

CoPRM: A Collaborative Process Reward Model for Guiding Search Agent Reasoning

Xingquan Zhang^{1,2}, Yang Zhao^{1,2,*}, Chong Li^{1,2},
Donglei Yu^{1,2}, Pu Jian^{1,2}, Chengqing Zong^{1,2,3,*}

¹School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China

²State Key Laboratory of Multimodal Artificial Intelligence Systems,
Institute of Automation, CAS, Beijing, China

³School of Information Science and Technology, ShanghaiTech University, Shanghai, China
{zhangxingquan2022, zhaoyang2015, lichong2021}@ia.ac.cn
{yudonglei2021, jianpu2023}@ia.ac.cn, cqzong@nlpr.ia.ac.cn

Abstract

Large Language Model (LLM)-based search agents have demonstrated superior performance in knowledge-intensive tasks through interleaved reasoning and search interactions. Process Reward Models (PRMs) can enhance search agent performance by guiding step selection during inference. However, existing PRMs for search agents face a critical bottleneck: assessment unreliability in their training data. This issue stems from an over-reliance on single-source supervision, forcing a compromise between global solvability and local validity. We propose **CoPRM** (Collaborative Process Reward Model), a process reward model trained on high-quality preference data generated through our Collaborative Calibration methodology. This approach synthesizes immediate quality assessments with outcome-informed evaluations through a dual-strategy mechanism: strictly enforcing *consensus filtering* to eliminate unreliable signals, and employing an *outcome-informed gating* mechanism to resolve heterogeneous cross-type trade-offs. Trained on 10k preference pairs, CoPRM provides plug-and-play test-time guidance, enabling diverse search agents to achieve consistent performance improvements across four multi-hop QA benchmarks, demonstrating substantial gains, robust cross-domain generalization, and superior efficiency.¹

1 Introduction

To address the limitations of Large Language Models (LLMs) (Chowdhery et al., 2023; Zhao et al., 2026) in multi-step reasoning tasks requiring access to evolving knowledge or out-of-domain information (Ji et al., 2023), integrating LLMs with

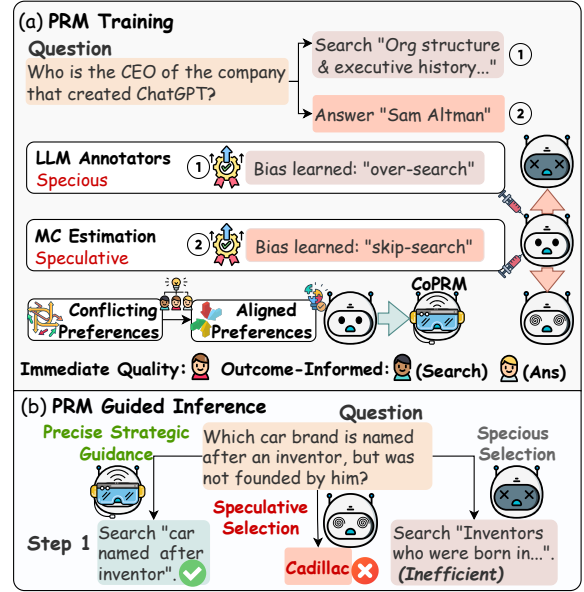


Figure 1: Illustration of CoPRM. Existing PRMs for search agents suffer from assessment unreliability due to single-source supervision, while CoPRM synthesizes immediate quality with outcome-informed signals to ensure precise strategic guidance.

search engines has emerged as a promising solution (Jin et al., 2025a; Zheng et al., 2025b; Xiang et al., 2025). This integration enables LLMs to iteratively acquire necessary knowledge during reasoning, giving rise to *LLM-based search agents* that dynamically interact with external knowledge sources (Gao et al., 2025b; Singh et al., 2026). Although extensive research on training methodologies (Asai et al., 2024; Song et al., 2025; Chen et al., 2025b; Jin et al., 2025b) has significantly enhanced the capabilities of these agents, they still struggle to distinguish the optimal step among generated candidates during inference, leading to suboptimal reasoning paths.

Process Reward Models (PRMs) offer a com-

*Equal corresponding authors.

