SR-SEARCH: STRUCTURED ROUTING OF SEARCH AND REFINE INFORMATION GAINS FOR FINE-GRAINED CREDIT ASSIGNMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning (RL) has become a dominant paradigm for enhancing retrieval-augmented generation (RAG) in large language models. However, existing frameworks such as Search-R1 rely on trajectory-level outcome rewards assigned to all generated tokens along an execution path. This coarse and sparse signal cannot credit intermediate decisions, and in complex multi-hop question answering (QA) it fails to isolate whether a specific retrieval acquired relevant information or whether the subsequent refinement preserved and used that evidence. To address these limitations, we propose SR-Search, a fine-grained credit-assignment framework for search agents trained via Group Relative Policy Optimization (GRPO). Within each valid search–refine cycle, the policy evaluates the log-probability of the ground-truth answer before search, after retrieval, and after refinement; the resulting likelihood differences define Search Information Gain (SIG) and Refinement Information Gain (RIG), respectively, without any auxiliary reward model. SR-Search routes SIG to search actions and RIG to refinement actions, and further introduces a short-horizon delayed advantage that propagates refinement gains backwards to optimize the preceding retrieval decisions. Across seven QA benchmarks, SR-Search outperforms outcome-reward baseline AutoRefine by 3.7 points and the strongest step-reward baseline GiGPO by 2.1 points. These results demonstrate that temporally localized credit assignment can improve both retrieval and evidence refinement in RL-based search agents.

1 INTRODUCTION

Large language models (LLMs) have demonstrated strong capabilities across a wide range of tasks, yet their parametric knowledge is static and bounded by the training corpus. Retrieval-augmented generation (RAG) mitigates this limitation by grounding generation in external corpora (Lewis et al., 2020). Recent agentic search systems go further, allowing an LLM to iteratively issue queries, observe retrieved documents, and update its reasoning within a multi-turn interaction. RL frameworks such as Search-R1 and ReSearch train such search behaviors directly from task feedback (Jin et al., 2025a; Chen et al., 2025), and AutoRefine introduces an explicit refinement action after retrieval (Shi et al., 2025). While this paradigm makes retrieval adaptive, it turns search-augmented reasoning into a sequential decision problem where effective learning requires properly crediting the intermediate actions that acquire and transform evidence.

Most RL-based search agents rely on trajectory-level outcome rewards (Jin et al., 2025a; Shi et al., 2025). Under GRPO (Shao et al., 2024), the terminal reward is standardized across rollouts and uniformly assigned to all generated tokens. While critic-free, this outcome signal indicates which rollouts succeed, not why. In multi-hop QA, failure can stem from flawed queries, unhelpful documents, or discarded evidence, while success may mask redundant searches. Sharing an identical advantage across these heterogeneous actions creates a severe mismatch between reward and behavior granularity.

Recent studies alleviate this mismatch with denser supervision, but they assign credit at coarser units—a step, a repeated environment state, a retrieval step, or an entire interaction turn (detailed in §2). However, existing methods neither distinguish evidence acquisition via search from its

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.

subsequent refinement, nor leverage the downstream utility of refinement to credit the preceding retrieval.

Our key observation is that the value of a retrieval is rarely immediate: documents often mix relevant facts with distracting noise, yielding marginal or negative immediate likelihood gains until refinement distills the evidence. This creates a short-horizon dependency where search provides evidence and refinement unlocks its utility. An effective credit mechanism should thus evaluate both stages independently while routing bounded hindsight feedback from refinement back to search.

To this end, we propose **SR-Search**, a fine-grained credit-assignment framework that realizes this decomposition within GRPO. For each search–refine cycle, the old policy $\pi_{\theta_{\text{old}}}$ evaluates the ground-truth log-probability under three successive contexts; the resulting gains are treated as constants during optimization. The change before and after document acquisition defines the *search information gain* (SIG), and the change before and after refinement defines the *refinement information gain* (RIG). SR-Search normalizes SIG and RIG independently within each rollout group, routes the calibrated credits to the responsible action spans, and additionally propagates a bounded fraction of the same-cycle RIG back to the preceding search as a short-horizon delayed advantage. Figure 1 previews the per-benchmark performance against the strongest RL baselines.

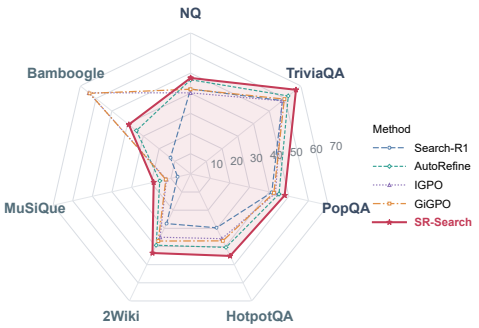


Figure 1: Exact Match (EM, %) across seven QA benchmarks. Base models are used where available; instruct variants are used for IGPO and GiGPO.

These local advantages augment the trajectory-level outcome advantage without displacing it: general reasoning tokens still optimize answer accuracy, documents remain masked, and delayed routing reuses existing RIG calculations without requiring extra critic or process models. Our main contributions are:

- We propose **SR-Search**, a fine-grained, critic-free credit-assignment framework for RL-based search agents that mitigates sparse trajectory-level feedback and action-level credit mismatch.
- We formulate search and refinement information gains (SIG and RIG) from the policy's ground-truth log-probability shifts, and introduce a short-horizon delayed advantage that routes downstream refinement gains back to upstream search as an explicit cross-action credit channel.
- Extensive evaluations on seven QA benchmarks show that SR-Search improves average EM by 3.7 points over AutoRefine and outperforms strong step-reward baselines, especially on multi-hop reasoning.

2 Related Work

Agentic Search via Prompting and In-Context Reasoning. Early retrieval-augmented systems treat retrieval as an inference-time heuristic: RAG conditions generation on retrieved passages (Lewis et al., 2020), while ReAct (Yao et al., 2023) and Search-o1 (Li et al., 2025) interleave reasoning traces with tool calls. Relying on hand-crafted prompts and rules, their query timing and refinement decisions remain brittle under noisy or ambiguous evidence.

Reinforcement Learning for Search-Augmented Agents. Subsequent work optimizes search policies directly from task feedback. Toolformer learns API use through self-supervised filtering of generated calls (Schick et al., 2023), while Self-RAG trains a generator to predict retrieval and critique tokens using offline supervision (Asai et al., 2024). RL frameworks such as Search-R1 (Jin et al., 2025a) and ReSearch (Chen et al., 2025) train multi-turn retrieval policies via GRPO (Shao et al., 2024). AutoRefine introduces explicit knowledge refinement (Shi et al., 2025). However, aggregate rollout rewards do not directly identify which search or refinement action produced a useful intermediate state.

