

Observation-Aware Stabilization of Reinforcement Learning for Retrieval-Augmented Generation

Naixin Zhang and Tong Lin*

State Key Laboratory of General Artificial Intelligence,
School of Intelligence Science and Technology,
Peking University, Beijing 100871, China
zhangnaixin@stu.pku.edu.cn lintong@pku.edu.cn

*Correspondence: lintong@pku.edu.cn

Abstract

Reinforcement learning (RL) for retrieval-augmented language models enables policies to interleave generated reasoning with retrieved observations. However, in Search-R1-style PPO, training may become unstable after the model has already reached strong validation performance. This paper studies late-stage collapse as a training-stability problem in search-based RL. Four metrics are proposed to quantify validation drawdown (VDD), advantage-tail expansion (ATE), actor update stress (AUS), and observation-to-actor contamination in generalized advantage estimation (GAE), denoted as OAGC. The analysis further shows how observation tokens can influence actor updates through value targets, value-model learning, and GAE. To address this issue, Observation-aware Stabilization (OAS) combines observation residual decay, old-value anchoring, and boundary smoothing. On three benchmark datasets (NQ, TriviaQA, and HotpotQA), OAS improves Search-R1 PPO under multiple retrieval budgets. In the main Top-3 setting, our method improves the exact match (EM) measure from 31.8/49.6/18.7 to 37.6/56.8/23.9 and reduces OAGC from 0.331 to 0.088, with lower search-budget saturation and no increase in the no-search rate.

1 Introduction

Large language models are increasingly used as reasoning agents that interact with external environments. In knowledge-intensive question answering, retrieval-augmented generation (RAG) retrieves external evidence before generation (Lewis et al., 2020), but a single-pass pipeline is often insufficient for multi-hop evidence gathering, query reformulation, and intermediate verification (Gao et al., 2023). Recent work on agentic RAG and tool-use RL therefore invokes search tools during reasoning, including ReAct (Yao et al., 2023) and Search-R1 (Jin et al., 2025). These methods model question

answering as a multi-step trajectory that interleaves reasoning, actions, and observations (Xiong et al., 2025; Qian et al., 2025; Zhang et al., 2025).

This paradigm makes reinforcement learning a natural framework for optimizing search-tool behavior from outcome-level rewards or process-level feedback, unlike supervised fine-tuning on fixed human-annotated or distilled trajectories. RL can teach models when to search, how to reformulate queries, how to use retrieved evidence, and when to stop (Jin et al., 2025). In Search-R1-style training, the search engine is incorporated into the environment, and interactive search-reasoning trajectories can be optimized with PPO (Schulman et al., 2017; Jin et al., 2025) or GRPO (Shao et al., 2024; Jin et al., 2025).

However, interactive search reasoning can also introduce severe stability problems. Prior work on multi-turn agent RL and LLM RL has reported reward collapse, entropy collapse, policy staleness, and gradient spikes in long-sequence or multi-turn environments (Wang et al., 2025; Zheng et al., 2025; Dong et al., 2026a). These findings suggest that search-tool RL should be evaluated not only by final answer accuracy but also by whether model behavior and optimization signals remain stable during training.

Our reproduction experiments show that this stability cannot be guaranteed. We observe a late-stage failure mode distinct from failing to learn: the model reaches a reasonable intermediate validation score, but later suffers a validation drawdown while both tool use and answer generation degrade. Final scores cannot distinguish such late-stage collapse from consistently low-scoring runs.

We analyze this late-stage collapse problem through the token structure of search trajectories. A single sequence contains both generated tokens and observation tokens. Generated tokens are actions sampled by the actor. In contrast, observation tokens are environmental feedback returned

by the retriever. Although they remain in the context, observation tokens should not directly contribute to the actor loss because they are external feedback rather than actions sampled by the policy. Throughout the paper, observation tokens denote retriever-returned text, observation residuals denote their temporal-difference (TD) residual terms, and generated-token advantages denote advantages at actor-loss positions. Existing implementations typically mask observation tokens out of the actor loss (Jin et al., 2025; Chai et al., 2025), but they rarely explain their role in value fitting, TD residuals, generalized advantage estimation (GAE), and advantage propagation (Schulman et al., 2016, 2017). This paper therefore asks whether the current treatment can transmit observation-side value errors or TD residuals into generated-token advantages, thereby affecting late-stage stability.

We therefore examine training stability through token-level signals rather than final scores alone. Our analysis first defines late-stage collapse as a validation drawdown after an earlier performance peak. It then asks whether this collapse is accompanied by systematic gaps between generated and observation tokens in actor-critic signals. We further ask whether these signals can guide a stabilization method that preserves observations as context while reducing their interference with generated-token policy updates.

To answer these questions, we use four diagnostic metrics for training dynamics: validation drawdown (VDD), advantage tail expansion (ATE), actor update stress (AUS), and observation-to-actor GAE contamination (OAGC). These metrics summarize performance drawdown, advantage-tail expansion, actor update pressure, and observation-to-actor contamination. Guided by these diagnostics, we trace how observation-side residuals can affect generated-token advantages through value-target construction, value-model updates, and GAE. This motivates Observation-aware Stabilization (OAS), which down-weights observation-side TD residuals, anchors value drift around observation spans, and smooths abrupt boundary value changes while keeping observation tokens available as context.

Our contributions are summarized as follows:

1. We introduce a diagnostic suite of VDD, ATE, AUS, and OAGC. These metrics characterize late-stage collapse from four perspectives: performance drawdown, advantage tails, actor update pressure, and observation-to-actor

GAE contamination.

2. We conduct a token-level mechanism analysis that separates value, return, and advantage signals for generated and observation tokens. We use OAGC and domain gaps to trace how observation-side errors enter generated-token advantages.
3. We propose Observation-aware Stabilization (OAS), which stabilizes search-tool PPO through observation residual decay, old-value anchoring, and boundary smoothing. Main results and ablations show that OAS improves exact match and reduces nVDD, ATE, and OAGC. The time-aligned analysis further connects ATE, AUS, and OAGC with validation drawdown and shows that OAS maintains more stable late-stage training dynamics than PPO.

2 Related Work

2.1 Retrieval-Augmented and Tool-Augmented Language Agents

Retrieval-augmented generation (RAG) uses external knowledge to mitigate outdated knowledge, factual hallucination, and weak evidence traceability (Lewis et al., 2020; Gao et al., 2023). Early RAG systems typically retrieve once before generation; this is effective for knowledge-intensive QA but limited by retrieval noise and fixed workflows in multi-hop reasoning. ReAct (Yao et al., 2023) interleaves reasoning traces, actions, and observations, moving retrieval from static preprocessing to a tool action within the reasoning trajectory.

Search-augmented reasoning systems let models decide when to search and how to form queries. Search-R1 (Jin et al., 2025) incorporates search into the RL environment, where model-generated `<search>...</search>` spans trigger retrieval and returned evidence is appended as `<information>...</information>`. RAG-Gym (Xiong et al., 2025) emphasizes process supervision and critic evaluation for agentic RAG, while RLFactory (Chai et al., 2025) implements a generate-parse-invoke-update loop with observation markers. These works define the setting studied here: the model generates search actions, receives observations, and continues reasoning before answering.

2.2 Reinforcement Learning for Language Models and Tool-Use Rewards

Reinforcement learning is now central to language-model post-training. RLHF (Ouyang et al., 2022), PPO (Schulman et al., 2017), and GAE (Schulman et al., 2016) provide foundations for instruction following and actor-critic training, while DeepSeek-R1 (Guo et al., 2025) and DeepSeekMath (Shao et al., 2024) show the role of outcome rewards and GRPO-style methods in long-chain reasoning. Search-tool RL adds tool timing, query formulation, evidence use, and stopping decisions to this setting.

Tool-use tasks therefore make reward design and credit assignment harder. ToolRL (Qian et al., 2025) argues that rewards should cover intermediate decisions such as tool selection, execution, and result use, not only final answer matching. ReasonRAG (Zhang et al., 2025) decomposes agentic RAG supervision into query generation, evidence extraction, and answer generation, aiming to reduce outcome-only sparsity. Turn-level credit assignment work (Zeng et al., 2025) further shows that sharing one trajectory-level advantage makes different search turns difficult to attribute.

2.3 Training Instability in Multi-Turn Agent Reinforcement Learning

Instability in LLM reinforcement-learning dynamics is a recurring issue. Prior work reports entropy collapse, reward variance collapse, policy staleness, gradient spikes, catastrophic updates, and advantage bias (Wang et al., 2025; Zheng et al., 2025; Dong et al., 2026a). RAGEN (Wang et al., 2025) observes Echo Trap in multi-turn agent RL, where local reward overfitting coincides with reward variance cliffs, entropy drops, and gradient spikes. More general LLM RL stability work links token-level objectives to training-inference discrepancy and policy staleness (Zheng et al., 2025).

Tool-use settings further amplify these issues. Search-R1 (Jin et al., 2025) and RLFactory (Chai et al., 2025) feed external results back as observations and mask retrieved/tool-return tokens out of the policy loss, marking tool feedback as different from model-generated text. ARPO (Dong et al., 2026b) and AEPO (Dong et al., 2026a) show that such feedback changes entropy, rollout sampling, and policy updates; SimpleTIR (Xue et al., 2025) links performance collapse to feedback-induced distributional drift and gradient norm explosion.

Understanding Tool-Integrated Reasoning (Lin and Xu, 2025) further shows that reward shaping can weaken main-task correctness signals under group-normalized advantages. Together, these works show that tool calls change the feedback distribution, advantage attribution, and gradient propagation on which model updates depend.

Overall, existing work shows that tool feedback changes trajectory distributions, exploration, and optimization signals, but less is known about how observation tokens affect search-RL stability. Retrieved results both alter subsequent generation as context and receive special training treatment, such as exclusion from actor loss. They may still affect value estimation, GAE, or advantage computation. This gap motivates our diagnostic focus on VDD, ATE, AUS, OAGC, and token-domain value/advantage gaps.

3 Trajectory-Level Instability Metrics

The failures in search-tool RL do not always mean that the model fails to learn from the start. In PPO training of Search-R1-style methods, we observe a more subtle dynamic: the model first reaches a useful validation score in the middle of training, but later regresses, accompanied by degraded tool use and abnormally long outputs. To distinguish this phenomenon from low-performing runs that never reach useful validation performance, we formalize it as trajectory-level optimization instability. We characterize it with four metrics: validation drawdown (VDD), advantage tail expansion (ATE), actor update stress (AUS), and observation-to-actor GAE contamination (OAGC). Full symbol definitions and construction details are provided in Appendix A.

Let y_t denote the validation score around training step t . VDD measures how far the current validation score has dropped from its historical best:

$$\begin{aligned} \text{VBest}_t &= \max_{s \leq t} y_s, \\ \text{VDD}_t &= \text{VBest}_t - y_t, \\ \text{nVDD}_t &= \frac{\text{VDD}_t}{|\text{VBest}_t| + \epsilon}. \end{aligned} \tag{1}$$

VDD serves as the outcome-side reference metric for late-stage collapse; nVDD_t makes drawdown magnitudes comparable across runs.

