PiCA: Pivot-Based Credit Assignment for Search Agentic Reinforcement Learning

Dongyi Liu[1*], **Yifan Niu**[1*], **Qinwen Wang**[1], **Han Xiao**[1], **Jia Li**[1,2†]

[1] The Hong Kong University of Science and Technology (Guangzhou)
[2] The Hong Kong University of Science and Technology

Abstract

Large Language Model (LLM)-based search agents trained with reinforcement learning (RL) have significantly improved the performance of knowledge-intensive tasks. However, existing methods encounter critical challenges in long-horizon credit assignment: (i) Reward Sparsity, where models receive only outcome feedback without step-level guidance to differentiate action quality; (ii) Isolated Credit, where credit is assigned to steps independently, failing to capture sequential dependencies; and (iii) Distributional Shift, where rewards are estimated on templates that deviate from the model's natural generative distribution. To address these issues, we propose **Pi**vot-Based **C**redit **A**ssignment (PiCA), a novel step reward mechanism that reformulates the search trajectory as a sequential process of cumulative search progress. Unlike prior isolated step rewards, PiCA defines process rewards as success probabilities dependent on the historical context based on Potential-Based Reward Shaping (PBRS). This approach identifies pivot steps, which comprise target golden sub-queries and sub-answers derived from historical trajectories, as information peaks that significantly boost the likelihood of a correct final answer. By anchoring these step rewards to the final task objective, PiCA provides dense, pivot-aware and trajectory-dependent guidance while maintaining distributional consistency. Extensive experiments show that PiCA outperforms existing strong baselines across seven knowledge-intensive QA benchmarks, achieving 15.2% and 4.4% improvements for 3B and 7B models. The consistent performance gains across various models show PiCA's robust generalization. The code is available at `https://github.com/novdream/PiCA`.

1 Introduction

Large Language Model (LLM)-based search agents [13, 43, 39] have recently redefined the paradigm for addressing long-horizon, knowledge-intensive tasks, such as multi-hop question answering [4, 31] and open-domain information seeking [54, 53]. For example, popular search agents, such as WebDancer [45], WebLeaper [36] and MiroThinker [37], can autonomously refine queries, summarize retrieved information from external environments through search tools (*e.g.*, online API, local corpus).

A primary bottleneck in these long-horizon tasks lies in incorrect credit assignment [20, 35, 52], specifically the misattribution of rewards to less important steps. Precise credit assignment is thus crucial for enabling search agents to decide when to search, how to formulate or refine queries, and how to incorporate retrieved evidence into multi-step reasoning.

Recently, Reinforcement Learning (RL) [34] has emerged as a promising paradigm for developing adaptive and autonomous search agents with verifiable rewards. However, existing methods face

[*] Equal contribution.
[†] Correspondence to: Jia Li (`jialee@ust.hk`).

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

three primary limitations: **(1) Sparse Rewards**. Early efforts [13, 33] primarily rely on outcome-only supervision (*i.e.*, answer accuracy). These approaches often lead to biased reward estimation, where high answer rewards are mistakenly given to wrong intermediate steps (*e.g.*, redundant query turns, wrong sub-answer turns). **(2) Isolated Credit Assignment**. While new methods [43, 47] attempt to incorporate fine-grained step rewards, they often suffer from isolated credit assignment where the step reward is estimated based solely on the local quality of the current turn. This neglects the inherent nature of knowledge-intensive tasks as a Markov Decision Process (MDP) [20]. **(3) Distributional Shift**. Recent methods [39, 47] concatenate ground-truth data with intermediate steps to estimate answer probability to give dense step rewards. However, this concatenation triggers a distributional shift since such sequences are absent from the model's natural generation, resulting in biased rewards.

To address these limitations, we propose **Pi**vot-Based **C**redit **A**ssignment (PiCA), which reformulates the search trajectory as a sequential process of cumulative search progress to reach the final correct goal. Unlike prior works that reward steps in isolation, we define the process reward as a success probability dependent on the historical trajectory based on potential-based reward shaping (PBRS) [44, 24]. In our formulation, PiCA follows a core intuition:

Main Intuition

The probability of reaching the correct answer in multi-hop tasks increases with the cumulative acquisition of **pivot steps** (*i.e.*, deriving target golden sub-queries and sub-answers based on historical trajectory)

Under this formulation, each pivot step represents a distinct peak in search progress, directly increasing the success probability of the final goal. Conversely, recognizing that non-pivot steps do not necessarily compromise the final goal, we explicitly link each process reward to the final outcome via conditional probability to better align with the final task objective. By augmenting PPO with our turn-level rewards, extensive experiments across seven in-domain and out-of-domain QA benchmarks confirm the efficacy of our method in retrieving relevant information and synthesizing evidence into correct, verifiable answers in knowledge-intensive tasks. Our main contributions are two folds: (1) We propose **PiCA**, a novel credit assignment framework where step rewards are dependent on the entire historical trajectory and can reflect which step has effective information. (2) PiCA outperforms competitive baselines by 15.2% (3B) and 4.4% (7B) average on seven multi-hop QA benchmarks.

2 Related Work

2.1 Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) enhances generative models by incorporating external knowledge, overcoming the limitations of static parameters [17]. Current research can be categorized into three stages based on the depth of integration between retrieval and reasoning. First, prompt-driven augmentation methods leverage prompt engineering to guide models through query decomposition and multi-turn retrieval. Representative works like REPLUG [30] align retrievers with black-box models, while FLARE [11] introduces active retrieval based on generation confidence. However, these methods are often susceptible to interference from redundant information in long contexts [21]. Second, fine-tuning and reflection-based architectures employ supervised fine-tuning (SFT) to empower models with self-critique capabilities. Self-RAG [3] utilizes reflection tokens for iterative optimization, CRAG [49] introduces corrective retrieval mechanisms, and RetroLLM [19] focuses on fine-grained evidence extraction to improve information utilization. Third, inference-time scaling and search-based methods, inspired by reasoning models like o1, focus on increasing computational investment during inference. RAG-star [10] and AirRAG [7] utilize Monte Carlo Tree Search (MCTS) for path exploration, while Search-o1 [18] models retrieval as an agentic behavior, significantly enhancing planning capabilities in complex tasks. Despite these advances, RAG still faces core bottlenecks: cascading errors where early missteps propagate through the reasoning chain, noise interference in long contexts affecting precise evidence extraction, and high computational costs or latency associated with search-augmented methods.

2.2 Reinforcement Learning for Agentic Search

Reinforcement learning (RL) has become a key component in post-training large language models for reasoning and decision-making. Outcome-supervised paradigms, such as RLHF and PPO-style methods (e.g., GRPO), have demonstrated strong performance on verifiable tasks including mathematics and code generation [25, 27, 29]. However, these approaches primarily rely on final outcome signals, which leads to a fundamental credit assignment challenge in long-horizon reasoning tasks: the model lacks explicit feedback on how intermediate reasoning steps contribute to the final result [2].To address this limitation, prior work has explored optimizing search and reasoning behaviors within an RL framework. For instance, Search-R1 formulates search as a reinforcement learning environment to learn query generation strategies [13], while StepSearch introduces step-wise rewards based on search progress to guide retrieval behavior [43]. TIPS further constructs dense rewards based on improvements in answer likelihood to stabilize long-horizon training [48]. In addition, Zerosearch simulates search environments and controls retrieval quality, enabling efficient training without relying on real-world search APIs [33].Despite these advances, existing methods still lack explicit modeling of step dependencies and the evolution process in multi-step reasoning, making it difficult to capture reasoning deviations and step-level contributions, thereby limiting fine-grained credit assignment in complex multi-hop scenarios.

3 Preliminary

In this section, we first establish the search agent task formulation in Section 3.1, followed by a detailed description of the search agentic RL pipeline in Section 3.2.

3.1 Task Formulation

Given a question q, a language model π_θ generates a trajectory $y = (\tau_1, \tau_2, \ldots, \tau_T)$, where T denotes the total number of interaction turns. Each turn τ_t is defined as a composite semantic block consisting of three functional components: a reasoning step <think>, a tool invocation <search>, and an environment observation <information>. The entire trajectory is assigned a binary label $l \in \{0, 1\}$, which serves as the ground-truth signal extracted from the last turn in <answer> tag.