¹<https://github.com/krzz2q723-tech/CoPRM>

promising avenue to address this challenge by providing test-time guidance to steer models toward accurate problem-solving trajectories (Zhang et al., 2025b). Unlike mathematical PRMs that verify internal logical deductions based on binary correctness (Zheng et al., 2025a), PRMs for search agents must evaluate the strategic utility of complex external interactions (Gao et al., 2025b). This requires distinguishing between logically valid steps based on information gain and making nuanced cross-type decisions (e.g., determining when to search versus answer).

However, developing effective PRMs for search agents is constrained by assessment unreliability stemming from single-source supervision. As illustrated in Figure 1(a), current approaches typically depend on either Monte Carlo (MC) estimation or LLM-based annotators. Methods relying on MC estimation (Sun et al., 2025c; Feng et al., 2025) use ground-truth outcomes to ensure global solvability but fail to penalize logical gaps by indiscriminately rewarding speculative steps. Conversely, LLM-based annotators (Xiong et al., 2026; Jiang et al., 2025) offer dense evaluations of local validity. Crucially, without objective execution feedback, they are prone to hallucination, frequently endorsing plausible but futile steps due to inherent plausibility bias. These unreliable signals prevent PRMs from learning accurate value functions. As illustrated in Figure 1(b), this results in misguidance during inference: outcome bias leads to hallucinated shortcuts, while plausibility bias drives the agent into inefficient search loops. Beyond data limitations, a structural barrier persists: the lack of a unified step definition renders existing methods incompatible with the reasoning loop of outcome-based Reinforcement Learning (RL) agents (Lin et al., 2025), precluding "plug-and-play" guidance and leaving their test-time scaling largely unexplored.

To bridge these gaps, we propose **CoPRM** (**Collaborative Process Reward Model**), a PRM designed to provide plug-and-play guidance for diverse search agents. We introduce a Collaborative Calibration methodology that generates high-quality preference data through a dual-strategy framework. To eliminate unreliable signals in same-type (search-search or answer-answer) comparisons, we strictly enforce *consensus filtering*, retaining only pairs where immediate quality and outcome utility agree. To resolve heterogeneous cross-type (search-answer) trade-offs, we employ an *outcome-informed gating* mechanism

using objective ground-truth evidence to determine whether information sufficiency warrants termination. Trained on this calibrated data, CoPRM delivers precise strategic guidance to navigate search agents toward optimal paths.

Extensive experiments demonstrate that CoPRM successfully translates this high-quality data into superior inference guidance. CoPRM provides effective plug-and-play guidance, achieving consistent performance gains across diverse search agents without agent-specific retraining. Moreover, CoPRM exhibits superior efficiency, achieving robust performance with 10k training samples and reducing average task completion iterations by 12.9%.

2 Related Work

LLM-Based Search Agents. While Large Language Models demonstrate strong reasoning capabilities (Gemini Team, 2025; Zhang et al., 2026), they still encounter knowledge bottlenecks that lead to hallucination during complex reasoning processes (Li et al., 2024b; Sun et al., 2025b). To address this limitation, researchers have developed LLM-based search agents that can invoke retrievers as tools, enabling dynamic interaction with external knowledge through iterative search and reasoning (Gao et al., 2025b; Singh et al., 2026). These search agents can be categorized into three main paradigms based on their training methodologies: prompt engineering approaches (Yao et al., 2023; Trivedi et al., 2023; Li et al., 2025), Supervised Fine-Tuning (SFT) on high-quality trajectories (Asai et al., 2024; Schick et al., 2023), and Reinforcement Learning (Kaelbling et al., 1996) approaches (Song et al., 2025; Chen et al., 2025b; Jin et al., 2025b). However, these optimization approaches primarily focus on the training phase and lack attention to inference-time scaling.

PRMs for LLM-Based Search Agents. Recent advancements have adapted PRMs to knowledge-intensive tasks. A prominent line of research necessitates model-specific Monte Carlo Tree Search (MCTS) sampling to collect extensive PRM training data, distinguished by their step-wise annotation strategies: employing either MC estimation (Sun et al., 2025c; Feng et al., 2025) or LLM-based labeling (Jiang et al., 2025). Conversely, RAG-Gym (Xiong et al., 2026) adopts a framework eschewing MCTS integration, where a critic trained on LLM-annotated steps guides actor reasoning. However, these approaches face limitations: re-

liance solely on LLM annotations or MC estimation introduces assessment unreliability, and model-specific effectiveness prevents generalization. A detailed discussion of related work on PRMs and test-time scaling is provided in Appendix L.

