remains challenging for all methods (max 27.7%), reflecting the difficulty of deeply compositional reasoning.

6.4 Effect of Policy Optimization Algorithm

Table 3 compares ReGain against alternative policy optimization algorithms under identical conditions (same training data, and retrieval setup). PPO and GRPO achieve similar overall performance (47.2% vs. 46.6% avg F1), indicating that the choice of base PO algorithm has limited impact without process-level reward signals. ReGain, which augments GRPO with IG-driven rollback and per-turn reward, outperforms all baseline PO algorithms on average and on every multi-hop dataset, with the largest gains on multi-hop tasks; on the single-hop NQ and TriviaQA it remains on par with the strongest baselines.

7 Conclusion

We presented ReGain, a closed-loop search quality control framework for training retrieval-augmented reasoning agents via reinforcement learning. Our approach uses information gain to detect low-quality searches and actively rolls back the generation state, replacing misleading content with explicit query reformulation feedback. This detect-intervene-learn loop addresses both the credit assignment and error propagation problems inherent in training. Experiments on seven QA benchmarks demonstrate consistent improvements over existing methods, with particularly strong gains on multi-hop reasoning tasks where search quality is critical.

Limitations

The rollback mechanism adds computational overhead to the generation loop: each rollback requires recomputing the IG probe on a restored context and generating a new search attempt, resulting in additional forward passes proportional to the rollback rate (18.6% at convergence). Measured on Qwen2.5-7B-Instruct under our main-experiment setup, this amounts to $+25.4\%$ training time (12.46 h → 15.63 h for 150 steps) and $+28.4\%$ end-to-end inference latency (1307 s → 1679 s), while the effect on the stored trajectory is negligible ($\approx$5.6 loss-masked tokens per turn). Strategies to amortize this cost, such as batched IG computation or adaptive probe frequency, are worth exploring. A full breakdown of the training, inference and token costs, together with the analytical estimates and the measurement protocol, is given in §A.3.

AI USE STATEMENT

We used GPT-5.2 to synthesize the rollback-aware trajectories that form the SFT cold-start training data described in §3.2. We have reviewed all AI-assisted work, and we take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *Proceedings of ICLR*, 2024.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. Improving language models by retrieving from trillions of tokens. In *Proceedings of ICML*, 2022.

Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z. Pan, Wen Zhang, Huajun Chen, Fan Yang, Zenan Zhou, and Weipeng Chen. Research: Learning to reason with search for llms via reinforcement learning. In *Proceedings of NeurIPS*, 2025.

Kenneth Ward Church and Patrick Hanks. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29, 1990.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, Hoboken, NJ, 2 edition, 2006.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Nature*, 645:633–638, 2025.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. In *Proceedings of NeurIPS*, 2025.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of COLING*, 2020.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43, 2023.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Ö. Arık, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. In *Proceedings of COLM*, 2025.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of ACL*, 2017.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. In *Transactions of the ACL*, 2019.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of NeurIPS*, 2020.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of ACL*, 2023.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of EMNLP*, 2023.

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

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. In *Proceedings of NeurIPS*, 2024.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*, 2025.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single hop question composition. In *Transactions of the ACL*, 2022.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of ACL*, 2023.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, and Zhenzhe Ying. Information gain-based policy optimization: A simple and effective approach for multi-turn search agents. In *Proceedings of ICLR*, 2026.

Ziliang Wang, Xuhui Zheng, Kang An, Cijun Ouyang, Jialu Cai, Yuhang Wang, and Yichao Wu. Stepsearch: Igniting llms search ability via step-wise proximal policy optimization. *arXiv preprint arXiv:2505.15107*, 2025.

Tongyu Wen, Guanting Dong, and Zhicheng Dou. Smartsearch: Process reward-guided query refinement for search agents. In *Proceedings of SIGIR*, 2026.

Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. Corrective retrieval augmented generation. *arXiv preprint arXiv:2401.15884*, 2024.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of EMNLP*, 2018.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. *arXiv preprint arXiv:2504.03160*, 2025.

A Appendix

A.1 Hyperparameters

Table 4: Full hyperparameter configuration for both training phases.

Parameter	Value
SFT Cold Start	
Base model	Qwen2.5-7B-Instruct
Teacher model	GPT-5.2
Training samples	18,246
Epochs	4
Learning rate	5×10^{-5} (cosine)
Effective batch size	128
Max sequence length	16,384
RL	
Max search turns	5
Learning rate	1×10^{-6}
KL coefficient	0.001
clip ratio	0.2
IG threshold (τ)	1.0
Max rollback retries (M)	2
IG reward weight (w_{ig})	1.0
Self-probe tokens (k)	5
Discount (γ)	1.0
Retriever	E5, top-3
Training steps	150

A.2 Rollback Threshold Ablation

Table 5 reports the full sweep over the rollback threshold τ, together with the resulting rollback rate during training. Performance is stable for $\tau \in [0.75, 1.25]$ and degrades at both extremes, with τ=1.0 giving the best overall trade-off.