We formulate the agentic search process as a MDP denoted by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma)$. At each turn t, $s_t = (q, \tau_{\leq t-1})$ contains the question q and the sequence of previously generated interaction turns $\tau_{\leq t-1}$. Based on this state, π_θ generates actions a_t following the format $a_t =$ <think><search>. Upon executing the think and search action a_t, the external search engine tools $\mathcal{T}$ returns an observation o_t with <information> tag. The interaction turn is then defined by their concatenation, $\tau_t = a_t \oplus o_t$. The state transition is deterministic, as the next state is uniquely determined by concatenating the previous sequence with the current output a_t. r_t denotes the reward provided by the environment or generated by the reward model for action a_t.

For PiCA training, we denote the training dataset as $\mathcal{D} = \{(q_i, y_i, l_i, \mathbf{z}_i)\}$, where q_i is the i-th question, y_i is the corresponding response consisting of a multi-step reasoning trajectory, l_i indicates whether the final answer is correct, and $\mathbf{z}_i$ is a sequence of binary labels where each $z_{i,j}$ signifies whether the j-th step in y_i is a pivot step. Formally, we define $\mathcal{D}_p = \{s_{i,j} \in y_i \mid z_{i,j} = \text{true}\}$ as the set of all pivot steps identified across all trajectories in the dataset $\mathcal{D}$, while the detailed construction prompts and criteria for these pivot labels are described in Appendix A.

3.2 Proximal Policy Optimization with Search Engine

Compared to standard PPO [28], at each turn t, the search agent's trajectory includes external environmental observation o_t. To ensure gradients are only propagated through the model's policy, we employ a token-level mask $I(y_t)$ to isolate model-generated actions a_t from the observations o_t.

$$\mathcal{J}_{\text{PPO}}(\theta) = \mathbb{E}_{q\sim\mathcal{D}, y\sim\pi_{\theta_{\text{old}}}(\cdot|q)}$$
$$\left\{ \frac{1}{\sum_{t=1}^{|y|} I(y_t)} \sum_{t=1:I(y_t)=1}^{|y|} \left[\frac{\pi_\theta(y_t|q, y_{<t})}{\pi_{\theta_{\text{old}}}(y_t|q, y_{<t})} A_t, \text{clip}\left(\frac{\pi_\theta(y_t|q, y_{<t})}{\pi_{\theta_{\text{old}}}(y_t|q, y_{<t})}, 1-\epsilon, 1+\epsilon \right) A_t \right] \right\}, \tag{1}$$

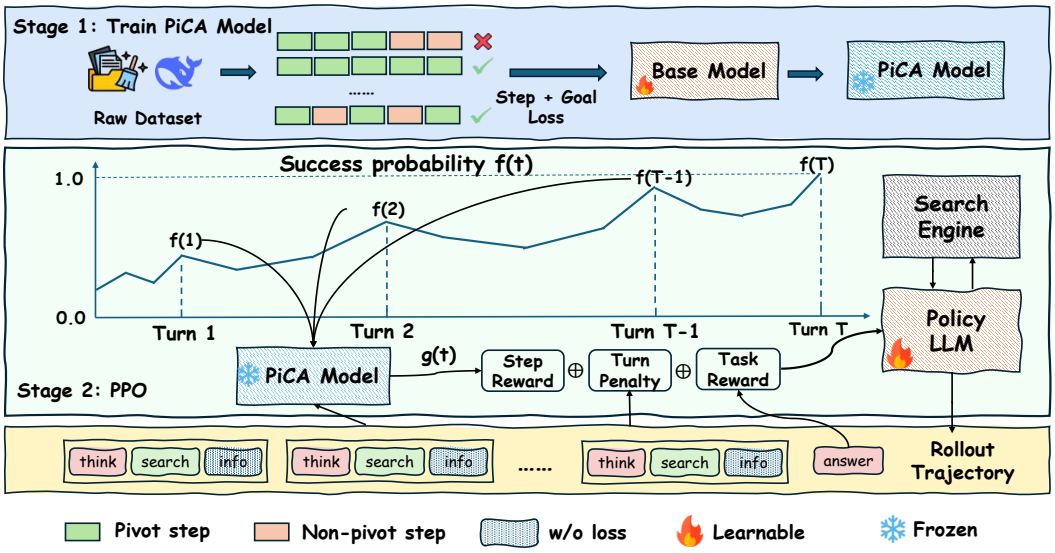


Figure 1: Overview of PiCA. Stage 1 is training a PiCA model on annotated pivot steps (Section 4.1). Stage 2 is policy optimization with PiCA (Section 4.2). The model generates trajectories with a frozen PiCA model assigning dense rewards based on relative success gain $g(t)$ derived from the success probability $f(t)$. The total reward integrates these step-wise gains, efficiency penalties, and final task outcomes, while retrieved content is masked during training.

where π_θ and π_{old} represent the current and previous policy models, respectively. $I(y_t) = 1$ if y_t is a LLM generated token (*i.e.*, <think>, <search>) and $I(y_t) = 0$ if y_t is a retrieved token (*i.e.*, <information>). The term ϵ is a clipping-related hyperparameter introduced in PPO to stabilize training. Following Generalized Advantage Estimation (GAE), A_t is estimated using V_ϕ and future rewards $\{r_{\geq t}\}$ derived from both the final answer and the reward model (Section 4.1).

4 Methodology

In this section, we detail the description of PiCA (Section 4.1). Building on this, we detail the design of the reinforcement learning algorithm with PiCA (Section 4.2).

4.1 Pivot-based Credit Assignment

To provide fine-grained supervision, we model the search trajectory as the evolution of the probability of reaching a correct answer based on historical states. In multi-step search, identifying an intermediate step as a fatal error is often ambiguous, since the model's self-correction mechanism [42, 15] allows it to recover from suboptimal states and eventually converge to the solution. However, the acquisition of pivot steps (*i.e.*, deriving target golden sub-queries and sub-answers based on historical trajectory) provides a clear signal of progress. To this end, our PiCA model generates step rewards based on the increment in success probability rather than the likelihood of failure. We define the success probability $f(t)$ as:

$$f(t) = P(l = 1 \mid s_t, a_t) \tag{2}$$

where $l = 1$ indicates the final answer is correct. The initial probability $f(0) = P(l = 1 \mid x)$ represents the prior difficulty of the problem given only the question. At the final step T, the state converges to a deterministic outcome: $f(T) = 1$ if the answer is correct, and $f(T) = 0$ otherwise.

Relative Success Gain. To measure the improvement in success likelihood elicited by the acquisition of new information, we calculate the relative success gain $g(t)$ based on the preceding steps in the trajectory. This metric quantifies the normalized change in success probability:

$$g(t) = \frac{f(t) - f(t-1)}{f(t-1)} = \frac{\Delta f(t)}{f(t-1)} \tag{3}$$

Here, $g(t) > 0$ implies the step is a productive advancement that raises the probability of success, while $g(t) < 0$ indicates the introduction of errors or logical confusion. Consequently, the success probability at any turn t can be decomposed as a product of these relative gains:

$$f(t) = f(0) \cdot \frac{f(1)}{f(0)} \cdot \frac{f(2)}{f(1)} \cdots \frac{f(t)}{f(t-1)} = f(0) \prod_{k=1}^{t} \left(1 + \frac{f(k) - f(k-1)}{f(k-1)}\right) \tag{4}$$
$$= f(0) \prod_{k=1}^{t} (1 + g(k))$$

Reward Shaping for Dense Signal. We employ Potential-Based Reward Shaping (PBRS) to map these probabilities into a dense reward signal. We define the potential function $\Phi(s_t)$ as the logarithm of the success probability:

$$\Phi(s_t) \equiv \log f(t) \tag{5}$$

Following the PBRS framework, the shaped reward r_t for the transition from s_{t-1} to s_t is given by:

$$r_t \equiv R(s_{t-1}, a_{t-1}, s_t) + \gamma \Phi(s_t) - \Phi(s_{t-1}) \tag{6}$$

By setting the intermediate environmental reward $R = 0$ and the discount factor $\gamma = 1$, the process reward simplifies to the log-ratio of relative success gain:

$$\begin{aligned} r_t &= \Phi(s_t) - \Phi(s_{t-1}) = \log f(t) - \log f(t-1) \\ &= \log\left(\frac{f(t)}{f(t-1)}\right) = \log\left(\frac{f(t-1) + \Delta f(t)}{f(t-1)}\right) = \log\left(1 + \frac{\Delta f(t)}{f(t-1)}\right) \\ &= \log(1 + g(t)) \end{aligned} \tag{7}$$

Reward Model Training. Based on the dataset $\mathcal{D}$ and pivot steps $\mathcal{D}_p$ and the success reward defined in Eq. (7), our training objective is designed to maximize the rewards of pivot steps while enabling the model to autonomously reward intermediate steps through final outcomes. The total loss consists of two components.