3 Preliminary

To facilitate our study, we present a unified formalization that captures the essential mechanisms of search agents (Jin et al., 2025a). We define the state, action, process units, and the reasoning process as follows:

State space $\mathcal{S}$: A state $s_t \in \mathcal{S}$ at time step t consists of:

$$s_t = (Q, H_t), \quad (1)$$

where Q is the initial query and H_t is the history of thoughts, searches, and retrieved results prior to time step t .

Action space $\mathcal{A}$: At each time step t , the agent can perform one of two types of actions $a_t \in \mathcal{A}$:

$$a_t = \begin{cases} \langle \text{search} \rangle q_t \langle / \text{search} \rangle \\ \langle \text{answer} \rangle \text{ans}_t \langle / \text{answer} \rangle \end{cases}. \quad (2)$$

The *search* action generates a search query q_t to retrieve relevant external information. The *answer* action produces a final response ans_t .

Process unit (i.e., step) $\mathcal{P}$: Each process unit p_t combines a reasoning step $T_t = \langle \text{think} \rangle \text{Think}(s_t) \langle / \text{think} \rangle$ with an action:

$$p_t = (T_t, a_t). \quad (3)$$

We distinguish between two types of process units based on the action type: *search steps* where a_t is a search action, and *answer steps* where a_t is an answer action.

The agent operates by iteratively executing process units. At each time step t , the agent may generate k candidate steps to form the candidate set $P_t = \{p_t^1, \dots, p_t^k\}$, where each p_t^i is either a search step or an answer step. We denote the search step subset as $P_t^S \subseteq P_t$ and the answer step subset as $P_t^A \subseteq P_t$. The agent selects one candidate to execute. If a search step is selected, the process continues to the next iteration with updated state; if an answer step is chosen, the process terminates.

4 Method

This section develops our CoPRM framework for enhancing search agent reasoning through process-level guidance. As shown in Figure 2, the framework consists of two main components: (a) a Collaborative Calibration methodology for generating high-quality preference data by synthesizing immediate quality signals with outcome-informed signals, followed by PRM training on this data, and (b) a guided inference process that leverages the trained CoPRM to select the optimal step during search agent reasoning.

4.1 CoPRM Training Data Collection

Our CoPRM requires training data in the form of preference pairs (s_t, p_t^+, p_t^-) , where s_t represents the current state and p_t^+, p_t^- are the preferred and less preferred candidate steps. To obtain preference data for effective training, we propose a collaborative reward mechanism that synthesizes immediate quality signals with outcome-informed signals through Collaborative Calibration. As shown in Figure 2(a), our data generation methodology implements this mechanism through three carefully designed stages (detailed in Appendix H):

Stage 1: State Generation and Immediate Quality Assessment. To collect a diverse set of states s_t and their corresponding candidate steps P_t for our training data, we employ a ranker-guided iterative framework that simultaneously provides immediate quality assessments. At each time step, the search agent generates a set of candidate steps P_t . A ranker then evaluates each candidate $p_t^i \in P_t$ based on the quality of its analytical reasoning and the appropriateness of the proposed action within the current context. The highest-ranked candidate is selected to advance the state, thereby ensuring the exploration of high-quality trajectories. This ranking order naturally defines our immediate preference function, $G(p_t^i, p_t^j)$ (denoted as G_{ij}). We establish a unified convention for all preference functions employed in this work—Immediate (G), Outcome-Informed (O), and Final Collaborative ($Pref$): each returns +1 if p_t^i is preferred, -1 if p_t^j is preferred, and 0 if they are equivalent.

Stage 2: Outcome-informed Evaluation. While Stage 1 provides immediate quality signals, our collaborative reward mechanism additionally requires outcome-informed signals to capture the long-term effectiveness of candidate steps. Since search steps