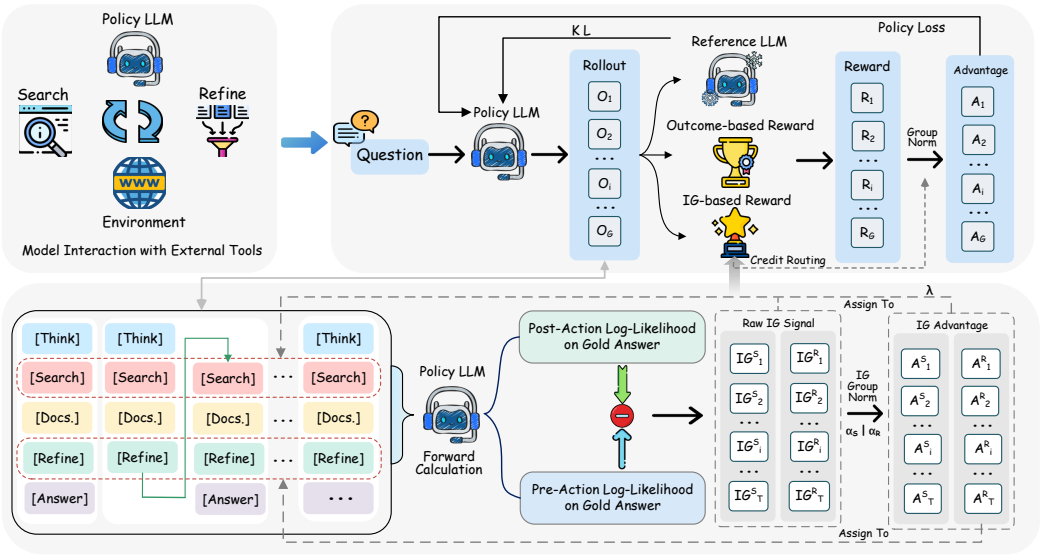


Figure 2: Overview of SR-Search. The old policy $\pi_{\theta_{\text{old}}}$ evaluates the ground-truth log-probability at three checkpoints per cycle, yielding SIG and RIG. Calibrated gains are routed to search and refinement spans, with a bounded fraction of RIG propagated backwards to search via short-horizon delayed advantage, augmenting the trajectory-level GRPO outcome advantage.

Process-Level Supervision and Credit Assignment in Search Agents. To provide denser supervision, recent studies move the training signal inside the trajectory. Process reward models score intermediate steps but typically require step-level annotations or separately trained verifiers (Lightman et al., 2024). Information-gain and group-relative approaches instead derive fine-grained supervision without additional annotations. Specifically, StepSearch (Zheng et al., 2025) and IGPO (Wang et al., 2026) formulate information-gain rewards at the step and turn levels, respectively, while GiGPO (Feng et al., 2025) constructs comparison groups from repeated anchor states. MR-Search (Xiao et al., 2026) improves exploration through cross-episode reflection, while SmartSearch (Wen et al., 2026) optimizes intermediate queries through process rewards and selective query refinement. Yet, existing methods supervise entire turns or repeated states without isolating evidence acquisition from refinement, nor linking downstream refinement utility back to the initial search. SR-Search addresses this with localized SIG/RIG credit and bounded delayed advantage routing.

3 SR-Search

We present SR-Search below, with Figure 2 providing an overview: §3.1 formulates the search–refine interaction and reviews GRPO; §3.2–3.4 define, normalize, and route the dual-stage information gains; §3.5 gives the policy objective and overhead analysis. The full procedure is in Appendix C.

3.1 Preliminaries

Search–refine interaction. Following prior work (Jin et al., 2025a; Shi et al., 2025), we formulate search-augmented reasoning as a multi-turn interaction between a policy LLM and a search environment. Given a question q with ground-truth answer a, policy π_θ generates a rollout o_i that interleaves reasoning with search and refinement. In cycle k, the policy issues query $q_{i,k}$, receives retrieved documents $d_{i,k}$, and optionally produces a refinement $r_{i,k}$ that distills answer-relevant evidence (Shi et al., 2025). The accumulated context informs subsequent reasoning and query generation, allowing the policy to gather additional evidence before producing its final answer $\hat{a}_i$. Search queries and refinements are trainable actions, while retrieved documents are externally supplied observations. These observations remain available as conditioning context but are masked from the policy loss, so optimization applies only to model-generated tokens (Jin et al., 2025a).

Outcome-supervised GRPO. For each question, GRPO samples a group of G rollouts $\{o_i\}_{i=1}^{G}$ from the old policy $\pi_{\theta_{\text{old}}}$ and evaluates their final answers. Each rollout receives a terminal reward

based on answer correctness, providing supervision without a learned value critic. We use token-level F1 to capture partial answer overlap, balancing the precision and recall of predicted tokens relative to the ground truth. Treating both answers as bags of tokens, the reward is

$$R_i = \mathrm{F1}(\hat{a}_i, a) = \frac{2\,|\hat{a}_i \cap a|}{|\hat{a}_i| + |a|}, \tag{1}$$

where $|\hat{a}_i \cap a|$ counts matching tokens with multiplicity, and $|\hat{a}_i|$ and $|a|$ denote the respective answer lengths. The rewards are standardized within the group to obtain the trajectory-level outcome advantage

$$A_i^{\mathrm{orm}} = \frac{R_i - \mathrm{mean}\left(\{R_j\}_{j=1}^{G}\right)}{\mathrm{std}\left(\{R_j\}_{j=1}^{G}\right)}, \tag{2}$$

which is shared by all trainable tokens in o_i. This group-relative signal favors rollouts with more accurate answers, but assigns identical credit to their potentially heterogeneous intermediate actions. It therefore cannot distinguish successful evidence acquisition from effective refinement, motivating the action-level credit assignment introduced below.

3.2 Dual-Stage Information Gain

SR-Search quantifies each intermediate action by measuring how it changes the policy's log-probability of the ground-truth answer, without relying on an auxiliary reward model or external scorer. Specifically, we use the old policy $\pi_{\theta_{\mathrm{old}}}$, which also generates the current rollout group, to evaluate the log-probability of the ground-truth answer conditioned on the question and a rollout prefix:

$$\log \pi_{\theta_{\mathrm{old}}}(a \mid c) = \sum_{j=1}^{|a|} \log \pi_{\theta_{\mathrm{old}}}\left(a_j \mid c, a_{<j}\right), \tag{3}$$

where a_j is the j-th token of a. All information-gain values are computed before the policy update and treated as constants during normalization, credit routing, and optimization. The reference policy π_{ref} is used only for KL regularization.

For the k-th cycle of rollout o_i, let $o_{i,<q_{i,k}}$ denote the prefix of o_i ending immediately before the search query $q_{i,k}$, and let $o_{i,\le d_{i,k}}$ and $o_{i,\le r_{i,k}}$ denote the prefixes ending at the retrieved documents $d_{i,k}$ and at the refinement $r_{i,k}$, respectively.

Search information gain. SIG measures how much the search query and its retrieved documents increase the policy's confidence in the ground-truth answer:

$$\mathrm{IG}^{S}_{i,k} = \log \pi_{\theta_{\mathrm{old}}}\left(a \mid q,\, o_{i,\le d_{i,k}}\right) - \log \pi_{\theta_{\mathrm{old}}}\left(a \mid q,\, o_{i,<q_{i,k}}\right). \tag{4}$$

A positive $\mathrm{IG}^{S}_{i,k}$ indicates that the acquired evidence increases the policy's confidence in the ground-truth answer.