ATE and AUS characterize process-side abnormalities associated with the drawdown. ATE is computed from the advantage range over generated

tokens that participate in actor updates:

$$\begin{aligned} \text{AdvRange}_t &= \max_i \hat{A}_{t,i} - \min_i \hat{A}_{t,i}, \\ \text{ATE}_t &= \text{robust_z}_+(\text{AdvRange}_t). \end{aligned} \quad (2)$$

Here i denotes the flattened token-sample index over all generated tokens in the current batch that participate in the actor update. ATE indicates whether the advantage distribution used to weight actor updates exhibits abnormal tail expansion. AUS measures the stress induced by a complete actor update:

$$\begin{aligned} \text{AUS}_t &= \frac{1}{2} \text{robust_z}_+^{\text{shared}}(|\text{PostKL}_t|) \\ &+ \frac{1}{2} \text{robust_z}_+^{\text{shared}}(\text{PostClipFrac}_t). \end{aligned} \quad (3)$$

For each training step, KL and ClipFrac are measured on the same frozen actor-loss batch immediately before and after the complete actor update, including all mini-batches, PPO epochs, and optimizer steps. The PPO and OAS trajectories use one shared median/MAD reference constructed from all valid steps, placing the two methods on the same scale.

In search-tool tasks, elevated AUS indicates that a complete actor update has entered a high-deviation, high-clipping regime. Such update stress can affect ordinary next-token generation as well as search timing, query content, retrieved observations, and subsequent context.

OAGC captures the contribution of observation tokens to generated-token advantages. Let D_π be the set of generated-token positions that participate in actor loss, and let D_{obs} be the set of observation-token positions returned by the retriever. We first define the observation-side component of a generated-token advantage by retaining only future observation-token TD residuals, and then normalize its absolute mass by the total generated-token advantage mass:

$$\begin{aligned} A_t^{\text{obs}} &= \sum_{\ell \geq 0, t+\ell \in D_{\text{obs}}} (\gamma\lambda)^\ell \delta_{t+\ell}, \\ \text{OAGC} &= \frac{\sum_{t \in D_\pi} |A_t^{\text{obs}}|}{\sum_{t \in D_\pi} |A_t| + \epsilon}. \end{aligned} \quad (4)$$

Higher OAGC means that a larger fraction of generated-token advantage comes from observation-side residuals. We next examine whether ATE, AUS, and OAGC are associated with VDD. Across multiple groups of NQ runs using

Metric	Current VDD	Future VDD (10 / 20 / 30 steps)
ATE	0.636	0.614 / 0.562 / 0.462
AUS	0.683	0.598 / 0.559 / 0.572
OAGC	0.594	0.472 / 0.450 / 0.344

Table 1: Step-level pooled Pearson correlations between process metrics and validation drawdown.

Qwen2.5-3B-Instruct and Search-R1-style PPO, we compute step-level pooled Pearson correlations; future VDD corresponds to VDD after 10, 20, and 30 training steps.

As shown in Table 1, ATE, AUS, and OAGC are all positively correlated with current VDD and remain positively correlated with VDD at 10- to 30-step horizons. ATE and AUS have the strongest current correlations, while OAGC also shows a positive relationship with current drawdown and a weaker but persistent relationship with future drawdown. This supports using the three process metrics as complementary diagnostics for VDD-related instability.

4 Stability Analysis

Section 3 formalizes late-stage collapse as a training-dynamics phenomenon. We next analyze the following question in search-tool PPO: even though observation tokens should not receive actor-loss updates, why can they affect generated-token policy updates through the value-model update, TD residuals, and GAE?

A search-tool trajectory can be formulated as:

$$z_i = (x_i, G_{i,1}, O_{i,1}, G_{i,2}, O_{i,2}, \dots, G_{i,K_i}, y_i), \quad (5)$$

where $G_{i,k}$ is a model-generated reasoning, query, or answer token span, and $O_{i,k}$ is an observation span returned by the retriever. Existing Search-R1-style search-tool training keeps $O_{i,k}$ in the sequence so that later generated tokens can read it. It masks observation tokens in the actor loss, while return targets, TD residuals, GAE, and value-model updates may still be computed over sequences that include them. Figure 1(b) previews how this active value/GAE path can propagate observation-side signals to generated-token actor updates; Propositions 1–3 formalize its three stages.

Let $\mathcal{G}$ denote all generated-token positions that participate in actor loss. The PPO actor objective

is formulated as:

$$L_\pi(\theta) = \frac{1}{|\mathcal{G}|} \sum_{(i,t) \in \mathcal{G}} \min \left(\rho_{i,t}(\theta) \hat{A}_{i,t}, \right. \\ \left. \text{clip}(\rho_{i,t}(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_{i,t} \right). \quad (6)$$

The relevant quantity is therefore not the direct inclusion of observation tokens in the actor loss, but the construction of $\hat{A}_{i,t}$ by the value model and GAE. If the GAE residual chain still passes through observation positions, then:

$$\hat{A}_{i,t}^V = \sum_{\ell=0}^{T_i-t} (\gamma\lambda)^\ell m_{i,t+\ell}^V \delta_{i,t+\ell}. \quad (7)$$

Full derivations of the three observation-to-actor propagation stages are given in Appendix B.1–B.3.

Proposition 1 (Return/value target mismatch).

If GAE-based target construction still passes through observation positions, then the actual token-level target for a generated token differs systematically from a reference target. This reference target recurses only over generated-token decision points. Define the reference return target $\hat{R}_{i,j}^*$ and subtract it from the actual token-level target:

$$E_{i,j}^R = \hat{R}_{i,j}^V - \hat{R}_{i,j}^*. \quad (8)$$

This mismatch decomposes into three terms:

$$E_{i,j}^R = \underbrace{S_{i,j}^\lambda}_{\text{trace-decay mismatch}} + \underbrace{S_{i,j}^{\text{boot}}}_{\text{bootstrap-state mismatch}} \\ + \underbrace{C_{i,j}^{\text{obs}}}_{\text{observation residual contribution}}. \quad (9)$$

Here, S^λ captures the additional trace decay across observation spans, S^{boot} captures the change from token-level to generated-token-level bootstrap states, and C^{obs} captures observation-token TD residuals entering earlier generated-token targets. Appendix B.1 provides the complete decomposition and minimal-rollout cases.

Proposition 2 (Value model update propagation).

The target mismatch in Proposition 1 enters the value-model update and changes the gradient direction of generated-token value learning. When observation positions participate in the same value loss, generated-token supervision is rescaled by Z_G/Z_V , while E^R perturbs the generated-token gradient.

The first effect reduces the relative supervision weight assigned to generated-token decision points, and the second shifts their fitting targets from $\hat{R}^*$ toward the actual token-level target $\hat{R}^V$. Because the value model shares parameters and representations across positions, observation-side supervision can also alter the value baselines of nearby generated-token states. Appendix B.2 gives the complete gradient decomposition and propagation derivation.

Proposition 3 (Actor update propagation).

Observation-side signals in the value-model / GAE path enter the actor update by changing the advantage weights of generated-token policy gradients. The generated-token advantage can be decomposed as:

$$\hat{A}_{i,t} = A_{i,t}^G + A_{i,t}^O, \quad (i, t) \in \mathcal{G}, \quad (10)$$

Here, $A_{i,t}^O$ is the future observation-residual contribution that reaches the current generated token through GAE recursion. This is the theoretical counterpart of OAGC in Eq. (3.4): as the aggregate absolute mass of A^O increases relative to the total generated-token advantage $\hat{A}$, OAGC increases. Appendix B.3 gives the complete expansion.

In summary, observation-side biases propagate from target construction through value-model learning to generated-token actor updates, where their advantage contribution is measured by OAGC. This three-stage mechanism identifies one source of downstream bias propagation and motivates the OAS interventions introduced in the next section.

5 Observation-aware Stabilization

OAS is a mechanism-level stabilization method for Search-R1-style PPO. Section 4 traces a mechanism by which observation-token value errors and TD residuals can enter generated-token advantages and amplify instability. OAS therefore keeps observations as state information. It constrains how they enter value fitting, GAE, and boundary value transitions. Figure 1 summarizes this propagation path and locates the three OAS components within it.

Let $m_{i,t}^{V,\pi}$ and $m_{i,t}^{V,\text{obs}}$ denote generated-token and observation-token positions in the value/GAE

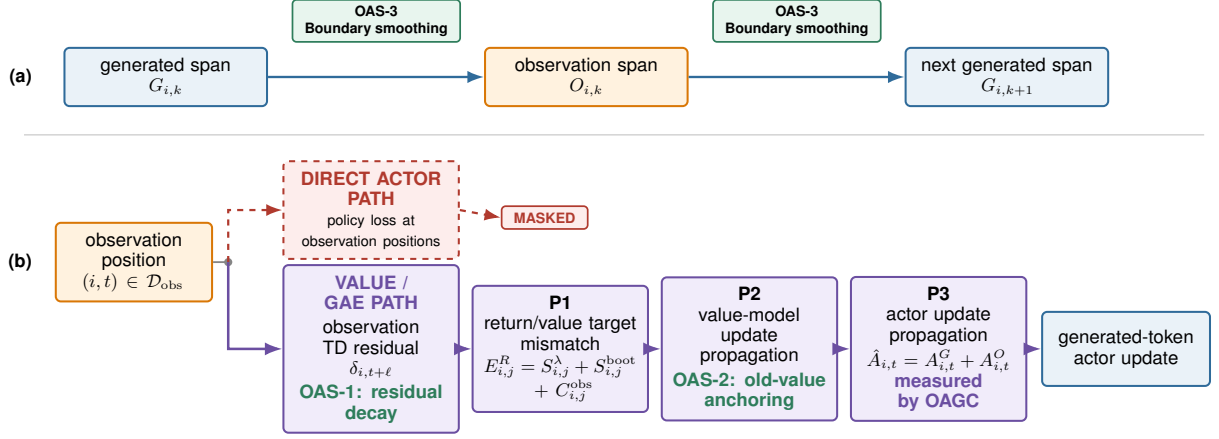


Figure 1: Observation-to-actor influence and OAS intervention points. (a) A generated span produces an observation that informs the next generated span. (b) Although its direct actor path is masked, the observation still reaches generated-token actor updates along the value/GAE path, through return/value target mismatch (P1), value-model update propagation (P2), and advantage propagation (P3). P1–P3 denote Propositions 1–3; OAS-1–3 denote residual decay, old-value anchoring, and boundary smoothing.