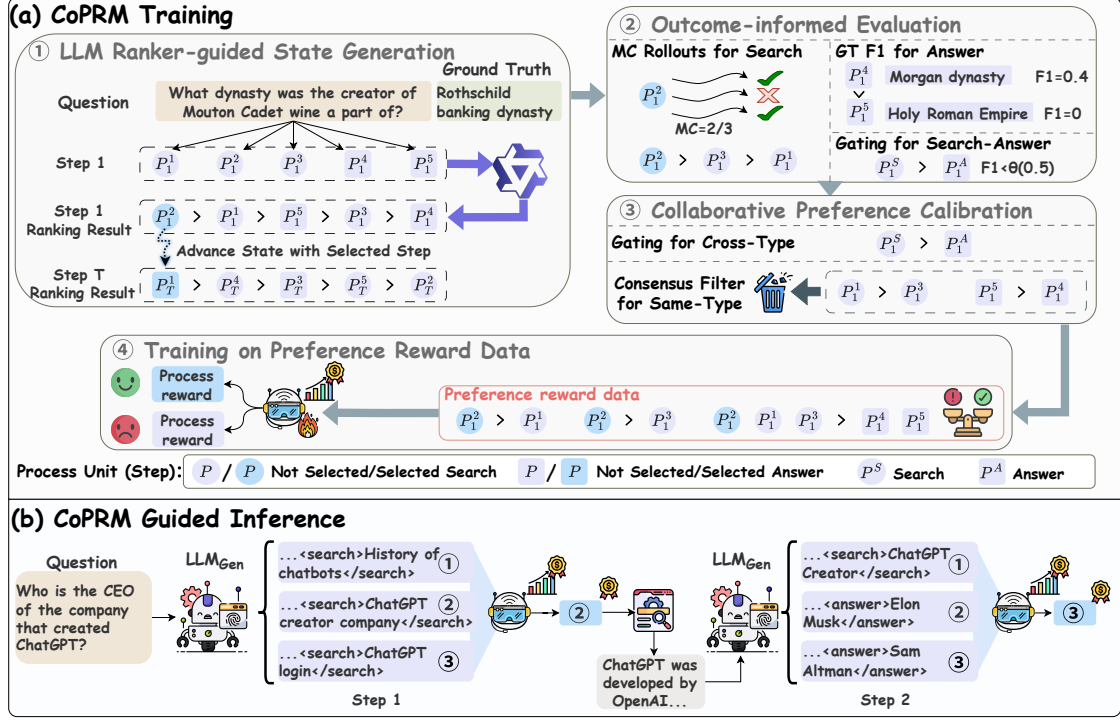


Figure 2: Overview of the CoPRM Framework: (a) a Collaborative Calibration methodology generates high-quality preference data by synthesizing immediate quality signals with outcome-informed signals for PRM training, and (b) a guided inference process leverages the trained CoPRM to select the optimal step during search agent reasoning.

and answer steps have fundamentally different characteristics in terms of outcome evaluation, we compute outcome-informed signals differently based on step types.

Ground Truth (GT) F1 Scoring for Answer Steps. For answer steps, we can directly evaluate their outcome-informed signals through ground truth comparison. Let $F1(p_t^a, ans^*)$ represent the F1 score between the generated answer in answer step p_t^a and the ground truth answer ans^* , which provides an objective measure of the answer step’s contribution to final success.

Monte Carlo Rollouts for Search Steps. For search steps, direct comparison with ground truth answers is not feasible since they do not produce final answers. Therefore, we estimate their expected outcome value through rollout-based evaluation. For each search step p_t^s , we perform N rollouts where the agent generates subsequent steps until a final answer is produced. Each rollout j yields a final answer ans_j . The rollout-based value estimate $V(s_t, p_t^s)$ is defined as the MC estimated success rate over all rollouts, where a rollout is considered successful if its final answer’s F1 score exceeds a

threshold τ :

$$V(s_t, p_t^s) = \frac{1}{N} \sum_{j=1}^N \mathbf{1}_{[F1(ans_j, ans^*) > \tau]}, \quad (4)$$

where ans^* is the ground truth answer and $\mathbf{1}_{[\cdot]}$ is the indicator function (1 if condition holds, 0 otherwise).

Similarly, let $O(p_t^i, p_t^j)$ (denoted as O_{ij}) represent the outcome-informed preference. For notational brevity, let $C_{ij}^S := (p_t^i \in P_t^S) \wedge (p_t^j \in P_t^S)$ and $C_{ij}^A := (p_t^i \in P_t^A) \wedge (p_t^j \in P_t^A)$ denote same-type conditions. For same-type pairs:

$$O_{ij}^{\text{same}} = \begin{cases} \text{sign}(V(s_t, p_t^i) - V(s_t, p_t^j)) & C_{ij}^S \\ \text{sign}(F1(p_t^i, ans^*) - F1(p_t^j, ans^*)) & C_{ij}^A \end{cases}, \quad (5)$$

where $\text{sign}(\cdot)$ is the sign function that returns $\{-1, 0, +1\}$.