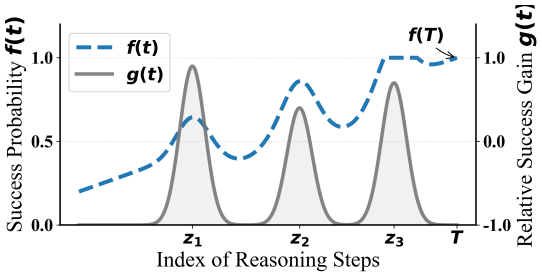


Figure 2: PiCA Step Reward

Step-level Explicit Supervision. For the pivot steps $t \in \mathcal{D}_P$ within each trajectory $\in \mathcal{D}$, we define the loss $\mathcal{L}_{\text{gold}}$ to ensure these actions provide positive search progress ($g_t > 0$). This explicitly encourages the model to recognize pivot steps as milestones that advance the state toward correctness:

$$\mathcal{L}_{\text{gold}} = -\sum_{L \in \mathcal{D}} \sum_{t \in \mathcal{D}_p} \log(g_t) \tag{8}$$

Outcome-level Implicit Supervision. For the complete trajectory y, we utilize the outcome label $l \in \{0, 1\}$ to supervise the final success probability $f(T)$. This objective allows the model to autonomously reward non-pivot steps based on whether the overall search process converges to a correct solution. The outcome-level loss is formulated as:

$$\mathcal{L}_{\text{final}} = \begin{cases} -\log f(T), & \text{if } l = 1 \\ -\log(1 - f(T)), & \text{if } l = 0 \end{cases} \tag{9}$$

where $f(T) \to 1$ for successful trajectories and $f(T) \to 0$ for failures. The overall loss to be minimized is as follows:

$$\mathcal{L} = \frac{1}{|\mathcal{D}|} \sum^{\mathcal{D}} (\mathcal{L}_{\text{final}} + \lambda_g \cdot \mathcal{L}_{\text{gold}}) \tag{10}$$

where λ_g is the weight coefficient to balance the two loss terms.

4.2 Policy Optimization

We optimize the search policy π_θ using Proximal Policy Optimization (PPO) in Eq. 1. The objective is to maximize the expected return by calculating the advantage $\hat{A}_t$, which depends on the value function V_ϕ and the combined reward signal. In our method, the total reward R for a search trajectory is decomposed into two components: the final outcome reward (r_{out}) and the step reward (r_{step}).

Outcome Reward (r_{out}). To ensure the model adheres to structural constraints and reaches the correct solution, the outcome reward is applied only at the final token of the trajectory, which consists of a format reward and a F1 answer reward.

PiCA (r_{step}). The step-wise reward is assigned to the last token of each search behavior (*i.e.*, <search>) round to provide dense supervision. However, a naive summation of rewards may trigger reward hacking [32, 41]. To mitigate this, we introduce a step penalty for trajectories:

$$r_{step,t} = \begin{cases} PiCA(s_t, a_t), & \text{if } t < 3 \\ PiCA(s_t, a_t) - \lambda \cdot \alpha^{(t-3)}, & \text{if } t \geq 3 \end{cases} \tag{11}$$

where $\alpha \in [1, 1.5]$ is the exponential growth factor and $\lambda \in [0, 0.5]$ is the base penalty coefficient. This penalty ensures the model is encouraged to perform necessary search steps.

Policy Optimization. For a trajectory $y = (\tau_1, \tau_2, \ldots, \tau_T)$, we define the turn reward R_t as **the last token** (*i.e.*, <search>, <answer>) in turn t. Intermediate turns receive $r_{step,t}$, while the final turn T incorporates outcome reward r_{out} to signal the overall success or failure of the trajectory to the model.

$$R_t = \begin{cases} r_{step,t}, & \text{if } t < T \\ r_{step,T} + r_{\text{out}}, & \text{if } t = T \end{cases} \tag{12}$$

The turn-level advantage is $A_t = R_t + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$, where $V_\phi(s_t)$ is a baseline value predicted by critic model. To incorporate such long-horizon dependencies, we compute a discounted cumulative advantage to propagate outcome signals backward to earlier turns.

$$\tilde{A}_t = \sum_{l=0}^{T-t} (\gamma\lambda)^l A_{t+l}. \tag{13}$$

where $\gamma \in (0, 1]$ is the discount factor, and $\lambda \in [0, 1]$ controls the propagation of future advantage signals. For any token j generated during turn t, its assigned token-level advantage is $\tilde{A}_{(j)} = \tilde{A}_t$. With the discounted advantages $\tilde{A}_{(j)}$ defined above, we optimize the agent policy following the same structure as PPO but with a finer-grained credit assignment. Formally, let $r_j(\theta) = \frac{\pi_\theta(y_j|q, y_{<j})}{\pi_{\theta_{\text{old}}}(y_j|q, y_{<j})}$, the final objective is

$$\mathcal{J}_{\text{PiCA}}(\theta) = \mathbb{E}_{q,y}\left[\frac{1}{\sum I(y_j)} \sum_{j=1}^{|y|} I(y_j) \cdot \min\left(r_j(\theta)\tilde{A}_{(j)}, \text{clip}(r_j(\theta), 1-\epsilon, 1+\epsilon)\tilde{A}_{(j)}\right)\right] \tag{14}$$

5 Experiments

5.1 Experimental Setup

Reward Model Training. We train Qwen2.5-3B-Instruct on new MuSiQue dataset, which contains approximately 60,000 trajectories, along with step-level process annotations. The reward model is trained via full-parameter fine-tuning. More training details are provided in Appendix F.

Search Agent Training. Following a multi-turn question answering setup, we conduct experiments with two model scales, Qwen-2.5-3B-Instruct and Qwen-2.5-7B-Instruct [50]. For retrieval, we adopt the E5 encoder [40] over the 2018 Wikipedia corpus, retrieving 3 documents at each interaction step. The models are trained on a combined dataset constructed from the NQ and HotpotQA training splits. We evaluate both in-domain and out-of-domain performance on seven QA benchmarks: NQ, TriviaQA, PopQA, 2WikiMultiHopQA, MuSiQue, HotpotQA, and Bamboogle [14, 16, 22, 9, 38, 51, 26]. Performance is measured using Exact Match (EM) and F1 scores [12].

Implementation Details. We adopt a unified set of hyperparameters across all methods. We use PPO with GAE ($\lambda = 1$, $\gamma = 1$), a KL penalty coefficient $\beta = 0.001$, and a clipping ratio $\epsilon = 0.2$.

Table 1: Exact Match (EM) results on seven QA benchmarks. In-domain tasks include NQ and HotpotQA; out-of-domain tasks include TriviaQA, PopQA, 2WikiMultiHopQA, MuSiQue, and Bamboogle.