Method	Top- $k = 2$			Top- $k = 3$			Top- $k = 5$		
	NQ	TriviaQA	HotpotQA	NQ	TriviaQA	HotpotQA	NQ	TriviaQA	HotpotQA
Search-R1 PPO	30.6 (.86)	48.8 (2.43)	17.9 (.16)	31.8 (.44)	49.6 (.78)	18.7 (.99)	31.2 (.13)	<u>53.7</u> (.31)	19.1 (.24)
Search-R1 GRPO	29.0 (.28)	47.1 (.34)	16.3 (.19)	29.7 (.48)	47.8 (.25)	17.1 (.41)	28.5 (.13)	46.4 (.59)	17.5 (.47)
DAPO	34.1 (.19)	50.2 (.87)	18.7 (.12)	34.2 (1.66)	51.0 (.19)	19.4 (.23)	33.0 (.86)	50.0 (.43)	20.0 (.65)
KL-Cov	32.1 (.92)	49.6 (.79)	17.6 (.43)	32.6 (.25)	<u>52.7</u> (.55)	18.2 (.94)	31.8 (.34)	49.4 (.98)	18.8 (.11)
DAR	32.8 (.23)	50.3 (.46)	19.0 (.33)	33.5 (.36)	51.2 (.93)	19.8 (.22)	32.4 (.49)	50.2 (1.02)	<u>21.6</u> (.14)
I-PPO	31.3 (.29)	49.1 (1.09)	17.8 (.26)	31.9 (.37)	49.8 (.32)	18.5 (.78)	30.8 (.57)	48.6 (.41)	18.9 (.63)
Ours (OVR only)	30.7 (.16)	<u>51.6</u> (.70)	<u>19.9</u> (.61)	<u>34.8</u> (.86)	50.4 (.17)	<u>20.8</u> (.30)	<u>33.7</u> (.61)	51.5 (.55)	20.3 (.88)
Ours (OAS)	<u>33.5</u> (.58)	55.6 (.50)	22.7 (.97)	37.6 (.83)	56.8 (.49)	23.9 (.81)	36.4 (.41)	55.1 (.71)	24.5 (.29)

Table 2: Main OAS comparison with Qwen2.5-3B-Instruct. Exact-match percentages are reported as mean (sample SD) over two runs ($n = 2$). Bold and underlined means denote the best and second-best mean in each column, respectively.

domain. For sequence i , define

$$\rho_i = \frac{\sum_t m_{i,t}^{V,\text{obs}}}{\sum_t m_{i,t}^{V,\pi} + \epsilon}, \quad (11)$$

$$\alpha_i^{\text{obs}} = \alpha_{\min} + (1 - \alpha_{\min}) \exp(-\beta \rho_i),$$

$$0 \leq \alpha_{\min} < 1, \quad \beta > 0.$$

Thus, trajectories with larger observation-to-generation ratios retain less observation-side TD-residual mass. Appendix C.1 analyzes the coefficient’s monotonicity and parameter roles.

Observation contribution decay. OAS preserves generated-token residuals and scales observation-token residuals by α_i^{obs} :

$$w_{i,t}^\alpha = m_{i,t}^{V,\pi} + \alpha_i^{\text{obs}} m_{i,t}^{V,\text{obs}}. \quad (12)$$

For generated-token positions $(i, t) \in D_\pi$, the modified advantage and return target are

$$\hat{A}_{i,t}^\alpha = \sum_{\ell \geq 0} (\gamma \lambda)^\ell w_{i,t+\ell}^\alpha \delta_{i,t+\ell}, \quad \hat{R}_{i,t}^\alpha = V_{\phi_{\text{old}}}(s_{i,t}) + \hat{A}_{i,t}^\alpha, \quad m_{i,t}^{V,\pi} = 1. \quad (13)$$

Observation positions retain the original token-level target $\hat{R}^V$. The resulting domain-specific loss L_{value}^α therefore uses $\hat{R}^\alpha$ on generated positions and $\hat{R}^V$ on observation positions.

Observation old-value anchoring. To limit observation-side value drift, OAS augments L_{value}^α with

$$L_{\text{value}}^{\text{OVR}}(\phi) = L_{\text{value}}^\alpha(\phi) + \lambda_{\text{obs}} \mathbb{E}_{i,t} \left[(1 - \alpha_i^{\text{obs}}) m_{i,t}^{V,\text{obs}} \cdot (V_\phi(s_{i,t}) - V_{\phi_{\text{old}}}(s_{i,t}))^2 \right]. \quad (14)$$

The factor $1 - \alpha_i^{\text{obs}}$ strengthens anchoring on trajectories receiving greater observation-residual decay, while $m_{i,t}^{V,\text{obs}}$ restricts the penalty to observation positions.

Observation-boundary smoothing. OAS marks adjacent generated/observation transitions and pe-

nalizes their value gaps:

$$m_{i,t}^{\text{bd}} = m_{i,t}^{V,\pi} m_{i,t+1}^{V,\text{obs}} + m_{i,t}^{V,\text{obs}} m_{i,t+1}^{V,\pi},$$

$$J_{\text{bd}}(\phi) = \mathbb{E}_{i,t} \left[m_{i,t}^{\text{bd}} (V_{\phi}(s_{i,t}) - V_{\phi}(s_{i,t+1}))^2 \right]. \quad (15)$$

The final value objective is

$$L_{\text{value}}^{\text{OAS}}(\phi) = L_{\text{value}}^{\text{OVR}}(\phi) + \lambda_{\text{sm}} J_{\text{bd}}(\phi). \quad (16)$$

Together, contribution decay limits observation-residual propagation into generated-token advantages, old-value anchoring constrains observation-side value drift, and boundary smoothing regularizes local value discontinuities. Appendices C.2–C.4 provide the complete loss expansion and limiting cases.

6 Experiments

This section evaluates whether observation-aware stabilization (OAS) improves task performance, training dynamics, and search-budget control. We examine final QA performance across datasets, retrieval budgets, and model sizes; evaluate bounded transfer to FEVER fact verification; analyze how nVDD, ATE, AUS, and OAGC evolve together with EM; and study retrieval behavior and search-budget use during training.

6.1 Experimental Setup

We use a Search-R1-style search-tool RL framework on three benchmark datasets: NQ, TriviaQA, and HotpotQA. During each rollout, the model generates search actions, the retriever inserts observation tokens into the context, and the model continues reasoning before producing a final answer. Main-text experiments use Qwen2.5-3B-Instruct (Yang et al., 2024), with Qwen2.5-1.5B-Instruct used as an additional scale check. We report final EM across datasets and Top- k budgets with $k \in \{2, 3, 5\}$. We analyze nVDD, ATE, and OAGC in the Qwen2.5-3B Top-3 ablations, and time-align EM, nVDD, ATE, AUS, and OAGC for the controlled Qwen2.5-1.5B NQ Top-3 PPO and OAS trajectories. The OAS defaults are $\alpha_{\min} = 0.2000$, $\beta = 0.2812$, $\lambda_{\text{obs}} = 1.0000$, and $\lambda_{\text{sm}} = 0.0277$; all were frozen before formal evaluation. All methods within a setting use the same split, retrieval budget, generation limits, and exact-match scorer; full dataset, baseline, coefficient-selection, and sensitivity details are in Appendix D.

6.2 OAS Results Across Model Sizes

We first compare OAS with baselines on Qwen2.5-3B-Instruct, testing task performance across datasets and retrieval budgets. Table 2 reports mean (sample SD) over two runs for the main 3B comparison. OAS has a higher reported two-run mean than Search-R1 PPO in all nine settings and the highest mean among all compared methods in eight of nine settings. Under the Top-3 setting, its means are 37.6/56.8/23.9 EM on NQ/TriviaQA/HotpotQA, compared with 31.8/49.6/18.7 for Search-R1 PPO. OVR-only is already competitive, consistent with observation-side value regularization and residual decay contributing to the gain; full OAS has higher means than OVR-only in all nine settings, consistent with an additional contribution from boundary smoothing. The pattern is not tied to a single Top- k : NQ and TriviaQA have their highest OAS means at Top-3, while HotpotQA has its highest OAS mean at Top-5. Multi-hop questions can benefit from additional evidence, whereas shorter factoid questions may suffer from irrelevant observations and a heavier context-modeling burden.

Table 3 reports the corresponding 1.5B results. The 1.5B results follow the main trend. OAS has a higher reported two-run mean than Search-R1 PPO in all nine 1.5B settings, including Top-3 margins of 1.0/5.4/4.5 EM on NQ/TriviaQA/HotpotQA, and has the highest mean among all compared methods in seven of nine settings. On NQ, OAS has the highest mean at Top-2 but not at Top-3 or Top-5, suggesting that the smaller model is more sensitive to query quality, retrieval noise, and context use. When the model has limited capacity to filter observations reliably, adding more retrieved information does not necessarily improve performance.

6.3 Transfer to FEVER Fact Verification

To examine OAS beyond open-domain QA, we evaluate it on FEVER fact verification (Thorne et al., 2018) using Qwen2.5-1.5B-Instruct. FEVER assigns each claim one of three labels—SUPPORTS, REFUTES, or NOT ENOUGH INFO—using Wikipedia evidence. Each claim receives one fixed Top-3 retrieval, and PPO and OAS share the claims, observations, data order, training budget, reward, and evaluation protocol. We evaluate search-on three-way label accuracy at steps 25, 50, and 75 and compute nVDD over the same validation sequence.

As shown in Table 5, PPO peaks at 0.514 label

Method	Top- k = 2			Top- k = 3			Top- k = 5		
	NQ	TriviaQA	HotpotQA	NQ	TriviaQA	HotpotQA	NQ	TriviaQA	HotpotQA
Search-R1 PPO	27.9 (.38)	45.0 (.95)	15.3 (.57)	28.4 (.77)	45.7 (1.14)	16.0 (.23)	29.5 (.52)	45.6 (.82)	16.3 (.56)
Search-R1 GRPO	26.2 (.86)	43.5 (.44)	14.6 (.13)	26.9 (.59)	44.1 (1.16)	15.2 (.89)	25.7 (.06)	42.9 (.94)	15.5 (.01)
DAPO	<u>29.4</u> (1.13)	46.1 (.25)	16.1 (.10)	<u>30.0</u> (.17)	46.8 (.90)	16.8 (.55)	33.8 (.52)	45.8 (.62)	17.0 (1.10)
KL-Cov	28.6 (.53)	45.4 (.88)	15.1 (.05)	29.1 (.47)	46.0 (.99)	15.8 (.71)	28.0 (.47)	45.1 (.84)	16.0 (.60)
DAR	29.0 (.29)	46.2 (.71)	16.3 (.62)	28.5 (.08)	46.9 (.92)	17.0 (.67)	29.6 (.25)	46.0 (.17)	17.2 (.73)
I-PPO	28.0 (.40)	45.2 (.11)	15.5 (.88)	28.5 (2.42)	45.9 (.45)	16.2 (.18)	27.6 (.92)	<u>46.8</u> (.58)	16.4 (.96)
Ours (OVR only)	28.6 (.70)	<u>47.5</u> (.84)	<u>17.4</u> (.75)	30.2 (.76)	<u>48.3</u> (.30)	<u>18.2</u> (.23)	28.2 (.42)	46.4 (.79)	<u>18.6</u> (.69)
Ours (OAS)	32.7 (.57)	50.2 (.47)	19.3 (1.56)	29.4 (.64)	51.1 (.88)	20.5 (.38)	<u>32.1</u> (.49)	47.4 (1.35)	21.0 (.42)

Table 3: Additional OAS comparison with Qwen2.5-1.5B-Instruct. Exact-match percentages are reported as mean (sample SD) over two runs ($n = 2$). Bold and underlined means denote the best and second-best mean in each column, respectively.