Refinement information gain. When the same cycle contains a refinement, RIG measures the additional change induced by refining the retrieved content:

$$\mathrm{IG}^{R}_{i,k} = \log \pi_{\theta_{\mathrm{old}}}\left(a \mid q,\, o_{i,\le r_{i,k}}\right) - \log \pi_{\theta_{\mathrm{old}}}\left(a \mid q,\, o_{i,\le d_{i,k}}\right). \tag{5}$$

Thus, SIG credits evidence acquisition, while RIG rewards information distillation. Gains are marked valid only on syntactically complete action spans.

3.3 Group-wise Gain Normalization

Raw information gains are differences of log-likelihoods, whose scale varies considerably across questions and between the two stages. Analogous to the group-relative treatment of outcome rewards in Eq. (2), SR-Search standardizes SIG and RIG separately within each rollout group. For $\star \in \{S, R\}$, let $\mu^\star$ and $\sigma^\star$ denote the mean and standard deviation of all valid gains $\{\mathrm{IG}^{\star}_{i,k}\}$ collected across the G rollouts of the same question, without matching cycle indices across rollouts. Each gain is standardized and then clipped asymmetrically:

$$\widetilde{\mathrm{IG}}^{\star}_{i,k} = \mathrm{clip}\left(\frac{\mathrm{IG}^{\star}_{i,k} - \mu^\star}{\sigma^\star},\, -c_-,\, c_+\right), \tag{6}$$

where $c_-, c_+ > 0$ are the clipping bounds. The two stages are normalized independently because they reflect different context changes and noise characteristics; signed clipping (as opposed to positive-only filtering) preserves informative negative credits, while the asymmetric bounds ($c_- < c_+$) prevent persistently negative SIG from suppressing retrieval entirely. Groups with fewer than two valid gains or negligible variance receive zero credit.

3.4 Action-Level Credit Routing

SR-Search converts the calibrated gains into local advantages and routes each of them to the action responsible for it. The direct search and refinement advantages are defined as

$$A^S_{i,k} = \alpha_S \, \widetilde{\mathrm{IG}}^S_{i,k}, \qquad A^R_{i,k} = \alpha_R \, \widetilde{\mathrm{IG}}^R_{i,k}, \tag{7}$$

where α_S and α_R control the strength of the two channels.

Short-horizon delayed advantage. Immediate SIG can underestimate a search whose evidence becomes useful only after refinement extracts it from noisy documents. SR-Search therefore routes a fraction of the same-cycle RIG back to the preceding search:

$$A^{R\to S}_{i,k} = \lambda \, A^R_{i,k}, \tag{8}$$

where $\lambda \geq 0$ controls the delayed credit strength. This route never crosses cycle boundaries.

Composite token-level advantage. Let $\mathcal{S}_{i,k}$ and $\mathcal{R}_{i,k}$ denote the token index sets of the k-th search and refinement spans (each covering the opening tag, content, and closing tag). The advantage of token t is

$$A_{i,t} = \begin{cases} A^{\mathrm{orm}}_i + A^S_{i,k} + A^{R\to S}_{i,k}, & t \in \mathcal{S}_{i,k}, \\ A^{\mathrm{orm}}_i + A^R_{i,k}, & t \in \mathcal{R}_{i,k}, \\ A^{\mathrm{orm}}_i, & \text{otherwise}. \end{cases} \tag{9}$$

Action tokens receive uniform advantages without length scaling. Covering both content and syntactic boundary tags stabilizes structural formatting during policy optimization.

3.5 Policy Optimization and Computational Cost

SR-Search inserts the composite advantage into the standard clipped GRPO objective with KL regularization:

$$\begin{aligned} \mathcal{J}_{\mathrm{SR}}(\theta) = \mathbb{E}_{(q,a)\sim\mathcal{D},\, \{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\mathrm{old}}}(\cdot|q)} \Bigg[\frac{1}{G}\sum_{i=1}^{G} \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \min \Bigg(\frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\mathrm{old}}}(o_{i,t} \mid q, o_{i,<t})} A_{i,t}, \\ \mathrm{clip}\left(\frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\mathrm{old}}}(o_{i,t} \mid q, o_{i,<t})}, 1-\varepsilon, 1+\varepsilon \right) A_{i,t} \Bigg) - \beta \, \mathbb{D}_{\mathrm{KL}}\big(\pi_\theta \,\|\, \pi_{\mathrm{ref}}\big) \Bigg], \end{aligned} \tag{10}$$

where ε is the clipping ratio, β is the KL regularization coefficient toward π_{ref}, and retrieved document tokens are masked from the loss.

The primary overhead comes from the additional scoring passes of $\pi_{\theta_{\mathrm{old}}}$ across the three context checkpoints of each valid cycle, batched across rollouts; delayed routing directly reuses computed RIG without additional forward passes (quantified in §5.5).

4 Experiments

In this section, we conduct extensive experiments to answer the following research questions (RQs):

RQ1: Does SR-Search improve retrieval-augmented question answering over outcome-reward and step-reward RL baselines?

RQ2: How does each component, namely direct SIG, direct RIG, the short-horizon delayed advantage, and the group-wise gain normalization, contribute to the overall performance?

RQ3: How does SR-Search shape the search and refinement behaviors of the policy during training?

RQ4: Does the benefit of fine-grained credit assignment persist when scaling to larger backbones?

RQ5: How sensitive is SR-Search to the credit-channel hyperparameters, and does it accelerate convergence?

RQ6: How much additional training cost does the fine-grained credit assignment incur?

4.1 Experiment Setup

Datasets. We evaluate on seven benchmarks: three single-hop datasets (Natural Questions (NQ) (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023)) and four multi-hop datasets (HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), Bamboogle (Press et al., 2023)). Following the setting of prior works (Jin et al., 2025a; Shi et al., 2025), we train jointly on NQ and HotpotQA training splits, evaluating on test/dev splits of all seven benchmarks to test in-domain and out-of-domain generalization using Exact Match (EM) (details in Appendix A.1).

Baselines. We compare SR-Search against non-retrieval methods (Direct, SFT, R1 (Guo et al., 2025)), single-hop retrieval (Naive RAG (Lewis et al., 2020)), and multi-hop retrieval agents: prompting-based IRCoT (Trivedi et al., 2023) and Search-o1 (Li et al., 2025), outcome-reward Search-R1 (Jin et al., 2025a) and AutoRefine (Shi et al., 2025), and step-reward methods StepSearch (Zheng et al., 2025), IGPO (Wang et al., 2026), and GiGPO (Feng et al., 2025). All retrieval methods share the same retriever, Wikipedia corpus, top-k documents, and training data.

Implementation Details. We use Qwen2.5-3B (Base/Instruct) (Qwen Team, 2024) with the 2018 Wikipedia dump (Karpukhin et al., 2020) from FlashRAG (Jin et al., 2025b) and E5-base-v2 (Wang et al., 2022) as the retriever, following Search-R1 (Jin et al., 2025a). Each query retrieves top-3 passages (max 5 searches per rollout). We train via veRL (Sheng et al., 2025) with GRPO (group size 5, batch size 256, lr 1×10^{-6}, β=0.001, ε=0.2) on 8 A100-80GB GPUs. We set α_S=0.05, α_R=0.1, (c_-, c_+)=$(1.0, 2.0)$, and λ=0.1 (Appendix A.2).