Method	In-domain		Out-of-domain					Avg
	NQ	HotpotQA	TriviaQA	PopQA	2Wiki	MuSiQue	Bamboogle	
Qwen2.5-3B Instruct								
RAG	0.348	0.251	0.544	0.387	0.221	0.051	0.076	0.283
Search-o1	0.238	0.240	0.472	0.262	0.207	0.045	0.316	0.254
Search-R1	0.341	0.324	0.545	0.378	0.319	0.103	0.264	0.325
ZeroSearch	0.414	0.267	0.574	**0.448**	0.239	0.088	0.193	0.318
StepSearch	–	0.345	–	–	0.320	**0.174**	0.344	–
MT-PPO	0.397	0.255	0.562	0.405	0.214	0.062	0.080	0.282
TIPS	**0.435**	0.314	0.588	0.428	0.293	0.087	0.208	0.336
Ours	0.426	**0.400**	**0.612**	0.417	**0.408**	0.160	**0.347**	**0.396**
Qwen2.5-7B Instruct								
RAG	0.349	0.287	0.585	0.392	0.231	0.061	0.214	0.283
Search-o1	0.151	0.193	0.443	0.131	0.181	0.053	0.302	0.208
Search-R1	0.393	0.370	0.610	0.397	0.401	0.146	0.368	0.385
ZeroSearch	0.436	0.325	0.618	**0.515**	0.309	0.120	0.267	0.370
StepSearch	–	0.386	–	–	0.366	**0.226**	0.400	–
MT-PPO	0.424	0.265	0.551	0.416	0.228	0.069	0.112	0.295
TIPS	0.434	0.421	0.640	0.450	**0.430**	0.170	0.368	0.417
Ours	**0.460**	**0.424**	**0.641**	0.442	0.401	0.197	**0.419**	**0.426**

The batch size is set to 256, with a maximum context length of 4096 tokens and up to 4 retrieval turns per query. Training is conducted for 190 steps, or until performance collapses, using 8 $\times$ A800 GPUs with FSDP and gradient checkpointing. Additional details on training hyperparameters and the search engine server configuration are provided in Appendix F.

Baselines. We compare our method with a set of representative reinforcement learning approaches for search-augmented reasoning, including RAG, Search-o1, Search-R1, ZeroSearch, StepSearch, TIPS and MT-PPO. To ensure fair comparison, we follow prior work in adopting the same multi-turn question answering framework and retrieval configurations. All baselines are implemented following their original configurations, and we report the average performance for each model size.

5.2 Main Results

In Table 1 and 2, we compare with other competitive prompt-based and RL-based baselines on seven standard benchmarks (*i.e.*, in-domain, out-of-domain) to validate the effectiveness of our method. The EM and F1 scores are reported as the average of three independent runs. Based on these results, we can draw the following key observations:

(1) PiCA demonstrates excellent performance in knowledge-intensive tasks. As shown in Tables 1 and 2, Ours consistently outperforms competitive baselines on in-domain benchmarks. On Qwen2.5-3B, our method achieves the highest EM (0.400) and F1 (0.514) on HotpotQA, significantly surpassing models like StepSearch and Search-R1. It also maintains superior accuracy on NQ with a peak F1 score of 0.521. These results highlight the effectiveness of our approach in accurately retrieving and integrating knowledge to solve complex questions.

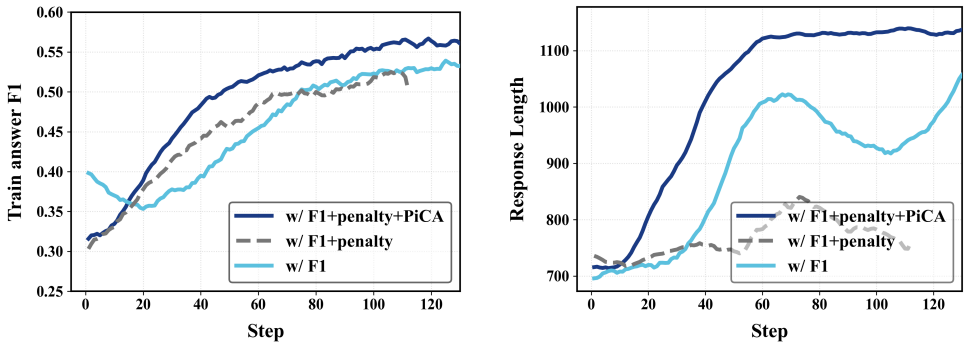
(2) PiCA shows strong generalization to out-of-domain scenarios. Across five out-of-domain benchmarks, our method achieves the highest average EM and F1 scores for both model scales. Notably, PiCA leads in challenging multi-hop tasks like MuSiQue and Bamboogle, outperforming strong RL-based methods such as TIPS. This consistent advantage across TriviaQA and 2Wiki underscores the robustness of PiCA when encountering diverse, unseen data distributions.

5.3 Ablation Study

To further investigate the contributions of PiCA, we conduct ablation experiments comparing three configurations: (w/ F1), which uses only rule-based outcome rewards; (w/ F1+penalty), which incor-

Table 2: F1 scores on seven QA benchmarks. In-domain tasks include NQ and HotpotQA; out-of-domain tasks include TriviaQA, PopQA, 2WikiMultiHopQA, MuSiQue, and Bamboogle.

Method	In-domain		Out-of-domain					Avg
	NQ	HotpotQA	TriviaQA	PopQA	2Wiki	MuSiQue	Bamboogle	
Qwen2.5-3B Instruct								
RAG	–	0.359	–	–	0.316	0.135	0.161	–
Search-o1	–	0.326	–	–	0.309	0.117	0.436	–
Search-R1	0.457	0.376	0.652	0.431	0.352	0.171	0.344	0.398
ZeroSearch	–	0.353	–	–	0.288	0.145	0.299	–
StepSearch	–	0.452	–	–	0.385	**0.261**	0.452	–
MT-PPO	0.478	0.342	0.629	0.448	0.261	0.111	0.140	0.344
TIPS	0.518	0.415	0.664	**0.474**	0.351	0.159	0.298	0.411
Ours	**0.521**	**0.514**	**0.691**	0.469	**0.472**	0.235	**0.463**	**0.481**
Qwen2.5-7B Instruct								
RAG	–	0.391	–	–	0.226	0.142	0.316	–
Search-o1	–	0.288	–	–	0.289	0.127	0.427	–
Search-R1	0.496	0.484	0.693	0.428	0.465	0.256	0.501	0.475
ZeroSearch	–	0.432	–	–	0.370	0.204	0.409	–
StepSearch	–	0.502	–	–	0.431	**0.312**	**0.530**	–
MT-PPO	0.506	0.353	0.621	0.465	0.286	0.121	0.209	0.366
TIPS	0.483	0.533	0.700	0.490	**0.466**	0.278	0.516	0.495
Ours	**0.553**	**0.542**	**0.713**	**0.493**	**0.466**	0.287	**0.530**	**0.512**



(a) Answer F1 score

(b) Response length

Figure 3: Comparison of PiCA with different rewards

porates a negative reward for each search step to encourage efficiency; and (w/ F1+penalty+PiCA), our full hybrid reward framework as shown in Figure 3 and Figure 4. We observe as follows:

(1) PPO with our step reward demonstrates the most prominent performance advantages. Our reward consistently outperforms standard PPO across all benchmarks, with evaluation performance closely mirroring training gains. Unlike standard PPO, our method maintains a steady upward trajectory, ensuring superior reasoning accuracy and long-term optimization stability.

(2) Step penalty rewards significantly accelerate convergence and our rewards can prevent training collapse. Before 40 steps, penalties rewards accelerate early convergence by discouraging inefficient, long reasoning paths that typically yield negative outcomes. However, relying solely on penalties eventually triggers length hacking, where the model curtails response length to minimize costs in Figure 3b. As shown in Figure 3a, this behavior leads to the performance collapse observed in the w/ F1+penalty model after step 100. In contrast, PiCA maintains stable response lengths by balancing penalties with learned rewards.