Method	nVDD↓	ATE↓	OAGC↓
Search-R1 PPO	0.138	13.36	0.331
OAS w/o anchor	0.192	9.84	0.268
OAS w/o decay	0.164	8.72	0.301
OAS w/o smoothing	0.118	5.21	0.142
Full-response smoothing	<u>0.104</u>	<u>4.80</u>	<u>0.126</u>
OAS	0.082	3.56	0.088

Table 4: OAS ablations with Qwen2.5-3B and Top-3. Lower values indicate smaller validation drawdown, advantage-tail expansion, and observation-to-actor contamination.

Method	Step 25	Step 50	Step 75
PPO	0.394 / 0.000	0.514 / 0.000	0.450 / 0.125
OAS	0.381 / 0.000	0.458 / 0.000	0.564 / 0.000

Table 5: FEVER fact-verification results with Qwen2.5-1.5B-Instruct and fixed Top-3 retrieval. Each entry reports search-on three-way label accuracy / nVDD.

accuracy at step 50 but falls to 0.450 at step 75, yielding a final nVDD of 0.125. OAS, in contrast, improves throughout training and reaches 0.564 at step 75 with a final nVDD of 0.000, outperforming PPO by 0.114. These contrasting trajectories show that OAS mitigates late-stage drawdown while improving final label accuracy on FEVER. Together, the results extend the drawdown reduction and final-performance benefits of OAS from open-domain QA to fact verification.

6.4 OAS Ablations and Process Metrics

To analyze OAS gains, Table 4 removes or replaces individual components in the setting of Qwen2.5-3B and Top-3, and compares process metrics. Full OAS obtains the lowest nVDD, ATE, and OAGC. Relative to Search-R1 PPO, it reduces nVDD from 0.138 to 0.082, ATE from 13.36 to 3.56, and OAGC from 0.331 to 0.088. These results show that OAS improves validation stability, limits advantage-tail expansion, and reduces observation-to-actor con-

tamination.

The ablations align with this mechanism. Removing old-value anchoring increases nVDD, while removing contribution decay leaves OAGC close to the PPO baseline, indicating that observation residuals more easily enter generated-token advantages. Removing boundary smoothing still improves over PPO but remains weaker than full OAS, suggesting that boundary value jumps can affect later advantages. Full-response smoothing is also weaker than boundary smoothing on nVDD, ATE, and OAGC, supporting local generated/observation boundaries as the more targeted intervention points.

6.5 Time-aligned Training Diagnostics

Figure 2 aligns EM, nVDD, ATE, AUS, and OAGC at 25-step intervals for PPO and OAS with Qwen2.5-1.5B on NQ under Top-3 retrieval. EM is evaluated every five steps, and nVDD at each displayed point is computed from the complete validation history up to that step. ATE, AUS, and OAGC are measured at the corresponding training steps.

Both methods improve EM during the early and middle stages of training, reaching 0.318 for PPO and 0.316 for OAS at step 175. PPO EM subsequently decreases to 0.278 at step 250, while its nVDD increases to 0.215 and its ATE, AUS, and OAGC reach 10.130, 7.330, and 0.328. At the same

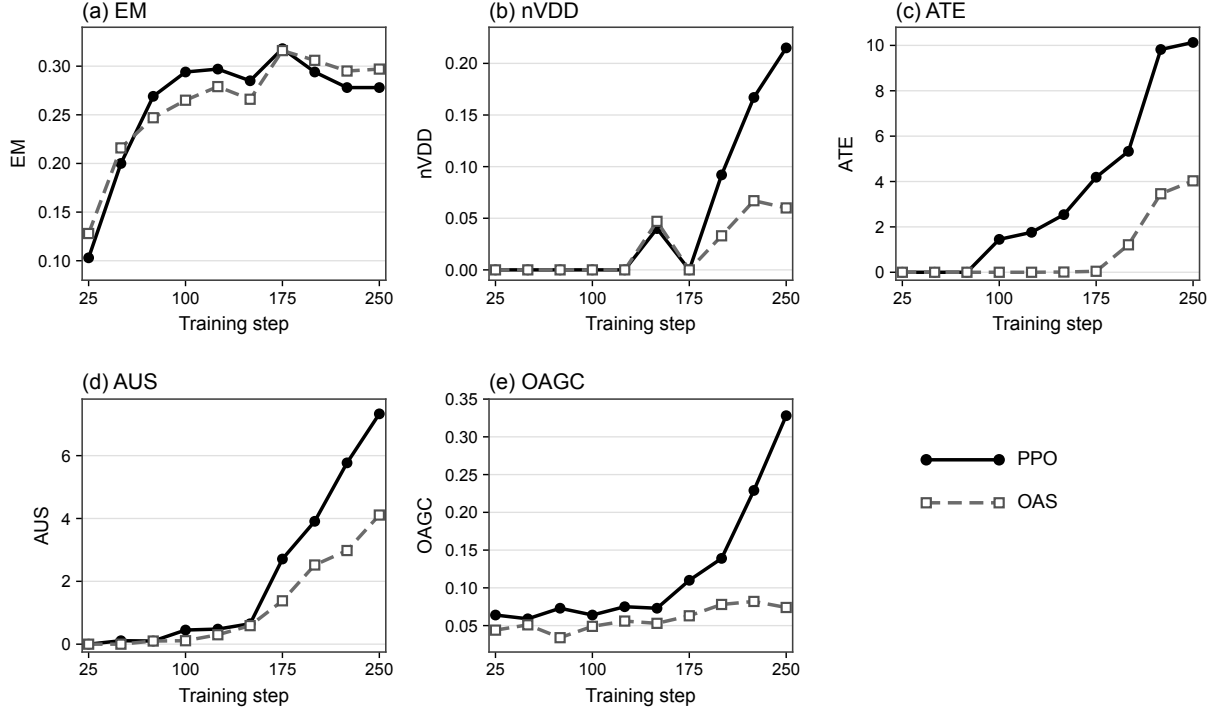


Figure 2: Time-aligned validation and process diagnostics for PPO and OAS with Qwen2.5-1.5B on NQ under Top-3 retrieval. Solid black lines with circular markers denote PPO, and dashed gray lines with square markers denote OAS. Values are displayed every 25 training steps; nVDD is computed from validation performed every five steps. AUS is measured across complete actor updates and uses a shared PPO/OAS median/MAD reference.

step, OAS retains an EM of 0.297, with nVDD, ATE, AUS, and OAGC of 0.060, 4.030, 4.110, and 0.074.

The aligned trajectories show that advantage-tail expansion and actor-update stress accumulate before and alongside the largest validation drawdown, while observation-to-actor contamination increases mainly in the later stage. Together, the metrics connect changes in optimization dynamics with the subsequent decline in validation performance.

The figure presents one controlled trajectory per method and therefore characterizes temporal behavior within this experiment without estimating run-to-run uncertainty or defining universal diagnostic thresholds.

6.6 Retrieval Behavior and Training Dynamics

On a fixed Qwen2.5-1.5B NQ Top-3 evaluation set, OAS shows more restrained late-stage search-budget use while preserving active tool engagement. At step 250, OAS achieves higher EM and lower nVDD than PPO (0.297/0.060 versus 0.278/0.215), with fewer executed searches and lower budget saturation (1.154/0.127 versus 1.673/0.357) and a lower no-search rate (0.017 ver-

sus 0.039). Complete step-150/250 statistics, the proxy definition, and detailed discussion are provided in Appendix E and Table 10.

7 Conclusion

This paper studies late-stage instability in search-tool RL and identifies observation-token propagation through value learning and GAE as an under-explored pathway. We introduce diagnostic metrics and OAS, a lightweight method combining observation residual decay, old-value anchoring, and boundary smoothing. Across open-domain QA and a bounded FEVER transfer, OAS improves final task performance while reducing nVDD, ATE, and OAGC. The time-aligned analysis connects increasing ATE, AUS, and OAGC with validation drawdown, while the retrieval-behavior analysis shows lower late-stage search-budget saturation without increased tool avoidance. The nearly unchanged answer-string miss proxy does not establish improved retrieval relevance or faithfulness. Together, the diagnostics, mechanism analysis, and OAS intervention characterize and mitigate instability in search-tool RL.

Limitations

Our experiments evaluate Qwen2.5-1.5B/3B models across three open-domain QA datasets and further extend to Qwen2.5-1.5B FEVER fact verification under fixed Top-3 Wikipedia retrieval. The FEVER results show that the stability and final-performance benefits of OAS are not confined to open-domain QA. We have not yet evaluated larger model scales, real web-browsing environments, multimodal tools, or more complex long-horizon agent tasks, and leave validation in these settings to future work.

Our mechanism analysis is not a complete causal account of search-tool RL instability. OAGC, domain-specific value-loss results, and the OAS ablations are consistent with mask-induced actor-critic domain mismatch. However, search-tool RL instability may also depend on reward sparsity, retriever noise, policy drift, long-sequence optimization, and shifts in the sampling distribution. We treat observation-side value and TD-residual propagation as a measurable pathway targeted by OAS, not as the only explanation for collapse.

Finally, OAS is designed primarily for PPO-style training with a critic and GAE. Its components may need to be reformulated for critic-free methods such as GRPO or systems with different state representations, masking rules, and tool-feedback formats. OAS also introduces additional hyperparameters and measurable training-time overhead. Appendix D.2 reports one matched cost comparison using Qwen2.5-1.5B on NQ with Top-3 retrieval, and Appendix D reports a one-at-a-time coefficient sensitivity study in a Qwen2.5-3B NQ Top-3 setting. Broader computational profiling across model scales, hardware platforms, and task regimes, broader coefficient sweeps, and interactions with other stabilization techniques remain future work.

Potential risks also remain. Stabilizing retrieval-augmented RL may make search agents more reliable at using retrieved content, but retrieved evidence can still be noisy, biased, outdated, or manipulated. More stable training therefore does not remove the need for retrieval filtering, factuality checks, and safety evaluation before deployment.

Acknowledgments

AI assistants were used for language polishing, drafting support, and checking submission-related text. The authors reviewed, edited, and verified all

AI-assisted content and remain fully responsible for the paper.