4.2 Overall Performance

Table 1 presents the results across the seven benchmarks, and Figure 1 visualizes the per-benchmark comparison among the strongest RL methods. We highlight the following observations (RQ1):

RL-based methods consistently outperform prompting-based baselines. All RL methods substantially outperform non-RL approaches (direct generation, naive RAG, IRCoT, Search-o1), confirming the necessity of policy optimization for multi-hop retrieval (Jin et al., 2025a; Shi et al., 2025).

SR-Search achieves the best overall performance. With Qwen2.5-3B-Base, SR-Search attains an average EM of 0.442 and SR-Search-Instruct 0.432, improving over AutoRefine by 0.037 and 0.036 and surpassing the strongest step-reward baseline GiGPO (0.421), with the best or second-best result on six of seven benchmarks. As SR-Search shares the same interaction template, retrieval environment, and outcome reward as AutoRefine, the gain is directly attributable to action-level credit assignment.

Multi-hop benchmarks benefit the most. SR-Search improves over AutoRefine by 0.047 on HotpotQA and 0.043 on 2Wiki (11.6% and 10.9% relative), where multiple interdependent search–refine cycles make trajectory-level credit mismatch most severe (§4.3).

Step-reward methods remain competitive only on narrow benchmarks. SR-Search outperforms step-reward baselines on six datasets. SR-Search surpasses all step-reward competitors except on Bamboogle (125 instances), where IGPO/GiGPO achieve higher EM; however, SR-Search leads by 2.1 points on overall average and consistently excels on multi-hop benchmarks.

4.3 Ablation Study

To answer RQ2, we ablate the four components of SR-Search (Table 2).

Table 1: Overall performance (EM) of SR-Search and baselines on seven QA benchmarks with Qwen2.5-3B backbones. The best and second-best results are highlighted in **bold** and underline, respectively. † marks the in-domain datasets whose training splits are used for training; the remaining datasets are out-of-domain. "–" denotes unavailable results.

	Single-Hop QA			Multi-Hop QA				
Methods	NQ†	TriviaQA	PopQA	HotpotQA†	2Wiki	MuSiQue	Bamboogle	**Avg.**
w/o Retrieval								
Direct Generation	0.106	0.288	0.108	0.149	0.244	0.020	0.024	0.134
SFT	0.249	0.292	0.104	0.186	0.248	0.044	0.112	0.176
R1-Instruct	0.210	0.449	0.171	0.208	0.275	0.060	0.192	0.224
R1-Base	0.226	0.455	0.173	0.201	0.268	0.055	0.224	0.229
w/ Single-Hop Retrieval								
Naive RAG	0.348	0.544	0.387	0.255	0.226	0.047	0.080	0.270
w/ Multi-Hop Retrieval (Prompting-Based)								
Search-o1	0.238	0.472	0.262	0.221	0.218	0.054	0.320	0.255
IRCoT	0.111	0.312	0.200	0.164	0.171	0.067	0.240	0.181
w/ Multi-Hop Retrieval (Outcome-Reward RL)								
Search-R1-Instruct	0.397	0.565	0.391	0.331	0.310	0.124	0.232	0.336
Search-R1-Base	0.421	0.583	0.413	0.297	0.274	0.066	0.128	0.312
AutoRefine-Instruct	0.436	0.597	0.447	0.404	0.380	0.169	0.336	0.396
AutoRefine-Base	0.467	0.620	0.450	0.405	0.393	0.157	0.344	0.405
w/ Multi-Hop Retrieval (Step-Reward RL)								
StepSearch-Instruct	–	–	–	0.345	0.320	0.174	0.344	–
StepSearch-Base	–	–	–	0.329	0.339	0.181	0.328	–
IGPO-Instruct	0.402	0.581	0.432	0.357	0.349	0.123	**0.645**	0.413
GiGPO-Instruct	0.420	0.595	0.424	0.369	0.370	0.126	0.641	0.421
SR-Search-Instruct (Ours)	0.465	**0.672**	0.455	0.446	0.421	0.179	0.383	0.432
SR-Search-Base (Ours)	**0.476**	0.669	**0.479**	**0.452**	**0.436**	**0.188**	0.392	**0.442**

Table 2: Ablation study (EM) of SR-Search with Qwen2.5-3B-Base. "w/o SIG" and "w/o RIG" remove the corresponding direct information-gain channel; "w/o SHDA" removes the short-horizon delayed advantage that propagates refinement gains back to the preceding search; "w/o Norm" replaces the group-wise gain normalization of Eq. (6) with the raw log-likelihood differences. The best results are in **bold**.

Methods	NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	**Avg.**
SR-Search	**0.476**	**0.669**	0.479	**0.452**	**0.436**	**0.188**	**0.392**	**0.442**
w/o SIG	0.429	0.636	0.463	0.425	0.409	0.131	0.320	0.402
w/o RIG	0.419	0.630	0.462	0.389	0.369	0.141	0.331	0.392
w/o SHDA	0.421	0.642	**0.484**	0.426	0.405	0.150	0.347	0.411
w/o Norm	0.398	0.624	0.452	0.387	0.360	0.118	0.266	0.372

Every component contributes. With all components, SR-Search reaches an average EM of 0.442. Removing SHDA lowers this value to 0.411; removing SIG or RIG lowers it to 0.402 or 0.392, respectively. Replacing group-wise normalization with raw gains produces the lowest average EM, 0.372. Thus, every ablation reduces aggregate performance, although the full model does not lead on every individual dataset. On PopQA, the variant without SHDA scores 0.484, slightly above the full model's 0.479.

Gain normalization is indispensable. Ablating normalization leads to the largest drop (-0.070 EM), even falling below AutoRefine (0.372 vs. 0.405). Because raw log-likelihood deltas vary widely, unnormalized signals allow high-variance cycles to destabilize gradient updates. This degradation is especially pronounced on multi-hop tasks (-0.084 vs. -0.050 single-hop) where longer trajectories accumulate heterogeneous gains.

Delayed credit makes the two direct channels cooperate. Without SHDA, each gain is confined to its own span, losing 0.031 EM. A search whose evidence becomes useful only after refinement gets a low immediate SIG, so the direct channels alone can pull the policy in conflicting directions; propagating the refinement gain backwards resolves this. The effect concentrates on multi-hop questions (0.035 vs. 0.026 single-hop).

Table 3: Performance (EM) on seven QA benchmarks with Qwen2.5-7B backbones. The best and second-best results are highlighted in **bold** and underline, respectively.

Methods	NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	**Avg.**
Search-R1-Base	0.469	0.627	0.449	0.410	0.272	0.173	0.456	0.408
AutoRefine-Base	0.484	0.659	0.487	0.451	0.405	0.187	0.512	0.455
IGPO-Instruct	0.440	0.601	0.454	0.429	0.425	0.168	0.399	0.417
GiGPO-Instruct	0.464	0.647	0.461	0.416	0.436	0.189	**0.689**	0.472
SR-Search-Instruct (Ours)	0.495	0.680	0.505	**0.491**	0.451	**0.201**	0.504	0.475
SR-Search-Base (Ours)	**0.507**	**0.691**	**0.518**	0.488	**0.457**	0.200	0.513	**0.482**

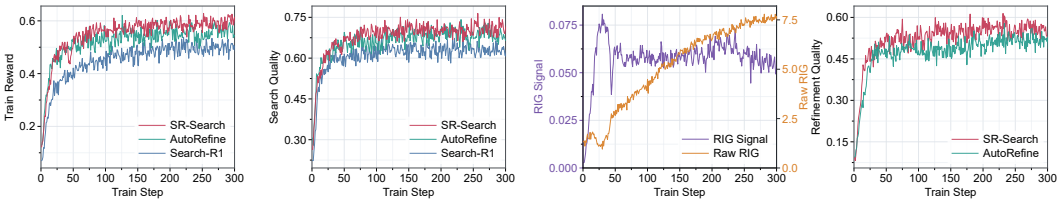


Figure 3: Training dynamics with Qwen2.5-3B-Base: (a) training reward, (b) search quality, (c) calibrated and raw RIG signals, and (d) refinement quality. Search and refinement quality measure the proportions of retrieved documents and refinement outputs containing the ground-truth answer, respectively.