For cross-type pairs where $p_t^i \in P_t^S$ and $p_t^j \in P_t^A$, we evaluate whether the answer step’s quality warrants termination:

$$O_{ij}^{\text{mixed}} = \begin{cases} 1 & \text{if } F1(p_t^j, ans^*) < \theta \\ -1 & \text{otherwise} \end{cases}. \quad (6)$$

Stage 3: Preference Construction via Collaborative Calibration. Having established immediate

quality signals and outcome-informed signals, we now integrate these signals to generate the final preference dataset. At each state s_t , the candidate set P_t may contain both search and answer candidates. Our Collaborative Calibration applies different strategies based on step types: for same-type pairs, we require consensus between both signals; for cross-type pairs (search vs. answer), we rely solely on outcome-informed evaluation via a gating mechanism (Eq. 6), as neither immediate quality assessment nor direct outcome score comparison is reliable across heterogeneous step types.

The final preference $Pref_{ij}$ for any distinct pair $p_t^i, p_t^j \in P_t$ is computed based on step types. When $Pref_{ij} = 0$, the pair is discarded.

$$Pref_{ij} = \begin{cases} +1 & G_{ij} > 0 \wedge O_{ij}^{\text{same}} > 0, C_{ij}^S \text{ or } C_{ij}^A \\ -1 & G_{ij} < 0 \wedge O_{ij}^{\text{same}} < 0, C_{ij}^S \text{ or } C_{ij}^A \\ O_{ij}^{\text{mixed}} & p_t^i \in P_t^S, p_t^j \in P_t^A \\ 0 & \text{otherwise} \end{cases}. \quad (7)$$

4.2 CoPRM Training

We train CoPRM r_ϕ by fine-tuning pre-trained language models using the preference dataset D where each preference pair (s_t, p_t^+, p_t^-) is rigorously calibrated: enforcing consensus for same-type steps and resolving cross-type trade-offs via outcome-informed gating. We optimize the parameters ϕ using a pairwise ranking loss (Ouyang et al., 2022):

$$L_{PRM}(\phi) = -\mathbb{E}_{(s_t, p_t^+, p_t^-) \sim D} \left[\log \sigma(r_\phi(s_t, p_t^+) - r_\phi(s_t, p_t^-)) \right]. \quad (8)$$

This loss optimizes r_ϕ to prioritize steps that balance local reasoning validity with global contribution to the final success, ensuring precise assessment of strategic value.

4.3 CoPRM-Guided Search Agent Inference

Our CoPRM, denoted as r_ϕ and trained via the methodology in Section 4.2, guides the agent’s candidate step selection at each iteration (see detailed algorithm in Appendix J). As depicted in Figure 2(b), the guided inference process generates a trajectory $(p_1^*, p_2^*, \dots, p_T^*)$, where T denotes the termination step ($T \leq T_{\max}$). At each step t , the process works as follows:

- (1) For the current state s_t , the agent generates candidate steps to form P_t .

- (2) CoPRM scores each candidate step and selects the optimal one for execution:

$$p_t^* = \arg \max_{p \in P_t} r_\phi(s_t, p). \quad (9)$$

- (3) If p_t^* is an answer step, set $T = t$ and terminate with final answer.
- (4) Otherwise, update state: $s_{t+1} = s_t \oplus (p_t^*, \text{Retrieved}(p_t^*))$, and continue to step $t + 1$.

5 Experiments

5.1 Experimental Setup

Datasets. Our preference dataset is constructed exclusively from the HotpotQA training set (Yang et al., 2018). For evaluation, we assess in-domain performance on the HotpotQA dataset. To evaluate out-of-domain generalization, we utilize three multi-hop question answering (QA) datasets: 2WikiMultihopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023). Following R1-Searcher (Song et al., 2025), we randomly sample 500 examples from the validation sets of HotpotQA, 2WikiMultihopQA, and MuSiQue, while using the complete test set for Bamboogle. We further evaluate CoPRM on BrowseComp-Plus (Chen et al., 2026), a fixed-corpus benchmark for multi-step web research.