References

- Jiajun Chai, Guojun Yin, Zekun Xu, Chuhuai Yue, Yi Jia, Siyu Xia, Xiaohan Wang, Jiwen Jiang, Xiaoguang Li, Chengqi Dong, Hang He, and Wei Lin. 2025. [RLFactory: A plug-and-play reinforcement learning post-training framework for LLM multi-turn tool-use](#). *arXiv preprint arXiv:2509.06980*.
- Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, and Ning Ding. 2025. [The entropy mechanism of reinforcement learning for reasoning language models](#). *arXiv preprint arXiv:2505.22617*.
- Guanting Dong, Licheng Bao, Zhongyuan Wang, Kangzhi Zhao, Xiaoxi Li, Jiajie Jin, Jinghan Yang, Hangyu Mao, Fuzheng Zhang, Kun Gai, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. 2026a. [Toward generalized web agent training: A deep dive into entropy-balanced reinforcement learning](#). In *Proceedings of the ACM Web Conference 2026*, pages 2126–2137. ACM.
- Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. 2026b. [Agentic reinforced policy optimization](#). In *International Conference on Learning Representations*.
- Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2023. [Retrieval-augmented generation for large language models: A survey](#). *arXiv preprint arXiv:2312.10997*.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. [DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning](#). *Nature*, 645(8081):633–638.
- Li He, He Zhao, Stephen Wan, Dadong Wang, Lina Yao, and Tongliang Liu. 2025. [Direct advantage regression: Aligning LLMs with online AI reward](#). *arXiv preprint arXiv:2504.14177*.
- Daniel R. Jiang, Jalaj Bhandari, Yukai Yang, Rémi Munos, and Tyler Lu. 2025. [Aligning LLMs toward multi-turn conversational outcomes using iterative PPO](#). *arXiv preprint arXiv:2511.21638*.
- Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Ö. Arik, Dong Wang, Hamed Zamani, and

- Jiawei Han. 2025. [Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning](#). In *Second Conference on Language Modeling*.
- Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. [TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.
- 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. 2019. [Natural questions: A benchmark for question answering research](#). *Transactions of the Association for Computational Linguistics*, 7:452–466.
- 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. 2020. [Retrieval-augmented generation for knowledge-intensive NLP tasks](#). In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.
- Heng Lin and Zhongwen Xu. 2025. [Understanding tool-integrated reasoning](#). In *NeurIPS 2025 Workshop MATH-AI*.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. [Training language models to follow instructions with human feedback](#). In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.
- Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tur, Gokhan Tur, and Heng Ji. 2025. [ToolRL: Reward is all tool learning needs](#). In *Advances in Neural Information Processing Systems*, volume 38, pages 105523–105553. Curran Associates, Inc.
- John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2016. [High-dimensional continuous control using generalized advantage estimation](#). In *International Conference on Learning Representations*.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. [Proximal policy optimization algorithms](#). *arXiv preprint arXiv:1707.06347*.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. [DeepSeekMath: Pushing the limits of mathematical reasoning in open language models](#). *arXiv preprint arXiv:2402.03300*.
- James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. [FEVER: a large-scale dataset for fact extraction and VERification](#). In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.
- Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, Yiping Lu, Kyunghyun Cho, Jiajun Wu, Li Fei-Fei, Lijuan Wang, Yejin Choi, and Manling Li. 2025. [RA-GEN: Understanding self-evolution in LLM agents via multi-turn reinforcement learning](#). *arXiv preprint arXiv:2504.20073*.
- Guangzhi Xiong, Qiao Jin, Xiao Wang, Yin Fang, Haolin Liu, Yifan Yang, Fangyuan Chen, Zhixing Song, Dengyu Wang, Minjia Zhang, Zhiyong Lu, and Aidong Zhang. 2025. [Supervising the search process produces reliable and generalizable information-seeking agents](#). *arXiv preprint arXiv:2502.13957*.
- Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Xiaosen Zheng, Zejun Ma, and Bo An. 2025. [SimpleTIR: End-to-end reinforcement learning for multi-turn tool-integrated reasoning](#). In *NeurIPS 2025 Workshop DL4C*.
- An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 23 others. 2024. [Qwen2.5 technical report](#). *arXiv preprint arXiv:2412.15115*.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. [HotpotQA: A dataset for diverse, explainable multi-hop question answering](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023. [ReAct: Synergizing reasoning and acting in language models](#). In *International Conference on Learning Representations*.
- Qiyang Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gao-hong Liu, Juncai Liu, LingJun Liu, Xin Liu, Haibin

Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, and 17 others. 2025. [DAPO: An open-source LLM reinforcement learning system at scale](#). In *Advances in Neural Information Processing Systems*, volume 38, pages 113222–113244. Curran Associates, Inc.

Siliang Zeng, Quan Wei, William Brown, Oana Frunza, Yuriy Nevmyvaka, Yang Katie Zhao, and Mingyi Hong. 2025. [Reinforcing multi-turn reasoning in LLM agents via turn-level credit assignment](#). In *ICML 2025 Workshop on Computer-use Agents*.

Wenlin Zhang, Xiangyang Li, Kuicai Dong, Yichao Wang, Pengyue Jia, Xiaopeng Li, Yingyi Zhang, Derong Xu, Zhaocheng Du, Huifeng Guo, Ruiming Tang, and Xiangyu Zhao. 2025. [Process vs. outcome reward: Which is better for agentic RAG reinforcement learning](#). In *Advances in Neural Information Processing Systems*, volume 38, pages 58701–58729. Curran Associates, Inc.

Chujie Zheng, Kai Dang, Bowen Yu, Mingze Li, Huiqiang Jiang, Junrong Lin, Yuqiong Liu, Hao Lin, Chencan Wu, Feng Hu, An Yang, Jingren Zhou, and Junyang Lin. 2025. [Stabilizing reinforcement learning with LLMs: Formulation and practices](#). *arXiv preprint arXiv:2512.01374*.

A Complete Definitions of Trajectory-Level Instability Metrics

A.1 Validation Drawdown

Validation drawdown (VDD) measures how much previously acquired validation performance is lost during training. For the dynamic in which the model first reaches usable performance and then degrades, VDD directly measures the current validation score’s drop from the historical best score. Let y_t be the validation score around training step t , and define the historical best validation score as:

$$\text{VBest}_t = \max_{s \leq t} y_s.$$

Validation drawdown and normalized validation drawdown are:

$$\begin{aligned} \text{VDD}_t &= \text{VBest}_t - y_t, \\ \text{nVDD}_t &= \frac{\text{VDD}_t}{|\text{VBest}_t| + \epsilon}. \end{aligned}$$

Here ϵ avoids instability when the historical best score is close to zero; nVDD_t normalizes the drawdown by the historical best score so that drawdown magnitudes are comparable across runs. VDD is the core outcome metric for measuring late-stage collapse in this paper, and serves as the outcome-side anchor for examining whether ATE, AUS, and OAGC appear simultaneously with, or before, the drawdown.

A.2 Advantage Tail and Actor Update Stress

VDD describes regression in training outcomes; ATE and AUS describe two optimization-side signals associated with that regression. Since the available logs only provide aggregate quantities such as advantage, KL, and clip fraction, ATE and AUS are constructed as heuristic proxies from observable logs.

Advantage tail expansion (ATE) focuses on whether the advantage weights used by actor updates exhibit tail expansion. PPO actor updates are weighted by advantages; if the tail of the advantage distribution expands abnormally, a small number of high-magnitude tokens can dominate the update direction. Let the advantage range at a training step be:

$$\text{AdvRange}_t = \max_i \hat{A}_{t,i} - \min_i \hat{A}_{t,i}.$$

Here i denotes the flattened token-sample index over all generated tokens in the current batch that participate in the actor update. We use the positive robust anomaly of AdvRange_t as ATE, where robust_z_+ keeps only robust z-scores above the normal range:

$$\begin{aligned} \text{robust_z}_+(x_t) &= \max(r_t, 0), \\ r_t &= \frac{x_t - \text{median}(x)}{1.4826 \cdot \text{MAD}(x) + \epsilon}. \end{aligned}$$

ATE is therefore defined as:

$$\text{ATE}_t = \text{robust_z}_+(\text{AdvRange}_t).$$

ATE serves as an advantage-tail proxy during training, indicating whether the advantage distribution used by actor updates has expanded abnormally. When GAE or return/value estimation gives a small number of tokens high-magnitude advantages, AdvRange_t increases and ATE rises.

ATE describes whether the advantage weights themselves are abnormal, while actor update stress (AUS) measures the stress induced by a complete PPO actor optimization step. We construct it from positive robust anomalies of post-step KL and policy-gradient clip fraction:

$$\begin{aligned} \text{AUS}_t &= \frac{1}{2} \text{robust_z}_+^{\text{shared}}(|\text{PostKL}_t|) \\ &\quad + \frac{1}{2} \text{robust_z}_+^{\text{shared}}(\text{PostClipFrac}_t). \end{aligned}$$

Here PostKL_t and PostClipFrac_t are measured on the same frozen actor-loss batch immediately

before and after a complete PPO actor optimization step, including all mini-batches, PPO epochs, and optimizer steps. The PPO and OAS trajectories use one shared median/MAD reference constructed from all valid steps. When KL and clip fraction both enter high ranges, the actor optimization step has moved away from the local update region that PPO is designed to maintain. In search-tool tasks, such deviation can affect ordinary next-token generation as well as search timing, query content, retrieved observations, and subsequent context.

A.3 Observation-to-Actor GAE Contamination

ATE and AUS describe the overall pressure on actor updates, but they do not distinguish whether advantage abnormalities come from model-generated tokens or external observation tokens. In search-tool trajectories, observation tokens do not participate in actor loss, but they can still affect preceding generated-token advantages through TD residuals and GAE. OAGC therefore measures the fraction of generated-token advantage contributed by future observation-token residuals.

Let D_π denote the set of generated-token positions that participate in actor loss, and let D_{obs} denote the set of observation-token positions returned by the retriever. For any $t \in D_\pi$, let its GAE advantage be A_t . Let $\delta_{t+\ell}$ be the TD residual at position $t + \ell$. Let γ and λ be the discount factor and the GAE parameter. Keeping only future observation-token contributions to this advantage gives:

$$A_t^{\text{obs}} = \sum_{\ell \geq 0, t+\ell \in D_{\text{obs}}} (\gamma\lambda)^\ell \delta_{t+\ell}.$$

This expression follows the temporal recursion of GAE, but keeps only positions in D_{obs} . It therefore gives the contribution of future observation-token TD residuals to the current generated-token advantage.

The resulting OAGC ratio is:

$$\text{OAGC} = \frac{\sum_{t \in D_\pi} |A_t^{\text{obs}}|}{\sum_{t \in D_\pi} |A_t| + \epsilon}.$$

Higher OAGC means that a larger fraction of generated-token advantage comes from observation-side residuals.

B Complete Derivation of the Observation-to-Actor Propagation Chain

B.1 Error Decomposition in Return/value Target Construction

B.1.1 Derivation for Proposition 1

Consider the i -th search-tool trajectory:

$$z_i = (x_i, G_{i,1}, O_{i,1}, G_{i,2}, O_{i,2}, \dots, G_{i,K_i}, y_i),$$

where $G_{i,k}$ is a model-generated reasoning, query, or answer token span, and $O_{i,k}$ is an observation span returned by the retriever or tool. Let the sequence positions be $t = 1, \dots, T_i$, and define:

$$m_{i,t}^\pi = \mathbf{1}[z_{i,t} \in G_i], \quad m_{i,t}^{\text{obs}} = \mathbf{1}[z_{i,t} \in O_i],$$

$$m_{i,t}^V \in \{0, 1\}.$$

Here m^π denotes generated-token positions kept in actor loss, m^{obs} denotes observation-token positions, and m^V denotes the mask used for GAE / value-target construction. Existing Search-R1-style search-tool training usually masks observation tokens in actor loss. At the same time, observation tokens remain in the context. The value-model update, return targets, TD residuals, and GAE may still use these positions. Denote generated-token positions and observation-token positions by:

$$\mathcal{G} = \{(i, t) : m_{i,t}^\pi = 1\},$$

$$\mathcal{O} = \{(i, t) : m_{i,t}^{\text{obs}} = 1\}.$$

Observation tokens are kept in the context because they provide state information for later generation. However, they may be excluded only from actor loss while still entering return targets and the GAE residual chain. In that case, generated-token targets contain additional terms.