5 Further Analysis

5.1 Training Dynamics

Figure 3 traces how SR-Search reshapes the policy's search and refinement behavior over the course of training. **(a)** SR-Search converges to a higher reward plateau ($\sim$0.60 vs. 0.57 for AutoRefine and 0.50 for Search-R1). **(b)** Search quality stays highest, as SIG rewards queries that raise the answer likelihood. **(c)** The raw RIG grows steadily while the calibrated RIG stabilizes, confirming that group-wise normalization produces a stationary training signal. **(d)** Refinement quality stays above AutoRefine, verifying that RIG improves evidence preservation beyond a trajectory-level reward.

5.2 Model Size

Results on Qwen2.5-7B (Table 3) show consistent gains: SR-Search achieves the best average EM with both Base (0.482) and Instruct (0.475) backbones, outperforming AutoRefine-Base (0.455) by 0.027 and GiGPO-Instruct (0.472) by 0.003, respectively. The Base variant also improves over AutoRefine on HotpotQA by 0.037 and on 2Wiki by 0.052. Together with the 3B results, these gains demonstrate the scalability of fine-grained credit assignment.

5.3 Hyperparameter Sensitivity

To answer RQ5, Figure 4 sweeps α_S, α_R, and λ around default values (α_S=0.05, α_R=0.1, λ=0.1). All curves exhibit smooth unimodal peaks at the default settings. The policy is most sensitive to search gain weight α_S (0.05$\rightarrow$0.1 degrades EM to 0.392) due to document noise amplification; α_R and delayed weight λ demonstrate stable performance across broad intervals.

5.4 Credit Dynamics and Convergence Behavior

Figure 5 further dissects how the credit channels reshape behavior. **(a)** Response length collapses from $\sim$1,600 to 700 tokens in the first 50 steps while reward rises, as positive credit is confined to informative spans and verbose reasoning is pruned. **(b)** Both advantage channels activate rapidly and stay within their clipped ranges; RIG remains above SIG, and neither collapses nor explodes. **(c)** SR-Search reaches 0.50 reward at step 29 vs. 41 (AutoRefine) and 110 (Search-R1); at 0.55, Search-R1 never arrives. The gap widens with the threshold, confirming that fine-grained credit is most valuable where sparse outcome rewards become uninformative.

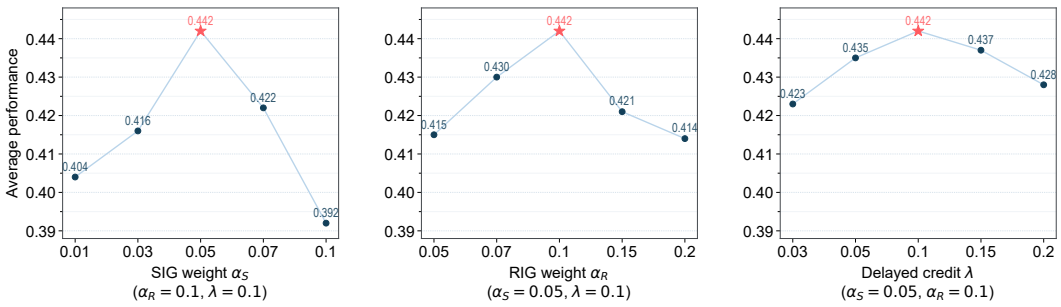


Figure 4: Sensitivity of the average EM to the SIG weight α_S, the RIG weight α_R, and the delayed credit coefficient λ, each varied with the other two fixed at their default values.

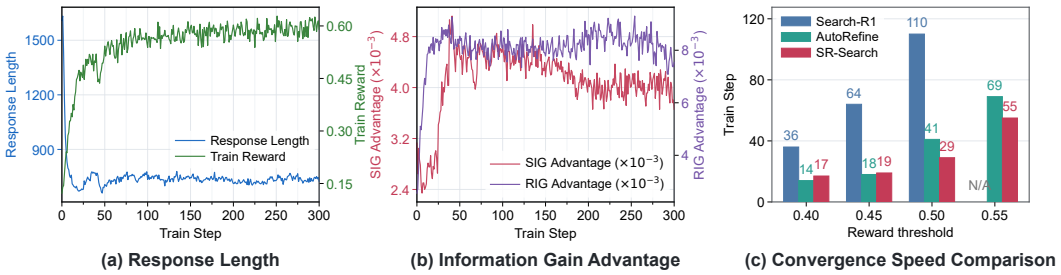


Figure 5: Mechanism analysis: (a) response length vs. reward, (b) SIG/RIG advantage activations, and (c) training steps to reach given reward thresholds ("N/A" means unreached within 300 steps).

5.5 Computational Cost

Figure 6 reports the per-step training costs. Advantage computation rises from 7.8s to 100.8s because $\pi_{\theta_{\text{old}}}$ performs additional scoring passes at the three checkpoints of each valid cycle; no auxiliary model is required.

Crucially, localized credit suppresses redundant searches and prunes verbose thoughts (cf. §5.4), reducing rollout generation time from 141.3s to 85.0s (-39.8%). Reference log-prob and actor update costs also decrease (-12.7% and -9.6%) with shorter sequence lengths. Overall, wall-clock time rises marginally from 239.5s to 266.5s per step (+11.2% net overhead), while adding zero inference overhead.

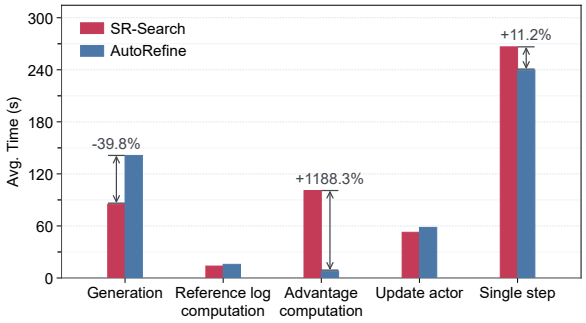


Figure 6: Per-step training time breakdown of SR-Search vs. AutoRefine

6 Conclusion, Limitation and Future Work

We present SR-Search, a fine-grained credit-assignment framework that decomposes each search–refine cycle into SIG and RIG, routes both gains to the responsible action spans, and propagates bounded same-cycle RIG back to the preceding search. On seven QA benchmarks, SR-Search achieves an average EM of 0.442, outperforming the strongest outcome-reward and step-reward baselines with only 11.2% training overhead.