Table 5: Ablation on rollback threshold τ. MH Avg. and Avg. are F1 scores (%) on multi-hop and all datasets respectively. Rollback Rate is the fraction of search actions rolled back during training. τ=1.0 achieves the best overall performance with a moderate rollback rate.

τ	MH Avg.	Avg.	Rollback Rate
0.5	46.04	48.82	18.15%
0.75	47.85	49.92	18.24%
1.0	**48.02**	**50.56**	18.62%
1.25	47.91	50.38	18.79%
1.5	47.28	50.07	17.93%
2.0	47.04	49.76	16.97%

A.3 Computational Cost Analysis

All timings use Qwen2.5-7B-Instruct with the retrieval setup of our main experiments (E5, top-3 documents, max 5 turns) and the same hardware for every configuration, so "w/o rollback" differs only in whether the intervention is enabled.

Training overhead. Analytically, each rollback adds one forward pass, so at the measured per-turn rollback rate of 18.6% (Table 5) the predicted overhead relative to a no-rollback rollout is $\approx$18.6%. Empirically, 150 training steps take 12.46 hours without rollback and 15.63 hours with rollback (+25.4%). The empirical figure slightly exceeds the estimate because each rollback also issues an additional retriever call and an IG probe computation on the restored context, and GRPO's group rollouts are gated by the slowest sample within a batch.

Inference overhead. Analytically, each rollback adds $\sim$1 turn-equivalent ($\approx$18.6% at the measured rate) and the self-probe additionally decodes k greedy tokens per turn ($< 3\%$ per turn), for $\approx$21–22% over a no-rollback multi-turn baseline. Empirically, end-to-end inference over our test sets (3,197 samples) takes 1307 s without rollback and 1679 s with rollback ($+28.4\%$); the $\sim$3 percentage points above the training overhead come from the self-probe, whose probe target must be generated at inference whereas training reuses the ground-truth answer. This price is modest in context: as Table 2 shows, ReGain$^-$ already outperforms all other methods on average without inference-time rollback, and unlike approaches that depend on a separately trained external model for supervision, our overhead is only k extra output tokens.

Token budget. From the trajectory's perspective the impact is negligible: a rollback deletes the failed query together with its retrieved evidence and inserts a short <system_note> ($\sim$30 tokens) in their place, so the final training sequence grows by only $18.6\% \times 30 \approx 5.6$ tokens per turn on average, and the note is loss-masked (§3.6). From the total generation cost perspective, the $\sim$18.6% of tokens emitted before the rollback decision cannot be reclaimed—but that is not an additional cost, it is precisely what the training and inference figures above already measure.

A.4 System Prompt

The following prompt is used for both SFT data generation and RL training/inference. The model receives this as the system message, with {question} replaced by the actual question.

> Answer the given question. You must follow a strict iterative protocol where every response follows one of the two allowed output patterns.
>
> **Output Patterns:**
> Search Pattern: <think>...</think><search>Query</search>
> Answer Pattern: <think>...</think><answer>Concise result</answer>
>
> **Protocol Instructions:**
> Reasoning First: Every time you receive new information (from <information> search results or a <system_note>), you must first conduct reasoning inside <think> and </think>.
> Knowledge Gap: After reasoning, if you find you lack some knowledge, you can call a search engine using <search>query</search>. The system will return results between <information> and </information>.
> Search Limitations: You can search as many times as you want.
> System Feedback: If a <system_note> is provided (e.g., instructing you not to search certain queries and try to search another query), you must incorporate this constraint into your next <think> block.
> Final Output: If you find no further external knowledge is needed, provide the answer inside <answer> and </answer> without detailed illustrations (e.g., <answer>1234</answer>).
>
> **Strict Constraint:**
> Your response must contain NOTHING outside the tags. Each turn must be either a <think>/<search> pair or a <think>/<answer> pair.
>
> Question: {question}

The <system_note> mentioned in the prompt is injected by our rollback mechanism (§3.4) when a low-quality search is detected. During normal operation (no rollback), the model never sees a <system_note>. The prompt explicitly instructs the model to incorporate such feedback, ensuring it knows to reformulate its query upon receiving rollback guidance.

B Algorithm Pseudocode

C Inference-Time Self-Probe

C.1 Does the Self-Probe Reinforce Wrong Beliefs?

Because the self-probe (§3.3.2) uses the model's own greedy prediction as the probe target, a natural concern is that it merely measures self-consistency and could therefore entrench an already-wrong

Algorithm 1 ReGain Training Loop (per batch)