PPO samples trajectories with the old policy and uses the importance ratio to express the difference between the current policy and the sampling policy:

$$\rho_{i,t}(\theta) = \frac{\pi_\theta(z_{i,t} \mid z_{i,<t})}{\pi_{\theta_{\text{old}}}(z_{i,t} \mid z_{i,<t})}.$$

On generated-token positions $\mathcal{G}$, the clipped actor objective is:

$$L_\pi(\theta) = \frac{1}{|\mathcal{G}|} \sum_{(i,t) \in \mathcal{G}} \min\left(\rho_{i,t}(\theta) \hat{A}_{i,t}, \text{clip}(\rho_{i,t}(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_{i,t}\right).$$

The actor objective sums only over generated tokens, but $\hat{A}_{i,t}$ is supplied by the value model and GAE. If the GAE residual chain passes through observation positions, the advantage is constructed as:

$$\delta_{i,t} = r_{i,t} + \gamma V_{\phi_{\text{old}}}(s_{i,t+1}) - V_{\phi_{\text{old}}}(s_{i,t}),$$

$$\hat{A}_{i,t}^V = \sum_{\ell=0}^{T_i-t} (\gamma\lambda)^\ell m_{i,t+\ell}^V \delta_{i,t+\ell}.$$

The corresponding token-level return target is:

$$\hat{R}_{i,t}^V = V_{\phi_{\text{old}}}(s_{i,t}) + \hat{A}_{i,t}^V.$$

The value loss is:

$$L_V(\phi) = \frac{\sum_{i,t} m_{i,t}^V (V_{\phi}(s_{i,t}) - \hat{R}_{i,t}^V)^2}{\sum_{i,t} m_{i,t}^V}.$$

Decompose m^V into generated and observation parts:

$$m_{i,t}^V = m_{i,t}^{V,\pi} + m_{i,t}^{V,\text{obs}}.$$

Substitution gives:

$$\hat{A}_{i,t}^V = \sum_{\ell=0}^{T_i-t} (\gamma\lambda)^\ell (m_{i,t+\ell}^{V,\pi} + m_{i,t+\ell}^{V,\text{obs}}) \delta_{i,t+\ell}.$$

Thus, whenever $m^{V,\text{obs}}$ is nonzero, observation-token TD residuals enter GAE. For generated-token positions that participate in actor loss, these residuals further enter the generated-token advantages used in the actor update.

To isolate the terms introduced by the actual token-level target construction, define a generated-token-level reference target. This target recurses only over generated-token decision points. It treats the observation span as a tool-induced state transition rather than an independent actor decision point.

For any generated-token position $(i, j) \in \mathcal{G}$, let j^+ be the next generated-token position after j in the same rollout, and let $\Delta_{i,j} = j^+ - j$. Define the span reward from j to j^+ as:

$$r_{i,j:j^+}^{\text{span}} = \sum_{h=0}^{\Delta_{i,j}-1} \gamma^h r_{i,j+h}.$$

Define the reference TD residual as:

$$\delta_{i,j}^* = r_{i,j:j^+}^{\text{span}} + \gamma^{\Delta_{i,j}} V_{\phi_{\text{old}}}(s_{i,j^+}) - V_{\phi_{\text{old}}}(s_{i,j}).$$

Let $j = j_0 < j_1 < \dots$ denote the generated-token positions starting from j . The generated-token-level GAE and return target are:

$$\begin{aligned} \hat{A}_{i,j}^* &= \sum_{q \geq 0} \gamma^{j_q-j} \lambda^q \delta_{i,j_q}^*, \\ \hat{R}_{i,j}^* &= V_{\phi_{\text{old}}}(s_{i,j}) + \hat{A}_{i,j}^*. \end{aligned}$$

This reference target is only a comparison baseline: it corresponds to the return signal received by generated tokens when only generated tokens are treated as decision points.

Define the difference between the actual token-level return target and the generated-token-level reference target:

$$E_{i,j}^R = \hat{R}_{i,j}^V - \hat{R}_{i,j}^*.$$

Decompose the actual return target into a generated-token term and an observation-token additive term:

$$\hat{R}_{i,j}^V = \hat{R}_{i,j}^\pi + C_{i,j}^{\text{obs}},$$

where:

$$\hat{R}_{i,j}^\pi = V_{\phi_{\text{old}}}(s_{i,j}) + \sum_{\ell=0}^{T_i-j} (\gamma\lambda)^\ell m_{i,j+\ell}^{V,\pi} \delta_{i,j+\ell},$$

$$C_{i,j}^{\text{obs}} = \sum_{\ell=0}^{T_i-j} (\gamma\lambda)^\ell m_{i,j+\ell}^{V,\text{obs}} \delta_{i,j+\ell}.$$

Then:

$$\begin{aligned} E_{i,j}^R &= \underbrace{\hat{R}_{i,j}^\pi - \hat{R}_{i,j}^*}_{\text{generated-token estimator mismatch}} \\ &+ \underbrace{C_{i,j}^{\text{obs}}}_{\text{observation residual contribution}}. \end{aligned}$$

The first term is the discrepancy between the token-level estimator and the generated-token-level estimand. It can remain even when observation residuals are removed. The second term is the direct contribution of observation-token residuals to GAE / value-target construction.

Expanding the first term yields two sources of generated-token estimator mismatch.

The first source is trace-decay mismatch. Actual token-level GAE multiplies by λ for every token crossed. By contrast, generated-token-level GAE multiplies by λ only when crossing a generated decision point. If j_q is the q -th generated-token position, then:

$$S_{i,j}^\lambda = \sum_q \gamma^{j_q-j} \lambda^q (\lambda^{j_q-j-q} - 1) \delta_{i,j_q}.$$

If observation tokens are inserted between j and j_q , then $j_q - j - q > 0$. Since $0 < \lambda < 1$, $\lambda^{j_q - j - q} < 1$. Future generated-token residuals are therefore additionally decayed. Thus, even when observation residuals do not enter as an additive term, inserting an observation span changes the GAE trace-decay path.

The second source is bootstrap-state mismatch. Token-level TD residuals bootstrap from the current token to the next token state. Generated-token-level residuals skip the observation span and bootstrap to the next generated-token state. Their difference is:

$$\begin{aligned} \delta_{i,j_q} - \delta_{i,j_q}^* &= \gamma V_{\phi_{\text{old}}}(s_{i,j_q+1}) \\ &\quad - \gamma^{\Delta_{i,j_q}} V_{\phi_{\text{old}}}(s_{i,j_q+1}) \\ &\quad - \sum_{h=1}^{\Delta_{i,j_q}-1} \gamma^h r_{i,j_q+h}. \end{aligned}$$

Let:

$$S_{i,j}^{\text{boot}} = \sum_q \gamma^{j_q-j} \lambda^q (\delta_{i,j_q} - \delta_{i,j_q}^*).$$

Then:

$$\hat{R}_{i,j}^\pi - \hat{R}_{i,j}^* = S_{i,j}^\lambda + S_{i,j}^{\text{boot}}.$$

Combining the above decompositions gives the complete mismatch:

$$\begin{aligned} E_{i,j}^R &= \underbrace{S_{i,j}^\lambda}_{\text{trace-decay mismatch}} + \underbrace{S_{i,j}^{\text{boot}}}_{\text{bootstrap-state mismatch}} \\ &\quad + \underbrace{C_{i,j}^{\text{obs}}}_{\text{observation residual contribution}}. \end{aligned}$$

Taking conditional expectation gives:

$$\begin{aligned} \mathbb{E}[E_{i,j}^R \mid s_{i,j}] &= \mathbb{E}[S_{i,j}^\lambda + S_{i,j}^{\text{boot}} \mid s_{i,j}] \\ &\quad + \mathbb{E}[C_{i,j}^{\text{obs}} \mid s_{i,j}]. \end{aligned}$$

where:

$$\begin{aligned} \mathbb{E}[C_{i,j}^{\text{obs}} \mid s_{i,j}] &= \sum_{\ell=0}^{T_i-j} (\gamma\lambda)^\ell \mathbb{E}[m_{i,j+\ell}^{V,\text{obs}} \delta_{i,j+\ell} \mid s_{i,j}]. \end{aligned}$$

Therefore, the conditional expectation of $C_{i,j}^{\text{obs}}$ vanishes only when the discounted observation-token residual sum has zero conditional expectation. In general, observation tokens affect generated-token return targets through two paths. The first is a structural trace-decay / bootstrap-state path induced by inserting observation spans. The second is a residual path in which observation-token TD residuals directly enter GAE.

B.1.2 Case 1: Minimal Rollout with One Observation Token

Take the minimal rollout:

$$G_0, O_1, G_2,$$

where G is a generated token and O is an observation token. Assume that only the final generated token receives outcome reward:

$$r_2 = y, \quad y \in \{+1, -1\}, \quad r_0 = r_1 = 0.$$

Let:

$$v_k = V_{\phi_{\text{old}}}(s_k).$$

For G_0 , the difference between the actual token-level target and the generated-token-level reference target is:

$$\begin{aligned} \hat{R}_0^V - \hat{R}_0^* &= \underbrace{\gamma^2 \lambda (\lambda - 1) (y - v_2)}_{\text{trace-decay mismatch}} \\ &\quad + \underbrace{\gamma v_1 - \gamma^2 v_2}_{\text{bootstrap-state mismatch}} \\ &\quad + \underbrace{\gamma \lambda (\gamma v_2 - v_1)}_{\text{observation residual contribution}}. \end{aligned}$$

In this case, all three sources appear explicitly. The first term is the additional trace decay induced by applying token-level GAE across the intermediate observation token. The second term is the difference between bootstrapping the actual TD residual to s_1 and bootstrapping the reference residual to s_2 . The third term is the TD residual of the observation token itself. The value v_1 is the value of an observation state. It is not associated with an actor action, yet it enters the generated-token return target. If v_1 is unstable, this error enters the actor update through $\hat{R}_0^V$ and $\hat{A}_0^V$. The same path can occur when the observation span differs sharply from generated spans in token distribution.

B.2 Derivation for Proposition 2

By Appendix B.1, the mismatch first arises in GAE / return construction. After the mismatched target is used for the value-model update, let:

$$e_{i,t} = V_\phi(s_{i,t}) - \hat{R}_{i,t}^V,$$

and let:

$$Z_G = \sum_{i,t} m_{i,t}^{V,\pi},$$

$$Z_O = \sum_{i,t} m_{i,t}^{V,\text{obs}},$$

$$Z_V = Z_G + Z_O.$$

The actual value loss is:

$$L_V = \frac{1}{Z_V} \left(\sum_{i,t} m_{i,t}^{V,\pi} e_{i,t}^2 + \sum_{i,t} m_{i,t}^{V,\text{obs}} e_{i,t}^2 \right).$$

On generated-token positions, the return-target decomposition in Proposition 1 gives:

$$\hat{R}_{i,t}^V = \hat{R}_{i,t}^* + E_{i,t}^R.$$

Thus the error term in the actual value-model update satisfies:

$$e_{i,t} = \left(V_\phi(s_{i,t}) - \hat{R}_{i,t}^* \right) - E_{i,t}^R.$$

If the value model were fit only on generated-token positions against the reference target, the ideal gradient would be:

$$\begin{aligned} \nabla_\phi L_V^{*,G} &= \frac{2}{Z_G} \sum_{i,t} m_{i,t}^{V,\pi} \\ &\cdot \left(V_\phi(s_{i,t}) - \hat{R}_{i,t}^* \right) \nabla_\phi V_\phi(s_{i,t}). \end{aligned}$$

The actual generated-token contribution to the value model gradient is:

$$\nabla_\phi L_V|_G = \frac{2}{Z_V} \sum_{i,t} m_{i,t}^{V,\pi} e_{i,t} \nabla_\phi V_\phi(s_{i,t}).$$

Substituting $e_{i,t}$ yields:

$$\begin{aligned} \nabla_\phi L_V|_G &= \frac{Z_G}{Z_V} \nabla_\phi L_V^{*,G} \\ &- \frac{2}{Z_V} \sum_{i,t} m_{i,t}^{V,\pi} E_{i,t}^R \nabla_\phi V_\phi(s_{i,t}). \end{aligned}$$

This expression gives the form of return-target mismatch inside the value-model gradient. The first term shows that the generated-token ideal gradient is scaled by Z_G/Z_V . This scaling appears because observation positions also enter the shared normalization denominator. The second term is the gradient shift introduced by $E_{i,t}^R$. It moves the value model from fitting $\hat{R}^*$ toward fitting the actual token-level target $\hat{R}^V$.