SR-Search requires gold answers and additional policy forward passes for information-gain computation during training; SIG captures the joint effect of query and documents, and delayed routing is limited to same-cycle dependencies. Future work will explore answer-free information-gain proxies, counterfactual retrieval baselines, and adaptive multi-cycle routing to extend the framework beyond labeled QA.

LLM USAGE STATEMENT

Following the ICLR 2027 AI policy for authors, we disclose our use of generative AI in this work. We used a large language model assistant (Gemini 3.8 Flash) for language polishing of author-written text, including grammar correction and improving the fluency of individual paragraphs, as well as for LaTeX formatting assistance such as table and figure environment typesetting. We have not used generative AI tools for research ideation, method design, experiment execution, data analysis, or the generation of any experimental results. All AI-polished text was reviewed and, where necessary, revised by the authors against the original technical content, and all quantitative claims were verified by the authors against the experimental records. We take full responsibility for the final content of this work, including all text, claims, and artifacts.

ETHICS STATEMENT

This work studies reinforcement learning for retrieval-augmented question answering. All experiments are conducted on publicly available academic benchmarks (NQ, TriviaQA, PopQA, HotpotQA, 2WikiMultiHopQA, MuSiQue, and Bamboogle) and a public 2018 Wikipedia corpus, none of which contains private or personally identifiable information collected by us. The proposed method improves the credit assignment of search agents and does not introduce capabilities beyond answering factual questions over a fixed corpus; nevertheless, as with all retrieval-augmented systems, generated answers may inherit factual errors or biases present in the underlying corpus, and downstream deployments should incorporate appropriate verification. We do not foresee direct negative societal impacts specific to this method, and no human subjects or sensitive data were involved.

REPRODUCIBILITY STATEMENT

We have made efforts to ensure the reproducibility of our results. The method is fully specified in §3, with the complete training procedure in Appendix C and the interaction prompt template in Appendix D. Dataset statistics, data splits, and preprocessing follow the public FlashRAG collection and are detailed in Appendix A.1; all training hyperparameters are enumerated in Appendix A.2. Our implementation builds on the open-source veRL framework, and the source code will be released upon publication.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations*, 2024.

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 *Advances in Neural Information Processing Systems*, 2025. doi: 10.52202/085713-2858.

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

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025. doi: 10.1038/s41586-025-09422-z.

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 the 28th International Conference on Computational Linguistics*, 2020.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O. Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. In *Conference on Language Modeling*, 2025a.

Jiajie Jin, Yutao Zhu, Zhicheng Dou, Guanting Dong, Xinyu Yang, Chenghao Zhang, Tong Zhao, Zhao Yang, and Ji-Rong Wen. FlashRAG: A modular toolkit for efficient retrieval-augmented

generation research. In *Companion Proceedings of the ACM on Web Conference 2025*, pp. 737–740, 2025b. doi: 10.1145/3701716.3715313.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 2020.

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. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, 2020.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366*, 2025.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

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 the 61st Annual Meeting of the Association for Computational Linguistics*, 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 the Association for Computational Linguistics: EMNLP 2023*, 2023.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, 2023.

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. HybridFlow: A flexible and efficient RLHF framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, 2025. doi: 10.1145/3689031.3696075.

Yaorui Shi, Sihang Li, Chang Wu, Zhiyuan Liu, Junfeng Fang, Hengxing Cai, An Zhang, and Xiang Wang. Search and refine during think: Facilitating knowledge refinement for improved retrieval-augmented reasoning. In *Advances in Neural Information Processing Systems*, 2025.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 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 the 61st Annual Meeting of the Association for Computational Linguistics*, 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 *International Conference on Learning Representations*, 2026.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

Tongyu Wen, Guanting Dong, and Zhicheng Dou. SmartSearch: Process reward-guided query refinement for search agents. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2026.

Teng Xiao, Yige Yuan, Hamish Ivison, Huaisheng Zhu, Faeze Brahman, Nathan Lambert, Pradeep Dasigi, Noah A. Smith, and Hannaneh Hajishirzi. Meta-reinforcement learning with self-reflection for agentic search. *arXiv preprint arXiv:2603.11327*, 2026.

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 the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Xuhui Zheng, Kang An, Ziliang Wang, Yuhang Wang, and Yichao Wu. StepSearch: Igniting LLMs search ability via step-wise proximal policy optimization. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 21805–21830, 2025. doi: 10.18653/v1/2025.emnlp-main.1106.

A More Experimental Details

A.1 Dataset Details

We conduct all experiments on seven open-domain QA benchmarks drawn from the FlashRAG collection (Jin et al., 2025b), covering both single-hop factoid questions and multi-hop questions that require composing evidence across documents. The training set merges the official training splits of NQ and HotpotQA, so that the policy is exposed to single-hop and multi-hop supervision simultaneously; no training example overlaps with any evaluation split. For evaluation, we use the official test splits when they are publicly available (NQ, TriviaQA, PopQA, and Bamboogle) and the development splits otherwise (HotpotQA, 2Wiki, and MuSiQue). Table 4 summarizes the statistics. Among the evaluation sets, NQ and HotpotQA are in-domain, while the remaining five benchmarks are used purely for out-of-domain generalization. All questions are answered against the same 2018 Wikipedia corpus (Karpukhin et al., 2020), regardless of the benchmark's original supporting corpus, which follows the unified evaluation protocol of Search-R1 (Jin et al., 2025a).

Table 4: Statistics of the seven QA benchmarks. "Hop" indicates whether answering typically requires single-hop or multi-hop evidence. NQ and HotpotQA additionally provide the training data.

Dataset	**Hop**	**Eval. split**	**# Train**	**# Eval.**
NQ	Single-hop	Test	79,168	3,610
TriviaQA	Single-hop	Test	–	11,313
PopQA	Single-hop	Test	–	14,267
HotpotQA	Multi-hop	Dev	90,447	7,405
2Wiki	Multi-hop	Dev	–	12,576
MuSiQue	Multi-hop	Dev	–	2,417
Bamboogle	Multi-hop	Test	–	125
Total	–	–	169,615	51,713

A.2 Training Details

Table 5 lists the main training hyperparameters. The GRPO configuration is kept identical to AutoRefine (Shi et al., 2025) to isolate the effect of credit assignment: we train for 300 steps with a global batch size of 256, a group size of 5, a constant learning rate of 1×10^{-6} without warmup, and BFloat16 precision under fully sharded data parallelism. Rollouts are generated with vLLM at temperature 1.0 and top-p 0.95, with at most five search calls per rollout, a maximum response length of 2048 tokens, and up to 1024 tokens of retrieved content per call. The four SR-Search-specific hyperparameters control the credit channels only: the channel weights α_S and α_R scale the calibrated SIG and RIG, the asymmetric bounds (c_-, c_+) clip the standardized gains, and λ sets the fraction of refinement credit routed back to the same-cycle search. Their default values are selected on the validation split via the one-factor sweeps in §5.3. SIG and RIG are computed from the log-probability of the ground-truth answer under the old policy $\pi_{\theta_{\text{old}}}$, which is held fixed during gain computation; all resulting values are treated as constants during normalization and credit routing, so no separately trained critic, process-reward model, or auxiliary scorer is introduced.