5.4 More Analysis

Generalization of PiCA. To evaluate the generalization of our method, we apply the same training setup to various base models across different families and scales. As shown in Table 3, our framework

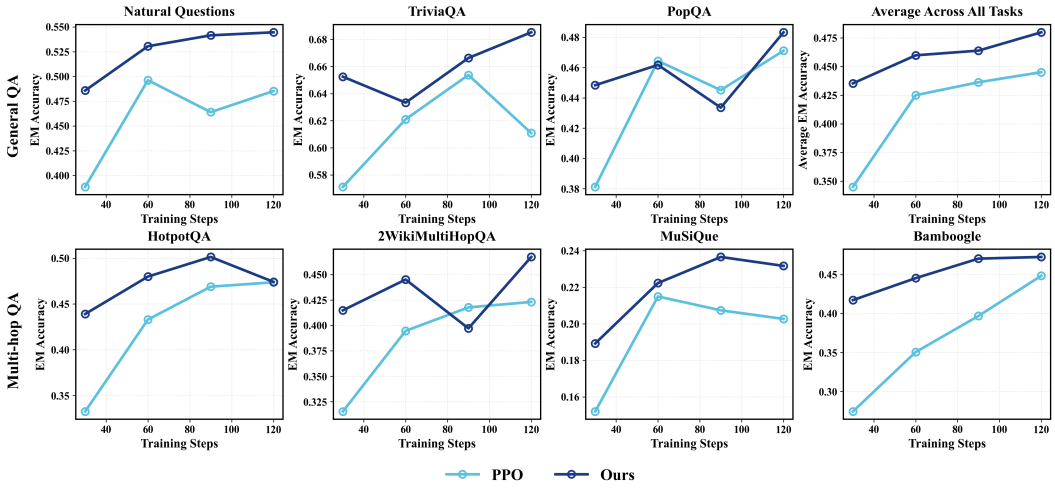


Figure 4: Evaluation results of PPO vs. PiCA

Table 3: Generalization of PiCA across model families and scales. EM/F1 are PiCA scores; percentages in parentheses indicate relative improvement over the outcome-only PPO baseline.

Model	EM	F1
Qwen2.5-3B-Instruct	39.6 (+15.1%)	48.1 (+13.7%)
Qwen3-4B-Instruct-2507	45.0 (+6.6%)	54.3 (+4.1%)
Qwen2.5-7B-Instruct	42.6 (+12.1%)	51.2 (+12.9%)
Llama3.1-8B-Instruct	40.2 (+34.0%)	48.5 (+29.1%)

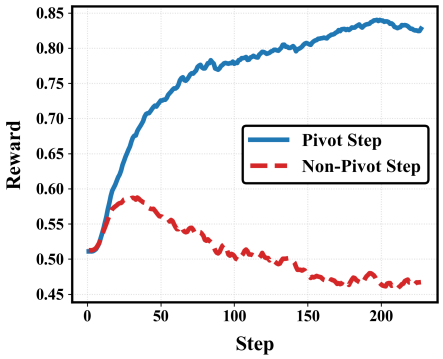


Figure 5: PiCA step rewards.

consistently enhances performance in all cases. While the 3B-scale model (Qwen2.5-3B) exhibits the most significant relative growth, the improvements remain consistent even on stronger base models. For instance, Qwen3-4B, which possesses higher initial capabilities, still achieves a +6.6% EM and +4.1% F1 boost. Furthermore, the method scales effectively to larger architectures like Qwen2.5-7B (+12.1%/12.9%) and Llama3.1-8B (+34.0%/29.1%), indicating that our framework robustly enhances search capability across different model families and parameter scales.

Step Reward Showcase. To evaluate the precision of PiCA, we conduct a fine-grained analysis of reward distribution during reward model training. As shown in Figure 5, PiCA effectively discriminates between **Pivot** and **Non-Pivot Steps**, with rewards diverging toward 0.8 and 0.45, respectively. Though normalized to $[0, 1]$ for visualization, these values correspond directly to the $[-1, 1]$ range used in our optimization framework. This provides a robust signal that prioritizes high-information-gain reasoning steps, as further detailed in the qualitative examples in Appendix G.

5.5 Generalize to web scenario and long-horizon scenario

Web Scenario. We further evaluate PiCA in a real-world web search scenario to assess its generalization ability, with results reported in Table 4. Following ARPO [6], we evaluate our methods on GAIA [23], Webwalker [46] and Humanity's Last Exam (HLE) [1] benchmarks. We report the Avg@3 metric which averages the F1 scores and LLM-as-a-Judge scores across 3 independent runs. During evaluation, each search retrieves up to 3 web pages, which are then processed by a summarization model instantiated with Qwen2.5-7B-Instruct. On average across GAIA, Webwalker, and HLE benchmarks, PiCA outperforms all baselines, demonstrating strong generalization to real-world web search tasks.

Table 4: Avg@3 performance on GAIA, Webwalker and HLE with real web search engine. The best results are indicated with **bold**.

Benchmark	Search-R1	Stepsearch	Zerosearch	TIPS	IGPO	PiCA
GAIA	18.45	12.23	16.50	19.22	<u>19.75</u>	**20.44**
Webwalker	**13.99**	10.50	13.11	12.15	13.64	<u>13.97</u>
Humanity's Last Exam (HLE)	6.17	4.66	4.51	5.64	**7.02**	<u>6.46</u>

Table 5: Exact Match results on seven QA benchmarks. In-domain tasks include NQ and HotpotQA; out-of-domain tasks include TriviaQA, PopQA, 2WikiMultiHopQA, MuSiQue, and Bamboogle.

Method	NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue	Bamboogle
PPO	0.455	0.663	0.437	0.418	0.402	0.199	0.432
PiCA	0.469	0.711	0.465	0.430	0.427	0.213	0.472

Long-horizon Scenario. To assess generalization in long-horizon search tasks, we additionally train Qwen2.5-7B-Instruct with 10 retrieval turns. Table 5 reports the results on NQ, TriviaQA, PopQA, HotpotQA, 2Wiki, MuSiQue, and Bamboogle. The results demonstrate that PiCA consistently improves performance over PPO and generalizes to longer-horizon search.

Pivot Step Distribution Analysis. For both QA and deepsearch task, we use Qwen2.5-7B-Instruct to rollout trajectories. Then, we use PiCA Qwen2.5-3B-Instruct reward model to score each turn. Here turn1 represents the proportion of pivots within turn1 data. The result in Table 6 shows that DeepResearch tasks face more complex environments and need multi-hop reasoning, so they need longer trajectories, while QA tasks are short-hop questions with a specific environment. Specifically, DeepSearch maintains a relatively high pivot ratio in later turns and peaks at turn 7 (0.8254), suggesting that long-horizon tasks require continued strategic re-planning beyond the early turns. In contrast, QA peaks at turn 3 (0.8317) and then drops at turn 4 (0.7364), consistent with its shorter trajectory and limited need for iterative reasoning.

Table 6: Pivot step distribution across turns for QA and DeepSearch tasks.

Turn	QA task	Deepsearch task
Turn Counts	4.72	7.18
turn0	0.7620	0.7413
turn1	0.7689	0.7796
turn2	0.8020	0.8175
turn3	0.8317	0.8237
turn4	0.7364	0.7792
turn5	–	0.7837
turn6	–	0.7931
turn7	–	0.8254
turn8	–	0.7675
turn9	–	0.7282

6 Conclusion

In this work, we present Pivot-Based Credit Assignment (PiCA), a novel credit assignment designed for search agentic reinforcement learning in knowledge-intensive tasks. By reformulating the search trajectory as a sequential process of cumulative search progress, PiCA effectively mitigates critical challenges in long-horizon credit assignment, such as reward sparsity and isolated credit. Our PiCA is theoretically grounded in Potential-Based Reward Shaping (PBRS) and identifies pivot steps as information peaks to provide dense, trajectory-dependent guidance without the risk of distributional shift. Extensive experiments over seven diverse multi-hop QA benchmarks demonstrate that PiCA achieves state-of-the-art performance, outperforming competitive baselines by up to 15.2% while ensuring the retrieval of verifiable and accurate evidence.