Require: Batch $\{(q_i, a_i^*)\}$, model π_θ, retriever $\mathcal{R}$
Compute baselines: $\log \pi_\theta(a^* \mid q)$ for all samples
for $t = 1$ to T **do**
  Generate reasoning + query: $y_t \sim \pi_\theta(\cdot \mid q, y_{<t}, o_{<t})$
  Retrieve: $o_t \leftarrow \mathcal{R}(\text{query}_t)$
  Compute IG: $\text{IG}_t \leftarrow \log \pi_\theta(a^* \mid q, y_{\le t}, o_{\le t}) - \log \pi_\theta(a^* \mid q, y_{\le t-1}, o_{\le t-1})$
  Compute relative change: $\Delta_t^{\text{rel}} \leftarrow \text{IG}_t / |\log \pi_\theta(a^* \mid q, y_{\le t-1}, o_{\le t-1})|$
  if $\Delta_t^{\text{rel}} < -\tau$ **and** retries $< M$ **and** not first turn **then**
    Rollback: discard y_t, o_t; restore to $(y_{\le t-1}, o_{\le t-1})$
    Inject: append <system_note> with failed queries
    Increment retry count; **continue**
  end if
  Record $r_t^{\text{turn}} \leftarrow w_{\text{ig}} \cdot \text{IG}_t$
end for
Compute outcome: $R^{\text{O}} \leftarrow \text{F1}(\hat{a}, a^*)$
Place rewards at token positions (§3.5)
Compute advantages via separate norm + turn-level accumulation
Update π_θ via dual-clip PPO with state masking

Table 6: GT-based vs. self-probe rollback at inference time on Qwen2.5-7B-Instruct (F1, %). Both rows use the *same* trained ReGain policy and identical inference conditions (E5 retriever, top-3, max 5 turns, τ=1.0); they differ only in the probe target used by the rollback trigger—the ground-truth answer (an oracle, not available in deployment) vs. the model's own k greedy tokens. The two probes behave almost identically (Δ Avg. $= 0.1$, Δ MH Avg. $= 0.3$), and the self-probe is even better on 3 of the 7 datasets, indicating that self-probing does not systematically entrench wrong beliefs.

	Multi-Hop				Single-Hop				
Rollback Probe	**2Wiki**	**Bamboogle**	**HotpotQA**	**MuSiQue**	**NQ**	**PopQA**	**TriviaQA**	**MH Avg.**	**Avg.**
GT-based (oracle)	53.4	55.8	**56.3**	**27.9**	**40.0**	51.9	69.8	**48.3**	**50.7**
Self-Probe (ours)	**54.7**	55.8	53.9	27.7	39.4	**52.4**	**70.0**	48.0	50.6

belief: a retrieval that contradicts a wrong prediction would lower the probe log-probability and be rolled back. We address this concern empirically and analytically.

Empirical comparison. Table 6 compares GT-based rollback (using the true answer as the probe target, an oracle unavailable in deployment) against self-probe rollback on the full test set under identical inference conditions, using the same trained ReGain policy. The two probes behave almost identically (Δ Avg. $= 0.1$, Δ MH Avg. $= 0.3$), and the self-probe is in fact better on 3 of the 7 datasets. Replacing the oracle probe with the model's own prediction therefore costs essentially nothing, which is not what one would expect if self-probing systematically reinforced wrong answers.

Table 2 provides a second, independent check. The "ReGain$^-$" row keeps rollback during training but disables self-probe rollback at inference, whereas "ReGain" enables both. Enabling inference-time self-probe rollback improves average F1 at both scales ($43.2 \rightarrow 44.8$ on 3B, $49.4 \rightarrow 50.6$ on 7B). If the self-probe were entrenching wrong beliefs, we would expect the opposite.

Why the self-probe cannot make things worse. The decision rule fires only when the probe log-probability drops sharply after retrieval, which yields two cases. If the current prediction is already correct, a sharp confidence drop can only be caused by genuinely misleading retrieved content, so rolling back is the correct action. If the current prediction is wrong and the retrieval reinforces that same wrong belief, the probe log-probability does not drop and the rule does not fire; the behavior degenerates to the no-rollback baseline. In other words, the self-probe can recover a trajectory that rollback would help, and in the adversarial case it simply abstains rather than actively degrading the prediction. The residual limitation is that the self-probe cannot *detect* a wrong belief that retrieval happens to confirm—it is a trigger for removing harmful context, not a correctness oracle.

C.2 Worked Example

Inference: Self-Probe Rollback (GT-Free)

Step 1: Generate self-probe after retrieval
Append `<think>Now there's enough information to answer</think><answer>` to context, greedily decode k tokens $\hat{a}^{\text{self}}$.

Step 2: Compute log-prob change on self-probe
$\log \pi_\theta(\hat{a}^{\text{self}} \mid q, y_{\le t-1}, o_{\le t-1}) = -3.1$ (confident before search)
$\log \pi_\theta(\hat{a}^{\text{self}} \mid q, y_{\le t}, o_{\le t}) = -11.4$ (confused after bad retrieval)

Step 3: Rollback decision
$\text{IG}^{\text{self}} = -11.4 - (-3.1) = -8.3$
$\Delta^{\text{rel}} = -8.3/3.1 = -268\%$
$\Delta^{\text{rel}} < -\tau = -1.0$: **ROLLBACK**

Step 4: Restore state & retry (same 3-step as training)
Cost: one generate (k tokens) + one forward pass per turn.

Figure 4: Inference-time rollback via self-probe. Without ground truth, the model generates k greedy tokens as its "best guess," then checks whether retrieval increases or decreases confidence in that guess. A large drop triggers rollback—no labels needed.