B More Experimental Results Analysis

B.1 F1 and EM Dynamics

Figure 7 tracks the validation F1 and EM of SR-Search throughout training. Both metrics rise steeply in the first 100 steps (EM from 0.127 to about 0.44) and then climb slowly before stabilizing around steps 240–300 (F1 $\approx$ 0.52–0.54, EM $\approx$ 0.44–0.45), indicating that the bulk of the RL benefit is realized in the early and middle phases of training. The EM curve remains consistently about 0.09 below F1, and the gap stays nearly constant over training: it originates from answers that contain the gold tokens together with redundant content, which passes the token-overlap test but fails exact matching. This echoes the discussion in §5.1 that refinement preserves evidence while occasionally introducing formatting redundancy into the final answer. Throughout training, the fraction of rollouts that terminate with a well-formed answer stays between 0.85 and 0.92, so the observed gains stem

Table 5: Main training hyperparameters of SR-Search.

	Hyperparameter	Value
GRPO	Total training steps	300
	Global batch size	256
	Micro batch size	64
	Learning rate	1×10^{-6}
	Rollout group size G	5
	Sampling temperature / top-p	1.0 / 0.95
	KL coefficient β	0.001
	Clip ratio ε	0.2
	Max search calls / retrieved docs per call	5 / 3
	Max response / observation length	2048 / 1024
SR-Search	SIG weight α_S	0.05
	RIG weight α_R	0.1
	Clipping bounds (c_-, c_+)	(1.0, 2.0)
	Delayed-credit coefficient λ	0.1

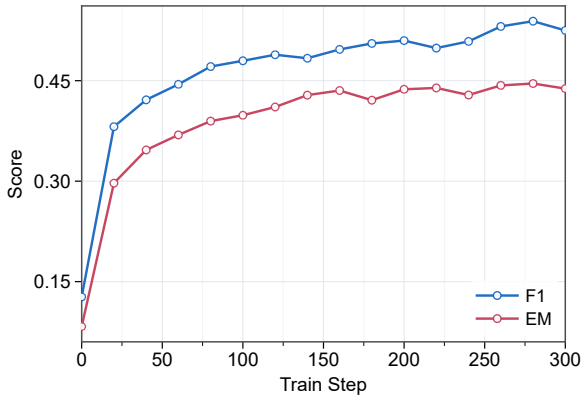


Figure 7: Validation F1 and EM of SR-Search over training.

from genuinely better search, refinement, and reasoning rather than from an improved completion rate.

B.2 Delayed Credit Analysis

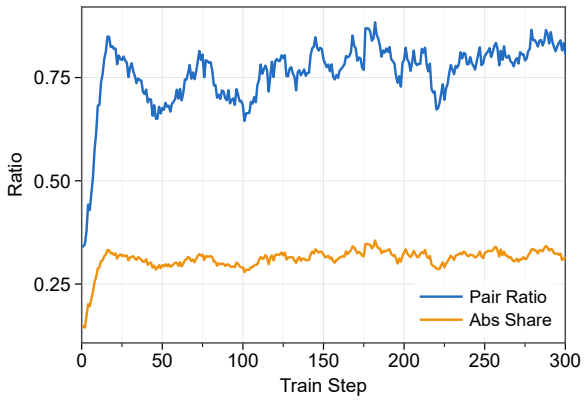


Figure 8: Dynamics of the delayed credit channel. Pair Ratio is the fraction of search actions that receive same-cycle RIG feedback; Abs Share is the fraction $|A^{R\to S}|/(|A^S| + |A^{R\to S}|)$ of the total local search credit contributed by the delayed channel.

Figure 8 examines how extensively the short-horizon delayed advantage of Eq. (8) is exercised during training. The pair ratio, i.e., the fraction of search actions that are paired with a valid same-cycle refinement and thus receive delayed credit, rises from about 0.55 in early training to a stable 0.78–0.81, showing that the delayed channel is not a sparse corner-case mechanism but a routinely activated credit path covering roughly four fifths of all searches. Meanwhile, the absolute share of the delayed credit within the total local search credit remains steady at 0.30–0.35: the hindsight feedback is sizable enough to matter, yet stays below half so that the immediate SIG remains the primary supervision for search, with the delayed channel acting as a complement rather than a replacement. Both curves evolve smoothly without abrupt jumps, indicating that the credit routing remains stable across the entire run, consistent with the design that delayed credit never crosses cycle boundaries and therefore preserves clean attribution.

B.3 Search Frequency

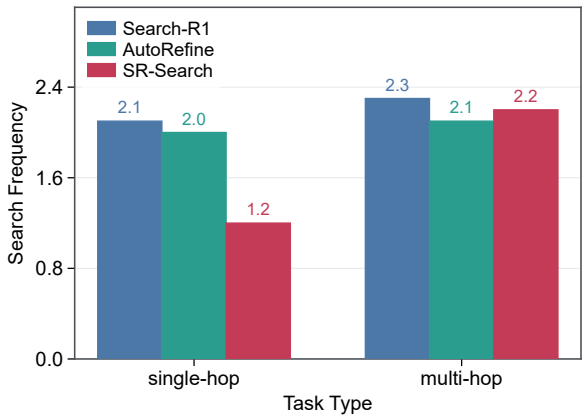


Figure 9: Average number of search calls per question on single-hop and multi-hop benchmarks.

Figure 9 compares the average number of search calls per question. On single-hop benchmarks, SR-Search issues only 1.2 searches, roughly 40% fewer than Search-R1 (2.1) and AutoRefine (2.0), while simultaneously achieving the highest single-hop average EM (0.541 vs. 0.512 and 0.472 in Table 1). The action-level credit thus teaches the policy to search less but more accurately, suppressing redundant retrievals that a trajectory-level reward cannot distinguish. On multi-hop benchmarks, SR-Search performs 2.2 searches, comparable to AutoRefine (2.1) and slightly below Search-R1 (2.3), yet converts the similar search budget into a markedly higher multi-hop average EM (0.367 vs. 0.325 and 0.191): the additional searches are spent on completing evidence chains rather than idling. The contrast between the two regimes (1.2 vs. 2.2 calls) further shows that SR-Search adaptively modulates its search depth according to task complexity, providing direct behavioral evidence for RQ3 that the credit channels shape, rather than merely reweight, the search policy.

C SR-Search Pseudocode

Algorithm 1 summarizes the complete training procedure of SR-Search. Each iteration proceeds in four stages. First, the old policy $\pi_{\theta_{\text{old}}}$ samples a group of rollouts per question by interacting with the search engine, and the trajectory-level outcome advantages are computed from the F1 rewards as in standard GRPO. Second, for every well-formed search–refine cycle, the same $\pi_{\theta_{\text{old}}}$ evaluates the ground-truth log-probability before the search, after the retrieved documents, and after the refinement, yielding the raw SIG and RIG. Third, the two gains are standardized and clipped separately within the rollout group, converted into the direct and delayed local advantages, and added onto the tokens of their responsible action spans. Finally, the policy is updated by the clipped surrogate objective with the composite token-level advantages. The group statistics in Eq. (6) require all valid gains of a question, so the scoring passes are batched over the whole group before normalization; the delayed channel reuses the already computed refinement credit and adds no scoring pass.