Evaluation Metrics. We evaluate model performance using two categories of metrics: lexical matching and semantic evaluation. For lexical matching, we use Exact Match (EM) and F1 score. For semantic evaluation, following Chen et al. (2025b), we employ Qwen2.5-72B-Instruct (Yang et al., 2024) as a judge (LJ) (Li et al., 2024a) to evaluate the semantic correctness of final answers against ground truth. For BrowseComp-Plus, we report answer accuracy and evidence recall, with further evaluation details provided in Appendix I.

Baselines. Our evaluation includes three types of baselines to comprehensively assess CoPRM’s effectiveness:

Foundational Baselines: We compare against: Naive Generation, Naive RAG, Iter-RetGen (Shao et al., 2023), IRCoT (Trivedi et al., 2023), and Search-o1 (Li et al., 2025).

Search Agent Baselines: We evaluate CoPRM’s guidance effectiveness across search agents implemented through different training paradigms: (1)

Method	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			Average		
	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
Qwen2.5-7B-Instruct (Foundation Baselines)															
Naive Generation	18.60	26.12	31.20	25.20	29.63	27.80	3.20	11.25	12.40	10.40	19.02	22.40	14.35	21.50	23.45
Naive RAG	33.20	44.07	51.00	23.20	30.61	29.60	6.60	13.62	17.00	20.00	29.23	36.00	20.75	29.38	33.40
Iter-RetGen	35.60	46.03	53.80	24.80	<u>30.82</u>	30.20	9.40	<u>17.25</u>	18.80	21.60	29.30	36.00	22.85	30.85	34.70
IRCoT	30.60	42.11	52.00	18.00	26.36	29.80	7.00	14.45	20.00	25.60	<u>36.80</u>	45.60	20.30	29.93	36.85
Search-o1	37.20	48.48	57.00	36.00	43.22	43.60	16.60	26.20	30.00	34.40	46.37	52.80	31.05	41.07	45.85
PE-ReSearch^{PE}															
+Greedy	24.00	37.32	59.40	23.20	34.35	46.00	13.80	22.68	29.60	21.60	35.35	48.80	20.65	32.43	45.95
+SC	37.40	46.99	57.40	38.60	<u>46.39</u>	49.00	17.40	<u>26.60</u>	<u>30.20</u>	36.80	47.54	50.40	<u>32.55</u>	<u>41.88</u>	<u>46.75</u>
+GymPRM	27.40	38.48	56.40	26.60	35.99	40.60	15.80	24.31	28.00	33.60	45.93	52.80	25.85	36.18	44.45
+RePRM	28.20	37.45	52.40	30.40	37.79	40.40	11.40	20.62	25.20	28.00	40.79	48.00	24.50	34.16	41.50
+CoPRM	36.60	47.82	60.00	39.60	47.98	49.20	21.80	31.37	35.00	44.80	56.02	60.80	35.70	45.80	51.25
ReSearch^{RL}															
+Greedy	45.40	56.77	66.80	48.80	54.56	56.20	23.80	35.47	38.20	44.80	57.41	58.40	40.70	51.05	54.90
+SC	44.20	56.71	65.20	49.00	<u>55.14</u>	<u>56.60</u>	26.00	<u>37.92</u>	<u>40.80</u>	45.60	56.45	58.40	41.20	51.56	<u>55.25</u>
+GymPRM	46.80	58.43	67.20	47.40	54.32	54.60	24.40	35.82	38.60	46.40	57.89	58.40	41.25	51.62	54.70
+RePRM	41.60	53.49	62.00	44.20	49.45	50.40	22.20	33.08	36.00	<u>48.80</u>	57.70	<u>60.80</u>	39.20	48.43	52.30
+CoPRM	49.00	61.56	72.20	51.80	58.28	59.40	26.00	38.01	41.40	52.80	63.70	66.40	44.90	55.39	59.85
Search-R1^{RL}															
+Greedy	47.60	59.40	67.80	42.00	47.32	46.80	23.80	33.90	36.20	42.40	54.58	56.80	38.95	48.80	51.90
+SC	45.20	57.59	65.40	40.80	45.98	46.60	24.40	33.35	<u>36.80</u>	44.00	56.03	57.60	38.60	48.24	51.60
+GymPRM	46.60	58.71	64.60	43.00	48.64	47.80	25.60	34.39	<u>36.80</u>	44.80	56.74	58.40	40.00	49.62	51.90
+RePRM	44.40	56.08	64.00	40.20	46.00	45.80	24.60	32.37	35.80	41.60	55.11	56.00	37.70	47.39	50.40
+CoPRM	48.20	60.50	68.60	43.40	49.04	49.60	<u>25.40</u>	34.92	39.00	49.60	62.31	64.80	41.65	51.69	55.50
R1-Searcher^{RL}															
+Greedy	44.00	56.24	62.60	46.60	52.76	53.00	<u>25.20</u>	<u>33.64</u>	<u>37.60</u>	45.60	56.52	60.00	<u>40.35</u>	49.79	53.30
+SC	44.40	57.15	63.20	47.80	53.19	53.60	25.20	33.29	36.80	43.20	55.04	58.40	40.15	49.67	53.00
+GymPRM	44.20	57.12	64.80	50.00	56.00	56.40	21.40	30.79	35.20	44.80	56.71	59.20	40.10	<u>50.16</u>	53.90
+RePRM	41.80	53.98	60.00	45.00	50.81	51.60	20.40	28.59	31.60	<u>44.80</u>	<u>55.73</u>	<u>59.20</u>	38.00	47.28	50.60
+CoPRM	46.80	60.61	69.40	<u>48.80</u>	56.27	56.80	26.20	36.64	40.00	41.60	53.42	58.40	40.85	51.74	56.15
RS-ReSearch^{SFT}															
+Greedy	46.60	58.99	68.20	51.80	58.14	59.20	26.00	<u>36.48</u>	<u>40.40</u>	<u>47.20</u>	<u>58.82</u>	59.20	<u>42.90</u>	<u>53.11</u>	<u>56.75</u>
+SC	46.00	58.11	67.20	52.40	58.97	58.60	24.60	34.80	36.60	46.40	59.00	59.20	42.35	52.72	55.40
+GymPRM	47.00	<u>60.19</u>	<u>68.60</u>	50.00	56.45	57.40	22.60	33.99	37.80	45.60	56.92	<u>59.20</u>	41.30	51.89	55.75
+RePRM	43.40	54.60	62.20	48.80	54.99	56.20	20.60	31.38	35.00	44.80	55.76	60.00	39.40	49.18	53.35
+CoPRM	50.20	63.38	71.80	52.40	<u>58.69</u>	<u>59.00</u>	<u>25.80</u>	37.38	41.20	49.60	59.73	63.20	44.50	54.80	58.80