For observation positions, the generated-token-level reference target has no corresponding independent decision point and no independent return target. Hence no analogous mismatch term relative to $\hat{R}^*$ can be written. The actual observation part of the value-model gradient is:

$$\nabla_\phi L_V|_O = \frac{2}{Z_V} \sum_{i,t} m_{i,t}^{V,\text{obs}} e_{i,t} \nabla_\phi V_\phi(s_{i,t}).$$

This gradient is not a biased estimate of a true generated-token-level target. Instead, it introduces an additional value supervision point. This point is absent from the generated-token-level reference target. For a single observation token O , the local value-model update is:

$$\nabla_\phi L_O = 2 \left(V_\phi(s_O) - \hat{R}_O \right) \nabla_\phi V_\phi(s_O).$$

If correct samples dominate the batch, observation-token targets tend to be high:

$$\hat{R}_O > V_\phi(s_O),$$

and the update raises values near the observation token. Conversely, if incorrect samples dominate the batch or sampling distribution:

$$\hat{R}_O < V_\phi(s_O),$$

the update lowers values near the observation token. Since observation tokens are not actor decision points, this direction has no direct policy-improvement meaning. Observations should affect value by changing later generated-token states rather than by independently fitting final outcomes.

As training proceeds, observation-position value-model updates can make the value model more sensitive to observation text distribution, length, position, and retrieved content. For a generated token G , the first-order effect of one observation-token value-model update on $V_\phi(s_G)$ is approximately:

$$\begin{aligned} \Delta V_\phi(s_G)|_O &\approx -\frac{2\eta}{Z_V} \left(V_\phi(s_O) - \hat{R}_O \right) \\ &\cdot \langle \nabla_\phi V_\phi(s_G), \nabla_\phi V_\phi(s_O) \rangle. \end{aligned}$$

If the representation gradients of s_G and s_O are correlated, the value error on the observation token moves the generated-token value prediction. In the next advantage computation, the generated-token baseline has therefore changed:

$$\hat{A}_{i,t} \text{ depends on } V_{\phi_{\text{old}}}(s_{i,t}).$$

B.3 Derivation for Proposition 3

By Appendix B.2, return-target mismatch can change subsequent value baselines through the value-model update. On generated-token positions, decompose the actual GAE advantage into generated-token and observation-token contributions:

$$\hat{A}_{i,t} = A_{i,t}^G + A_{i,t}^O, \quad (i, t) \in \mathcal{G}.$$

The two terms are:

$$A_{i,t}^G = \sum_{\ell \geq 0} (\gamma \lambda)^\ell m_{i,t+\ell}^{V,\pi} \delta_{i,t+\ell},$$

$$A_{i,t}^O = \sum_{\ell \geq 0} (\gamma \lambda)^\ell m_{i,t+\ell}^{V,\text{obs}} \delta_{i,t+\ell}.$$

Here $A_{i,t}^O$ is the part of future observation-token residuals that reaches the current generated-token advantage through GAE recursion. It corresponds to the observation-side advantage contribution in the OAGC definition in Eq. (3.4). The corresponding ratio is:

$$\text{OAGC} = \frac{\sum_{(i,t) \in \mathcal{G}} |A_{i,t}^O|}{\sum_{(i,t) \in \mathcal{G}} |\hat{A}_{i,t}| + \epsilon}.$$

If the value model has an unstable value representation over observation text, one-step TD residual magnitudes inside the observation span can increase. For observation-span tokens without direct task reward, typically:

$$\delta_{i,u} \approx \gamma V_{\phi_{\text{old}}}(s_{i,u+1}) - V_{\phi_{\text{old}}}(s_{i,u}).$$

Substituting this approximation gives the observation-token contribution to a generated-token advantage:

$$A_{i,t}^O = \sum_{\ell \geq 0} (\gamma \lambda)^\ell m_{i,t+\ell}^{V,\text{obs}} \cdot (\gamma V_{\phi_{\text{old}}}(s_{i,t+\ell+1}) - V_{\phi_{\text{old}}}(s_{i,t+\ell})).$$

C OAS Method Details

C.1 Properties of the Observation Decay Coefficient

The observation decay coefficient defined in Section 5 is:

$$\alpha_i^{\text{obs}} = \alpha_{\min} + (1 - \alpha_{\min}) \exp(-\beta \rho_i), \\ 0 \leq \alpha_{\min} < 1, \quad \beta > 0.$$

It sets the retained weight of observation-token TD residuals in generated-token GAE and return targets as a function of the trajectory-level observation-to-generation ratio ρ_i . Differentiating with respect to ρ_i gives:

$$\frac{\partial \alpha_i^{\text{obs}}}{\partial \rho_i} = -(1 - \alpha_{\min}) \beta \exp(-\beta \rho_i) < 0.$$

Thus, α_i^{obs} decreases monotonically with ρ_i . When a trajectory contains a higher fraction of observation tokens, OAS applies stronger decay to observation residuals. When $\rho_i = 0$, $\alpha_i^{\text{obs}} = 1$, and this term recovers the full observation-residual retention in the original token-level GAE.

The lower bound $\alpha_{\min}$ preserves a minimum contribution of observation tokens to later state values, preventing tool information from being completely removed from return estimation. Lower values of $\alpha_{\min}$ make the method closer to cutting off observation residuals. Higher values make it closer to original GAE.

The parameter β controls how quickly the decay increases as ρ_i grows. It can also be written as:

$$\beta = \frac{\log 2}{\rho_{1/2}},$$

where $\rho_{1/2}$ is the observation-to-generation length ratio at which the exponential decay term falls to one half. Smaller $\rho_{1/2}$ corresponds to faster decay, while larger $\rho_{1/2}$ corresponds to slower decay.

C.2 Contribution-Decayed Value Objective

For generated-token positions, OAS uses the contribution-decayed target $\hat{R}^\alpha$, whereas observation-token positions retain the original token-level target $\hat{R}^V$. The complete domain-specific value loss is

$$L_{\text{value}}^\alpha(\phi) = \mathbb{E}_{i,t} [m_{i,t}^{V,\pi} (V_\phi(s_{i,t}) - \hat{R}_{i,t}^\alpha)^2 \\ + m_{i,t}^{V,\text{obs}} (V_\phi(s_{i,t}) - \hat{R}_{i,t}^V)^2].$$

Here $\hat{R}_{i,t}^V$ is the original token-level value target. This domain-specific target construction retains observations as contextual states while reducing the contribution of their TD residuals to generated-token actor updates.

C.3 Observation Old-Value Anchoring

External observations can differ sharply from generated text in length and distribution. OAS therefore adds an observation old-value anchor to the contribution-decayed loss:

$$L_{\text{value}}^{\text{OVR}}(\phi) = L_{\text{value}}^\alpha(\phi) \\ + \lambda_{\text{obs}} \mathbb{E}_{i,t} [(1 - \alpha_i^{\text{obs}}) m_{i,t}^{V,\text{obs}} \\ \cdot (V_\phi(s_{i,t}) - V_{\phi_{\text{old}}}(s_{i,t}))^2].$$

Here $\lambda_{\text{obs}} \geq 0$ controls the anchoring strength. The observation mask excludes generated-token

positions from this penalty, while $1 - \alpha_i^{\text{obs}}$ assigns stronger anchoring to trajectories receiving stronger observation-residual decay. When $\alpha_i^{\text{obs}} = 1$, the anchor vanishes.

C.4 Boundary Smoothing and Limiting Cases

Domain-specific residual weights and value targets can create value gaps when a trajectory crosses between generated-token and observation-token spans. OAS marks these adjacent transitions with

$$m_{i,t}^{\text{bd}} = m_{i,t}^{V,\pi} m_{i,t+1}^{V,\text{obs}} + m_{i,t}^{V,\text{obs}} m_{i,t+1}^{V,\pi},$$

and penalizes their value differences using

$$J_{\text{bd}}(\phi) = \mathbb{E}_{i,t} \left[m_{i,t}^{\text{bd}} (V_{\phi}(s_{i,t}) - V_{\phi}(s_{i,t+1}))^2 \right].$$

The mask is zero away from adjacent generated/observation transitions, so non-boundary positions do not contribute to J_{bd} . The final value objective is

$$L_{\text{value}}^{\text{OAS}}(\phi) = L_{\text{value}}^{\text{OVR}}(\phi) + \lambda_{\text{sm}} J_{\text{bd}}(\phi),$$

$$\lambda_{\text{sm}} \geq 0.$$

These definitions imply three limiting properties. First, OAS reduces to the original token-level PPO value objective when there are no observation tokens, or when $\lambda_{\text{obs}} = \lambda_{\text{sm}} = 0$ and $\alpha_i^{\text{obs}} = 1$. Second, as the observation-to-generation length ratio increases, observation-token value drift and the observation-residual contribution to generated-token advantages are reduced together. Third, boundary smoothing acts only on adjacent generated/observation transitions and reduces boundary-gap effects on later advantage estimates.

D Experimental Setup Details

D.1 Experimental Setup Details

We use a Search-R1-style search-tool reinforcement learning framework. During reasoning, the model generates search actions. The environment returns observation tokens according to the search request, and the external evidence is inserted into the subsequent context. The model then produces a final answer after several rounds of search and reasoning. We keep the search environment, data splits, and training budget fixed when comparing stabilization methods.

The datasets are NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), and HotpotQA (Yang

Dataset	Scope/source	Train	Dev	Test/eval
NQ	English factoid QA / wiki	79,168	8,757	3,610
TriviaQA	English factoid QA / wiki+web	78,785	8,837	11,313
HotpotQA	English multi-hop QA / wiki	90,447	7,405	7,405

Table 6: Dataset documentation and split statistics. HotpotQA uses the development split for evaluation because the processed benchmark does not provide a separate test split in our setup.

et al., 2018). NQ and TriviaQA evaluate open-domain factoid question answering. TriviaQA provides a larger trivia-style QA benchmark. HotpotQA is a multi-hop open-domain QA dataset. It tests whether the method remains stable when answers require combining multiple evidence pieces. To study retrieval-budget sensitivity, we use Top- k values of 2, 3, and 5. Each search therefore returns 2, 3, or 5 candidate evidence passages.