Acknowledgments and Disclosure of Funding

This research was supported by NSFC 62572418, the Guangdong Provincial Talent Program under Grant (No.2024TQ08X366). Additionally, this work is partially supported by the Research Travel Grant of the Base of Red Bird MPhil (RBM) at The Hong Kong University of Science and Technology (Guangzhou). We would like to thank our RBM Project Supervisor Qinhao He and Academic Supervisor Jia Li for the academic support.

Competing Interests

The authors declare no additional competing interests beyond the institutional affiliations and support disclosed in this paper.

References

[1] Long and Phan, Alice Gatti, Nathaniel Li, Adam Khoja, Ryan Kim, Richard Ren, Jason Hausenloy, et al. A benchmark of expert-level academic questions to assess ai capabilities. *Nature*, 649(8099):1139–1146, January 2026.

[2] Jose A Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. Rudder: Return decomposition for delayed rewards. *Advances in Neural Information Processing Systems*, 32, 2019.

[3] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*, 2023.

[4] M Chen, L Sun, T Li, H Sun, Y Zhou, C Zhu, H Wang, JZ Pan, W Zhang, H Chen, et al. Research: Learning to reason with search for llms via reinforcement learning. arxiv 2025. *arXiv preprint arXiv:2503.19470*, 2025.

[5] DeepSeek-AI, Aixin Liu, Bei Feng, et al. Deepseek-v3 technical report, 2025.

[6] 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. Agentic reinforced policy optimization, 2025.

[7] Wenfeng Feng, Chuzhan Hao, Yuewei Zhang, Jingyi Song, and Hao Wang. Airrag: Activating intrinsic reasoning for retrieval augmented generation via tree-based search. *arXiv e-prints*, pages arXiv–2501, 2025.

[8] Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. A survey on llm-as-a-judge, 2025.

[9] 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*, pages 6609–6625, 2020.

[10] Jinhao Jiang, Jiayi Chen, Junyi Li, Ruiyang Ren, Shijie Wang, Wayne Xin Zhao, Yang Song, and Tao Zhang. Rag-star: Enhancing deliberative reasoning with retrieval augmented verification and refinement. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7064–7074, 2025.

[11] Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 7969–7992, 2023.

[12] Bowen Jin, Jinsung Yoon, Priyanka Kargupta, Sercan O. Arik, and Jiawei Han. An empirical study on reinforcement learning for reasoning-search interleaved llm agents, 2025.

[13] Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

[14] 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 (Volume 1: Long Papers)*, pages 1601–1611, 2017.

[15] Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal Behbahani, and Aleksandra Faust. Training language models to self-correct via reinforcement learning, 2024.

[16] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

[17] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

[18] 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. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5420–5438, 2025.

[19] Xiaoxi Li, Jiajie Jin, Yujia Zhou, Yongkang Wu, Zhonghua Li, Ye Qi, and Zhicheng Dou. Retrollm: Empowering large language models to retrieve fine-grained evidence within generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16754–16779, 2025.

[20] Minhua Lin, Zongyu Wu, Zhichao Xu, Hui Liu, Xianfeng Tang, Qi He, Charu Aggarwal, Hui Liu, Xiang Zhang, and Suhang Wang. A comprehensive survey on reinforcement learning-based agentic search: Foundations, roles, optimizations, evaluations, and applications, 2025.

[21] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the association for computational linguistics*, 12:157–173, 2024.

[22] 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 (volume 1: Long papers)*, pages 9802–9822, 2023.

[23] Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants, 2023.

[24] Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning (ICML)*, pages 278–287. Morgan Kaufmann, 1999.

[25] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[26] 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*, pages 5687–5711, 2023.

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

[28] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

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

[30] Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. Replug: Retrieval-augmented black-box language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8371–8384, 2024.

[31] 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, 2025.

[32] Joar Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward hacking, 2025.

[33] Hao Sun, Zile Qiao, Jiayan Guo, Xuanbo Fan, Yingyan Hou, Yong Jiang, Pengjun Xie, Yan Zhang, Fei Huang, and Jingren Zhou. Zerosearch: Incentivize the search capability of llms without searching. *arXiv preprint arXiv:2505.04588*, 2025.

[34] R.S. Sutton and A.G. Barto. Reinforcement learning: An introduction. *IEEE Transactions on Neural Networks*, 9(5):1054–1054, 1998.

[35] Hui-Ze Tan, Xiao-Wen Yang, Hao Chen, Jie-Jing Shao, Yi Wen, Yuteng Shen, Weihong Luo, Xiku Du, Lan-Zhe Guo, and Yu-Feng Li. Hindsight credit assignment for long-horizon llm agents, 2026.

[36] Zhengwei Tao, Haiyang Shen, Baixuan Li, Wenbiao Yin, Jialong Wu, Kuan Li, Zhongwang Zhang, Huifeng Yin, Rui Ye, Liwen Zhang, et al. Webleaper: Empowering efficiency and efficacy in webagent via enabling info-rich seeking. *arXiv preprint arXiv:2510.24697*, 2025.

[37] MiroMind Team, Song Bai, Lidong Bing, Carson Chen, Guanzheng Chen, Yuntao Chen, Zhe Chen, Ziyi Chen, Jifeng Dai, Xuan Dong, et al. Mirothinker: Pushing the performance boundaries of open-source research agents via model, context, and interactive scaling. *arXiv preprint arXiv:2511.11793*, 2025.

[38] 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.

[39] 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, 2026.

[40] 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.

[41] Xiaohua Wang, Muzhao Tian, Yuqi Zeng, Zisu Huang, Jiakang Yuan, Bowen Chen, Jingwen Xu, Mingbo Zhou, Wenhao Liu, Muling Wu, Zhengkang Guo, Qi Qian, Yifei Wang, Feiran Zhang, Ruicheng Yin, Shihan Dou, Changze Lv, Tao Chen, Kaitao Song, Xu Tan, Tao Gui, Xiaoqing Zheng, and Xuanjing Huang. Reward hacking in the era of large models: Mechanisms, emergent misalignment, challenges, 2026.

[42] Yifei Wang, Yuyang Wu, Zeming Wei, Stefanie Jegelka, and Yisen Wang. A theoretical understanding of self-correction through in-context alignment, 2024.

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

[44] E. Wiewiora. Potential-based shaping and q-value initialization are equivalent. *Journal of Artificial Intelligence Research*, 19:205–208, 2003.

[45] Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhengwei Tao, Dingchu Zhang, Zekun Xi, Gang Fu, Yong Jiang, et al. Webdancer: Towards autonomous information seeking agency. *arXiv preprint arXiv:2505.22648*, 2025.

[46] Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Linhai Zhang, Yulan He, Deyu Zhou, Pengjun Xie, and Fei Huang. Webwalker: Benchmarking llms in web traversal, 2025.

[47] Yutao Xie, Nathaniel Thomas, Nicklas Hansen, Yang Fu, Li Erran Li, and Xiaolong Wang. Tips: Turn-level information-potential reward shaping for search-augmented llms, 2026.

[48] Yutao Xie, Nathaniel Thomas, Nicklas Hansen, Yang Fu, Li Erran Li, and Xiaolong Wang. Tips: Turn-level information-potential reward shaping for search-augmented llms. *arXiv preprint arXiv:2603.22293*, 2026.

[49] Shi-Qi Yan et al. Corrective retrieval augmented generation. *arXiv preprint arXiv:2401.15884*, 2024.

[50] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025.

[51] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William 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*, pages 2369–2380, 2018.

[52] Chenchen Zhang. From reasoning to agentic: Credit assignment in reinforcement learning for large language models, 2026.

[53] Yubao Zhao, Weiquan Huang, Sudong Wang, Ruochen Zhao, Chen Chen, Yao Shu, and Chengwei Qin. Training multi-turn search agent via contrastive dynamic branch sampling, 2026.

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

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes].

   Justification: The contribution and scope have been reflected in the introduction and abstract, especially in the last part of the introduction.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: The limitation is discussed in the last part of the submission.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: Our work does not include theoretical result.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have disclose all the information for reproduction in the, method, experiment and appendix.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The dataset are open available, and the code will be released after acceptance.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The training and test details have been provided in the experiments and the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We have provided statistical significance tests in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We have provided the compute resources in the experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.

- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: This work match with the NeurIPS Code of Ethics in every respect.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We have discuss the broader impact in the appendix.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This paper poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We have properly credited the original owners when using existing assets.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: This paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [Yes]

    Justification: We have used the ChatGPT-4o for "Editing (*e.g.*, grammar, spelling, word choice)" as stated in OpenReview.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.

A Data Generation

Process. As described in introduction about **pivot steps**, we enrich approximately 12,000 training instances from the StepSearch [43] dataset. For each question, we interact with real search environment and utilize the DeepSeek-V3 API [5] to sample 5 times. Each trajectory is then annotated using an LLM-as-a-judge approach [8], following the structured evaluation prompts detailed in Appendix A. Through a sequential matching process, an intermediate step is labeled as a positive pivot step (+) if it successfully gives a target golden sub-answer with its corresponding query, while all other steps are assigned a negative label (−). Formally, we define $\mathcal{D}_p \subseteq \mathcal{D}$ as the set of all pivot steps. In addition to step-level signals, we assign binary outcome labels $l \in \{0, 1\}$ by performing an Exact Match (EM) between the final generated answer and the ground truth. To ensure the high fidelity of the training data, we implement a rigorous filtering pipeline that discards trajectories with structural inconsistencies, such as mismatches between the number of generated search rounds and their corresponding step labels. Furthermore, we conduct human-in-the-loop audits on a random subset of the data to verify that the identified pivot steps represent genuine logical junctures in the reasoning process.

Data Quality. To assess the quality of the generated data, we randomly sample 100 test trajectories and have 5 human annotators audit the pivot labels. Since fully annotating the entire dataset is infeasible, we reuse the 100 annotated trajectories and find that approximately 5% of the labels in the original training set are noisy; despite this, PiCA still achieves performance improvement. To further test robustness, we conduct an additional noise injection experiment by randomly flipping 10% of the DeepSeek-V3 annotated labels. We then examine distributional consistency by using the 100-step policy model to generate rollouts on the same 100 test queries, checking whether the reward model remains consistent with human annotations as the policy evolves. All metrics (*i.e.*, Human Judge Accuracy, Precision, and Recall) are computed against the majority-vote labels from the five annotators. The results in Table 7 demonstrate that PiCA remains robust under 10% label noise and stays aligned with human judgments throughout policy evolution.

Table 7: Evaluation of data quality under default, 10% label noise, and 100-step training settings, measured by human judge accuracy, precision, and recall.

Setting	Human Judge Accuracy	Precision	Recall
Default	0.9160	0.8357	0.8719
10% Label Noise	0.8530	0.8013	0.8526
100-step Training	0.9050	0.8270	0.8651

Prompt. The prompts below are used to generate step label for our reward model training. The labeling process evaluates each step in a trajectory against the golden sub-queries and sub-answers from the MuSiQue dataset in StepSearch. A step is labeled '+' if its search and reasoning successfully yield the corresponding golden sub-answer; otherwise, it receives a '-' label.

Trajectory Evaluation Prompt

As a Search-QA auditor, your task is to evaluate a multi-hop reasoning trajectory. You must compare each step's <search> and <think> process against the provided Golden Reference.
[Standard Reference]
- Ground Truth (GT): {gt}
- Golden Min Hop: {hop}
- Golden Sub-Queries: {golden_queries}
- Golden Sub-Answers: {golden_answers}
[Trajectory for Evaluation] {clean_trajectory}
[Strict Evaluation Rules]
Assign '+' or '-' based on these logic checks:
1. Step Quality (+):
- The <search> query effectively matches the intent of the 'Golden Sub-Queries' for that step.
- If the <information> block is missing the 'Golden Sub-Answers', but the model correctly recognizes this in <think> and searches again, it is still '+'.

2. Step Failure (-):
- Summary Omission: The <information> block contains the 'Golden Sub-Answers', but the model's <think> block ignores it or fails to summarize it, leading to a redundant or wrong next step.
- Logic Deviation: The <search> query significantly departs from the 'Golden Sub-Queries' logic or targets the wrong entity.
- Hallucination: The <think> block claims to have found information that is not in the <information> block.
- Fact Mismatch: The step produced an answer that contradicts or deviates from the Golden Sub-Answers/GT (e.g., searching for a location and picking the wrong entity from the results).
- False Positive Search: The query looks correct, but the model extracted the WRONG information from the search results.
- Final Answer Conflict: The final answer is gt but the model produced ans. This implies at least one step in the trajectory MUST have a logical flaw or information extraction error. Identify it and mark it '-'.
[Output Requirement]
Return ONLY a JSON object with:
"reason": A brief explanation of where the logic failed.
"labels": An array of '+' or '-' corresponding to each '</search>' tag in order.
Example: "reason": "Step 2 found the info but the model didn't summarize it.", "labels": ["+", "-", ...]
Note: The length of the "labels" array MUST exactly match the number of '</search>' tags.

Data Example

"question": "When did the alma mater of William C. Perry start issuing engineering degrees?",
"trajectory": [{
"step": 1, "observation": None, "think": <think>I need to parse the question: "When did the alma mater of William C. Perry start issuing engineering degrees?" First, I need to identify who William C. Perry is. Likely William C. Perry is a notable person, possibly a former US Secretary of Defense (William Perry, served 1994-1997). His full name is William James Perry, but sometimes referred to as William C. Perry? Actually, William Perry's middle name is James, not C. Maybe it's a different William C. Perry. Could be William C. Perry, maybe an academic or engineer. Let's search for William C. Perry to confirm. </think> "action": <search>William C. Perry alma mater</search>, "is_pivot": true, },
{ "step": 2, "observation": <information>Doc 1 (Title: William C. Perry)... Doc 2 (Title: Perry Belmont)... Doc 3 (Title: William Perry (American football))... </information>, "think": <think>The search results show a William C. Perry who graduated from the University of Kansas in 1922. That's likely the person in question. So his alma mater is the University of Kansas. The question asks: "When did the alma mater of William C. Perry start issuing engineering degrees?" So we need to find when the University of Kansas started issuing engineering degrees. We need to search for the history of engineering degrees at the University of Kansas. </think> "action": <search>University of Kansas start issuing engineering degrees</search>, "is_pivot": true, },
{ "step": 3, "observation": <information>Doc 1 (Title: University of Kansas)... Doc 2 (Title: University of Kansas)... Doc 3 (Title: University of Kansas)... </information>, "think": None
"action": <answer>1873</answer>, "is_pivot": true,
}],
"final_answer": "1873"

B Prompt for Research Plan on Question Answering

Following Search-R1, the prompts below are used to generate search trajectories during policy optimization.

Trajectory Generation Prompt

Answer the given question. You must conduct reasoning inside <think> and </think> first every time you get new information. After reasoning, if you find you lack some knowledge, you can call a search engine by <search> query </search> and it will return the top searched results between <information> and </information>. You can search as many times as your want. If you find no further external knowledge needed, you can directly provide the answer inside <answer> and </answer>, without detailed illustrations. For example, <answer> Beijing </answer>. Question: {question}

C Details of Benchmarks and Metrics

C.1 General Question Answering

Natural Questions (NQ). Natural Questions (NQ) [16] is a large-scale QA benchmark based on real Google search queries, where annotators label long answers (paragraph-level) and, when possible, short spans or yes/no answers from Wikipedia pages. The dataset contains approximately 307K training, 8K development, and 8K test examples, with annotations reflecting natural user information needs.We follow the standard evaluation protocol and report EM and F1 scores.

TriviaQA. TriviaQA [14] is a large-scale open-domain QA dataset constructed from trivia and quiz sources, featuring diverse question formulations and multiple evidence documents per query. It contains over 95K question–answer pairs and more than 650K question–answer–evidence triples, with evidence drawn from both web pages and Wikipedia. This setup challenges models in both retrieval and reasoning. We follow prior work and report EM and F1.