Algorithm 1 SR-Search Training

Require: training set $\mathcal{D}$, policy π_θ, reference policy π_{ref}, search engine, group size G, credit weights α_S, α_R, clipping bounds c_-, c_+, delayed coefficient λ

for each training iteration **do**
  $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$
  for each question (q, a) in the minibatch **do**
    Sample G rollouts $\{o_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(\cdot \mid q)$ by interacting with the search engine
    Compute outcome advantages $\{A_i^{\text{orm}}\}_{i=1}^{G}$ via Eq. (2)
    for each rollout o_i and each cycle k in o_i **do**
      if the search and document spans of cycle k are well-formed **then**
        Compute $\text{IG}_{i,k}^{S}$ with $\pi_{\theta_{\text{old}}}$ via Eq. (4) ▷ evidence acquisition
        if the refinement span of cycle k is complete **then**
          Compute $\text{IG}_{i,k}^{R}$ with $\pi_{\theta_{\text{old}}}$ via Eq. (5) ▷ evidence refinement
        end if
      end if
    end for
    Standardize and clip the collected SIG and RIG *separately* within the group via Eq. (6)
    Initialize $A_{i,t} \leftarrow A_i^{\text{orm}}$ for all tokens t of every rollout o_i
    for each cycle k with a valid gain **do**
      Compute $A_{i,k}^{S}$ and $A_{i,k}^{R}$ via Eq. (7)
      $A_{i,k}^{R\to S} \leftarrow \lambda\, A_{i,k}^{R}$ via Eq. (8) ▷ reuses the same-cycle RIG
      Add $A_{i,k}^{S} + A_{i,k}^{R\to S}$ to every token in $\mathcal{S}_{i,k}$, and $A_{i,k}^{R}$ to every token in $\mathcal{R}_{i,k}$
    end for
  end for
  Update π_θ by maximizing $\mathcal{J}_{\text{SR}}(\theta)$ in Eq. (10) with the composite advantages of Eq. (9)
end for

D Prompt Template

SR-Search uses the same interaction prompt as AutoRefine (Shi et al., 2025), which specifies the tag protocol for reasoning, searching, receiving documents, refining, and answering. The template is shown below; the question is substituted into the placeholder at the end.

Multi-Turn Search Interaction System Prompt

You are a helpful assistant excel at answering questions with multi-turn search engine calling. To answer questions, you must first reason through the available information using `<think>` and `</think>`. If you identify missing knowledge, you may issue a search request using `<search>` query `</search>` at any time. The retrieval system will provide you with the three most relevant documents enclosed in `<documents>` and `</documents>`. After each search, you need to summarize and refine the existing documents in `<refine>` and `</refine>`. You may send multiple search requests if needed. Once you have sufficient information, provide a concise final answer using `<answer>` and `</answer>`. For example, `<answer>` Donald Trump `</answer>`.

Question: `{question}`

E Case Study

To qualitatively illustrate how fine-grained credit assignment changes the behavior of the trained policy, we compare validation rollouts of SR-Search and AutoRefine on three identical questions drawn from the evaluation logs. The retrieved document blocks are truncated for brevity; all action spans are shown verbatim. The three cases highlight three recurring patterns: precise queries that retrieve decisive evidence (Case 1), stable action formatting (Case 2), and completing the evidence chain with fewer, more effective actions (Case 3).

Case 1: Query precision determines retrieval success. For the PopQA question below, SR-Search issues a bare keyword query whose top document is the film itself, and the refinement pins the screenwriter; AutoRefine's query retrieves an unrelated *Twin Peaks* page, and its refinement

then rationalizes the wrong evidence into a wrong answer. Under SIG, queries whose returned documents fail to raise the answer likelihood are explicitly penalized, steering the policy toward query formulations that retrieve decisive evidence.

Question: Who was the screenwriter for Two Weeks? **Gold Answer:** Steve Stockman

SR-Search (Ours) • **Correct**, 1 cycle

Action: <search> Who was the screenwriter for Two Weeks </search>

Observation: <documents> Doc 1 (Title: "Two Weeks (2006 film)") . . . </documents>

Refinement: <refine> From the documents, it's clear that the screenwriter for "Two Weeks" was Steve Stockman. . . . </refine>

Output: <answer> Steve Stockman </answer>

AutoRefine • **Wrong**, 1 cycle

Action: <search> Who was the screenwriter for Two Weeks? </search>

Observation: <documents> Doc 1 (Title: "Episode 7 (Twin Peaks)") . . . </documents>

Refinement: <refine> . . . we identified a movie called "Two Weeks" and we found that one of its writers was Mark Frost. </refine>

Output: <answer> Mark Frost </answer>

Case 2: Full-span credit stabilizes the action format. On a TriviaQA question, both methods retrieve the same Cole Porter document, but AutoRefine fails to close its document block and then degenerates into repeated invalid actions without ever producing an answer, whereas SR-Search completes a well-formed cycle and answers correctly. Because SR-Search routes credit over the complete action span including the opening and closing tags (§3.4), tag generation itself receives local supervision, which suppresses this failure mode.

Question: The song 'Wunderbar' comes from which Cole Porter musical? **Gold Answer:** Kiss Me, Kate

SR-Search (Ours) • **Correct**, 1 cycle

Action: <search> The song 'Wunderbar' comes from which Cole Porter musical </search>

Observation: <documents> Doc 1 (Title: "Cole Porter") . . . </documents>

Refinement: <refine> . . . the song 'Wunderbar' is from the musical "Kiss Me, Kate". . . . </refine>

Output: <answer> Kiss Me, Kate </answer>

AutoRefine • **Degenerate / No Answer**

Action: <search> From which musical does the song "Wunderbar" originate by Cole Porter? </search>

Observation: <documents> Doc 1 (Title: "Cole Porter") . . . [unclosed]

Feedback: *"My previous action is invalid. . . . Let me try again."* (×4, terminated without an answer)

Case 3: Fewer, more effective actions. For a 2Wiki comparison question, SR-Search resolves both entities with a single search whose refinement already establishes the two nationalities, answering correctly in one cycle. AutoRefine spends two cycles, and although its refinements correctly identify both persons as American, the final answer contradicts its own distilled evidence. The delayed advantage in SR-Search rewards searches whose evidence is successfully exploited downstream, encouraging compact evidence chains that the final reasoning can consume reliably.

Question: Do Khalil El-Amin and Tori Anthony share the same nationality? **Gold Answer:** yes

SR-Search (Ours) • **Correct**, 1 cycle

Action: <search> Do Khalil El-Amin and Tori Anthony share the same nationality </search>

Observation: <documents> Doc 1 (Title: "Khalil El-Amin") . . . American football offensive lineman . . . </documents>

Refinement: <refine> . . . Khalil El-Amin is American and Tori Anthony is American. . . . </refine>

Output: <answer> Yes </answer>

AutoRefine • **Wrong**, 2 cycles

Action 1: <search> Who is Khalil El-Amin and what is his nationality? Who is Tori Anthony . . . ? </search>

Observation 1: <documents> Doc 1 (Title: "May El-Khalil") . . . </documents>

Refinement 1: <refine> . . . Khalil El-Amin is an American football player. . . . </refine>

Action 2: <search> Who is Tori Anthony and what is her nationality? </search>

Observation 2: <documents> Doc 1 (Title: "Tori Anthony") . . . an American pole vaulter . . . </documents>

Refinement 2: <refine> . . . Tori Anthony is an American pole vaulter. </refine>

Output: <answer> No </answer>