Table 6 summarizes the dataset documentation and split statistics used in our experiments. All three benchmarks are English QA datasets built from Wikipedia or web-derived sources. They cover open-domain factoid and multi-hop question answering rather than demographic-group representation.

The main experiments use Qwen2.5-3B-Instruct as the base model, and we also report results with Qwen2.5-1.5B-Instruct. Final task performance is measured by exact match (EM). Training dynamics are measured by nVDD, ATE, AUS, and OAGC.

The base models have approximately 3B and 1.5B parameters, respectively, as indicated by their Qwen2.5 model names. The main experiments use Ray, vLLM rollout, and FSDP training on one node with 8 H20 GPUs per run. Each reported experimental setting was run twice, with about 250 training steps per run. The reported tables contain 72 main 3B settings, 6 additional 3B ablation/process settings, and 72 main-text 1.5B settings. A 3B run took about 10 hours, and a 1.5B run took about 6 hours. This gives an estimated total compute budget of about 19,400 H20 GPU-hours.

Unless otherwise stated, reported EM and diagnostic values are averages over the two repeated runs for each experimental setting, rather than maxima selected across runs. The best and second-best markings in the tables are based on the reported means. In Tables 2 and 3, each cell reports mean (sample SD) over the two runs ($n = 2$), where sample SD is computed with the $n - 1$ denominator.

Implementation uses the Search-R1-style training stack with Ray for distributed execution, vLLM

Item	Value
Training algorithm	PPO with GAE and Search-R1-style search interaction
Training steps	About 250 steps per reported run
Batch sizes	Train 512; validation 256; PPO mini-batch 256
Actor update	Learning rate 1×10^{-6} ; PPO epochs 1; clip ratio 0.2; entropy coefficient 0.001
Critic update	Learning rate 1×10^{-5} ; value clip 0.5; critic micro-batch 8–32
GAE / KL	$\gamma = 1.0$; $\lambda = 1.0$; fixed KL coefficient 0.001
Generation limits	Max prompt 4,096; response 500; observation 500; max turns 2
Sampling	Temperature 1.0; top- $p = 0.95$; top- $k = -1$
Retrieval budgets	Top- $k \in \{2, 3, 5\}$
OAS coefficients	$\alpha_{\min} = 0.2000$; $\beta = 0.2812$; $\lambda_{\text{obs}} = 1.0000$; $\lambda_{\text{sm}} = 0.0277$

Table 7: Core experimental setup and hyperparameters. We did not perform a broad hyperparameter search; shared values were kept fixed across datasets and methods, while retrieval budget and method-specific stabilization components were varied as described in the experiments.

for rollout generation, and PyTorch FSDP for actor, reference, and critic workers. Evaluation uses the framework’s rule-based exact-match scorer after dataset-specific answer normalization. We do not use external linguistic preprocessing packages such as NLTK, spaCy, or ROUGE for the reported EM results. The retriever is queried through the Search-R1 retrieval server with the Top- k budgets in Table 7; all compared methods within a setting use the same data split, retrieval budget, generation limits, and evaluation scorer.

The baselines include the following methods. Search-R1 PPO is the base Search-R1 search-tool RL method trained with PPO (Jin et al., 2025; Schulman et al., 2017). Search-R1 GRPO replaces PPO with group-relative policy optimization in the same search-tool framework. It provides a critic-free comparison (Jin et al., 2025; Shao et al., 2024). DAPO improves large-scale RL stability with decoupled clipping, dynamic sampling, token-level policy gradients, and length penalties (Yu et al., 2025). KL-Cov adds KL constraints to high-covariance tokens following a covariance-based analysis of policy-entropy change (Cui et al., 2025). DAR constructs advantage-regression weights from online rewards. It then performs stable policy improvement through weighted supervised fine-tuning (He et al., 2025). I-PPO iteratively fits a Q function. It uses the fitted Q function to guide later PPO updates for multi-turn dialogue or trajectory optimization (Jiang et al., 2025).

D.2 Computational Overhead

We measure the computational overhead of OAS using a matched comparison with Search-R1 PPO on NQ. Both methods use Qwen2.5-1.5B-Instruct, Top-3 retrieval, four NVIDIA H20 GPUs, and the same random seed, batch configuration, sequence limits, validation schedule, checkpoint policy, and 250-step training budget. Average step time is computed over all 250 training steps, and total training time includes the scheduled validation and checkpoint operations. Table 8 reports the resulting computational costs.

Method	Step time (s)	Total time (h)	Sampled peak GPU (GiB)	Non-ckpt. storage (MiB)
Search-R1 PPO	91.89	6.42	94.97	0.862
OAS	116.07	8.10	94.16	0.857

Table 8: Computational overhead on NQ with Qwen2.5-1.5B-Instruct and Top-3 retrieval using four NVIDIA H20 GPUs. Peak memory is the maximum sampled nvidia-smi memory usage during training. Non-checkpoint storage includes training logs, configuration files, and tracking metadata; model checkpoints and shared data or retrieval-index assets are excluded.

Among the measured costs, the main overhead of OAS is additional training time. OAS increases the average step time from 91.89 to 116.07 seconds, corresponding to a 26.32% increase, and increases the total training time from 6.42 to 8.10 hours. In contrast, its sampled peak GPU memory and non-checkpoint storage remain comparable to those of PPO. This result is consistent with the design of OAS: it introduces no additional model parameters, layers, or forward passes, while its residual-decay, old-value-anchoring, and boundary-smoothing terms add only $O(T)$ token-wise computation for a trajectory of length T . Since the reported time includes rollout generation, retrieval, validation, and checkpoint operations, the measured increase represents the end-to-end training overhead under this configuration.

D.3 OAS Coefficient Selection

Table 7 reports the method-specific OAS defaults. The parameter $\alpha_{\min}$ sets the minimum observation-residual retention. The parameter β was determined from the median observation/generated-token ratio on NQ trajectories. The regularization coefficients were scaled toward 5% of the base critic loss using a short probe without task outcomes. All OAS coefficients were fixed before formal evaluation.

D.4 OAS Coefficient Sensitivity

We evaluate the sensitivity of the OAS coefficients using a one-at-a-time design. Observation residual decay is jointly controlled by $\alpha_{\min}$ and β , while boundary smoothing is controlled by λ_{sm} . For each of these three coefficients, we evaluate $0.5\times$ and $2\times$ its default while holding all other settings fixed. The coefficient λ_{obs} controls the separate old-value anchoring mechanism and remains fixed at its default. The evaluated values were fixed before inspecting the sensitivity results.

The study uses Qwen2.5-3B, NQ, and Top-3 retrieval. It includes six one-at-a-time variants together with the default OAS configuration as a reference.

Configuration	Final EM	nVDD	OAGC
Default OAS	0.3760	0.0820	0.0880
$\alpha_{\min}, 0.5\times$	0.3352	0.0677	0.0795
$\alpha_{\min}, 2\times$	0.3794	0.0803	0.1591
$\beta, 0.5\times$	0.3077	0.1790	0.3821
$\beta, 2\times$	0.3672	0.0752	0.0899
$\lambda_{\text{sm}}, 0.5\times$	0.3727	0.0908	0.1024
$\lambda_{\text{sm}}, 2\times$	0.3758	0.0511	0.0746

Table 9: One-at-a-time OAS coefficient sensitivity on Qwen2.5-3B, NQ, and Top-3 retrieval. Each non-default row changes only the named coefficient; λ_{obs} remains fixed at its default.

Table 9 reveals an asymmetric sensitivity profile. Halving β produces the largest degradation, reducing Final EM from 0.3760 to 0.3077 and increasing nVDD/OAGC from 0.0820/0.0880 to 0.1790/0.3821, whereas doubling it remains close to the default. In contrast, λ_{sm} is comparatively insensitive within the tested range: both variants preserve Final EM (0.3727–0.3758), and the $2\times$ setting further lowers nVDD/OAGC to 0.0511/0.0746. The coefficient $\alpha_{\min}$ governs a performance–stability trade-off: halving it improves the stability diagnostics but reduces Final EM to 0.3352, while doubling it preserves Final EM but raises OAGC to 0.1591. Taken together, the tested lower- β setting is unfavorable, λ_{sm} requires little fine tuning over this range, and $\alpha_{\min}$ controls the balance between task performance and stability; the frozen default therefore provides a balanced operating point among the evaluated variants.

D.5 Artifact Use and Data Statement

We use existing datasets, models, and software artifacts only for research evaluation. Our use of NQ, TriviaQA, HotpotQA, FEVER, Qwen2.5,

Run	EM / nVDD	ATE / OAGC
PPO, step 150	0.285 / 0.040	2.540 / 0.073
PPO, step 250	0.278 / 0.215	10.130 / 0.328
OAS, step 150	0.266 / 0.047	0.010 / 0.053
OAS, step 250	0.297 / 0.060	4.030 / 0.074

Run	Searches / saturation	Miss	No search
PPO, step 150	1.116 / 0.137	0.512	0.021
PPO, step 250	1.673 / 0.357	0.514	0.039
OAS, step 150	1.060 / 0.073	0.493	0.011
OAS, step 250	1.154 / 0.127	0.509	0.017

Table 10: Time-aligned task, training-diagnostic, and retrieval-behavior statistics on the fixed Qwen2.5-1.5B NQ Top-3 evaluation set. The miss column reports an answer-string retrieval-miss proxy.

and Search-R1-style training code is intended for benchmarking and methodological analysis. We do not redistribute derived datasets, model checkpoints, or artifacts beyond the research context of this submission.

We do not collect new user data or release new raw-text datasets. Our experiments use existing public QA and fact-verification benchmarks, and we report only aggregate evaluation results. Since these benchmarks may contain naturally occurring entity names or web-derived text, we do not redistribute dataset contents, model checkpoints, or derived raw examples with this submission.

E Detailed Retrieval-Behavior Diagnostics

To complement EM with behavior-level evidence, we evaluate a fixed Qwen2.5-1.5B NQ Top-3 evaluation set at steps 150 and 250. Table 10 aligns EM and nVDD with ATE, OAGC, executed searches, budget saturation, no-search rate, and an answer-string retrieval-miss proxy. The proxy checks whether a retrieved observation lacks a normalized gold-answer alias. nVDD is computed from the full periodic-validation sequence.

Table 10 supports a focused behavioral conclusion: OAS stabilizes how the policy uses search as training proceeds. By step 250, OAS combines higher EM and lower drawdown (0.297/0.060 versus 0.278/0.215) with fewer executed searches and substantially lower budget saturation (1.154/0.127 versus 1.673/0.357). Its no-search rate is also lower (0.017 versus 0.039), arguing against the simple explanation that stability comes from using the tool less. The stability gain is therefore accompanied by more restrained allocation of the search budget—especially fewer repeated or budget-exhausting searches—while active tool engagement is pre-

served.

However, the nearly unchanged retrieval-miss proxy (0.509 versus 0.514) limits this conclusion to search-budget control and does not establish improved retrieval relevance or answer faithfulness.