PopQA. PopQA [22] is an entity-centric QA benchmark derived from Wikidata triples, designed to evaluate performance across both popular and long-tail knowledge. It contains approximately 14K examples, each constructed from subject–relation–object triples and enriched with metadata such as entity IDs, relation types, and Wikipedia page-view statistics. This setup enables analysis of retrieval bias and factual memorization. We report EM and F1 following standard evaluation.

C.2 Multi-Hop Question Answering

HotpotQA. HotpotQA [51] is a multi-hop QA dataset requiring reasoning across multiple Wikipedia documents, with sentence-level supporting fact annotations to encourage explainable reasoning. It contains approximately 113K examples and supports both distractor and fullwiki settings. We evaluate performance using EM and F1.

2WikiMultiHopQA. 2WikiMultiHopQA [9] is a large-scale multi-hop QA benchmark combining Wikipedia text with structured knowledge from Wikidata. It includes around 192K examples and provides both supporting facts and explicit reasoning paths in the form of triples. We follow the standard evaluation protocol and report EM and F1.

MuSiQue. MuSiQue [38] constructs multi-hop questions by composing independent single-hop questions, enforcing compositional reasoning and reducing shortcut learning. The dataset contains about 25K questions spanning 2–4 hops and provides intermediate reasoning steps. We report EM and F1 following prior work.

Bamboogle. Bamboogle [26] is a small but challenging dataset of 125 manually curated two-hop questions designed to minimize shortcut reasoning. Each question requires combining multiple facts from Wikipedia, while supporting evidence is not explicitly provided. We evaluate performance using EM and F1.

C.3 Exact Match (EM)

The Exact Match (EM) metric evaluates whether the predicted answer exactly matches any of the reference answers. Formally, EM is defined as a binary indicator:

$$\mathrm{EM}(\hat{y}, \mathcal{Y}) = \begin{cases} 1, & \text{if } \exists\, y \in \mathcal{Y} \text{ such that } \hat{y} = y, \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

Here, $\hat{y}$ denotes the predicted answer, and $\mathcal{Y}$ represents the set of all acceptable ground-truth answers.

C.4 F1 Score

The F1 score measures the token-level overlap between the predicted answer and a reference answer. Given a predicted token set $T_{\hat{y}}$ and a ground-truth token set T_y, the F1 score is computed as:

$$\mathrm{F1}(T_{\hat{y}}, T_y) = \frac{2 \cdot |T_{\hat{y}} \cap T_y|}{|T_{\hat{y}}| + |T_y|}. \tag{16}$$

When multiple reference answers are available, we compute the F1 score against each candidate and report the maximum value:

$$\mathrm{F1}(\hat{y}, \mathcal{Y}) = \max_{y \in \mathcal{Y}} \mathrm{F1}(T_{\hat{y}}, T_y). \tag{17}$$

D Limitation and Future Direction

While our method demonstrates significant improvements across various QA tasks, due to limited computational resources, we are unable to train larger models (e.g., 14B and 32B series). Although the observed trends are consistent, future work could extend this validation to ultra-large-scale models or a broader diversity of architectures to ensure the approach's generalizability across varying model capacities.

Moreover, our current framework predominantly relies on process reward models that necessitate external supervision or high-quality teacher signals. In follow-up studies, we plan to investigate more autonomous reward mechanisms, such as intrinsic motivation, self-reflective feedback, or information-gain-based rewards. Such explorations would reduce the dependency on labor-intensive annotations and pave the way for a more self-evolving and robust agent.

E Broader Impacts

By introducing a unified process reward framework, our work establishes a new paradigm for enhancing the transparency and reliability of LLM-based search agents. The transition from outcome-only supervision to granular, context-aware credit assignment not only mitigates the "black-box" nature of long-horizon reasoning but also ensures that agent behaviors are intrinsically aligned with verifiable information gain.

Furthermore, our approach provides a scalable solution for developing intelligent AI in knowledge-intensive domains. By effectively neutralizing the distribution inconsistency and credit assignment gaps, this research serves as a catalyst for the deployment of autonomous agents in high-stakes environments (*e.g.*, scientific research, legal analysis) where step-by-step accountability is paramount. Ultimately, by fostering more knowledgeable and self-correcting agents, our findings contribute to the broader goal of building verifiable, and human-aligned artificial intelligence systems that can navigate increasingly complex information landscapes.

F Implementation Details

F.1 Hyper-parameters

In Qwen backbone, we use <|vision_start|> token as special token to give turn-level rewards.

Table 8: Hyperparameter settings for RL training.

Parameter	**Value**
data.train_batch_size	256
data.val_batch_size	128
data.max_prompt_length	8192
data.max_response_length	800
data.max_start_length	2048
data.max_obs_length	800
data.shuffle_train_dataloader	True
algorithm.adv_estimator	gae
algorithm.gamma	1.0
algorithm.kl_ctrl.kl_coef	0.001
algorithm.no_think_rl	False
actor_rollout_ref.actor.optim.lr	$7e{-7}$
actor_rollout_ref.actor.optim.lr_warmup_steps_ratio	0.285
actor_rollout_ref.actor.ppo_mini_batch_size	32
actor_rollout_ref.actor.ppo_micro_batch_size	8
actor_rollout_ref.actor.state_masking	True
actor_rollout_ref.model.enable_gradient_checkpointing	True
actor_rollout_ref.model.use_remove_padding	True
actor_rollout_ref.actor.fsdp_config.param_offload	True
actor_rollout_ref.actor.fsdp_config.grad_offload	True
actor_rollout_ref.actor.fsdp_config.optimizer_offload	True
actor_rollout_ref.rollout.name	vllm
actor_rollout_ref.rollout.temperature	1.0
actor_rollout_ref.rollout.n_agent	5
actor_rollout_ref.rollout.tensor_model_parallel_size	1
actor_rollout_ref.rollout.gpu_memory_utilization	0.6
actor_rollout_ref.rollout.log_prob_micro_batch_size	16
actor_rollout_ref.ref.log_prob_micro_batch_size	16
actor_rollout_ref.ref.fsdp_config.param_offload	True
critic.optim.lr	$7e{-6}$
critic.optim.lr_warmup_steps_ratio	0.015
critic.ppo_micro_batch_size	16
critic.model.enable_gradient_checkpointing	True
critic.model.use_remove_padding	True
critic.model.fsdp_config.param_offload	True
critic.model.fsdp_config.grad_offload	True
critic.model.fsdp_config.optimizer_offload	True
reward_model.url	localhost:5000/get_reward
step_reward_scale	0.3
baseline_step_reward	0.55
outcome_reward_scale	1.5
trainer.critic_warmup	0
trainer.n_gpus_per_node	8
trainer.nnodes	1
trainer.total_epochs	15
trainer.total_training_steps	190
trainer.save_freq	60
trainer.test_freq	30
max_turns	5
retriever.topk	3
retriever.url	localhost:8000/retrieve

Table 9: Hyperparameter settings for Reward Model.

Parameter	Value
model.pretrain	3B model
data.split (train/test)	train / test
data.input_key	input
data.label_key	value
attn.implementation	flash_attention_2
training.packing	True
training.placeholder_token	<|vision_start|>
reward.tokens	{+, -}
crm.gamma	1.0
sequence.max_length	32768
batch.train_batch_size	256
batch.micro_batch_size	16
optimization.learning_rate	5×10^{-6}
optimization.max_epochs	1
optimization.max_samples	100K
precision	bf16
parallel.zero_stage	3
training.gradient_checkpointing	True
training.num_gpus	2
logging.save_steps	500
logging.eval_steps	100
logging.logging_steps	1

G Case Study



Figure 6: this case study presents a failed trajectory with an incorrect final answer. Although the overall result is wrong, some intermediate steps still align with the golden reasoning process. PiCA assigns high positive rewards to these informative steps, while giving lower or negative rewards to redundant or misleading steps that lead to the incorrect answer.



Figure 7: this case study presents a successful reasoning trajectory in which the intermediate reasoning and retrieval steps consistently align with the golden process, leading to a correct final answer. In such cases, PiCA assigns high positive rewards to each informative step, reflecting the consistency between the reasoning trajectory and the target solution path.