Table 1: EM (%), F1 (%), and LJ (%) results on QA benchmarks. All PRM variants are based on Llama. Results for Qwen-based PRMs are provided in Appendix A. The best result in each section is in **bold**, the second-best is underlined. ^{PE} = Prompt Engineering, ^{SFT} = Supervised Fine-Tuning, ^{RL} = Reinforcement Learning.

Prompt Engineering: PE-ReSearch; (2) **Supervised Fine-Tuning:** RS-ReSearch; (3) **Reinforcement Learning:** ReSearch (Chen et al., 2025b), R1-Searcher (Song et al., 2025), and Search-R1 (Jin et al., 2025b). PE-ReSearch is our implementation of a prompt-engineering based search agent, using prompts consistent with ReSearch (Chen et al., 2025b). RS-ReSearch is fine-tuned on successful reasoning paths selected via rejection sampling from all reasoning trajectories generated during the data collection process described in Section 4.1.

Inference-Time Guidance Baselines: We compare CoPRM against existing inference-time guidance methods: (1) **Greedy**; (2) **Self-Consistency (SC)** (Wang et al., 2023); (3) **GymPRM** (Xiong et al., 2026); and (4) **RePRM** (Sun et al., 2025c). Implementation details for all baselines are provided in Appendix G.

We quantify the guidance capability of CoPRM across diverse agent types and datasets by pair-

ing each search agent with different inference-time guidance methods while keeping the base agent architecture constant.

Implementation Details. We employ ReSearch-Qwen-7B-Instruct (Chen et al., 2025b) to filter the HotpotQA training set via 8-round rollouts, yielding 654 questions with both successful and unsuccessful reasoning paths. From these questions, we collect our preference dataset of 10,000 preference pairs via the methodology in Section 4.1 (with $\tau = 0.75$, $\theta = 0.5$ and $k = 10$; see Appendix C for sensitivity analysis), using the same agent as the state generator and QwQ-32B (Qwen Team, 2025b) as the LLM ranker. For the PRM backbone, we primarily use Llama-3.1-8B-Instruct (Llama) (Llama Team, 2024) in Table 1, and additionally train on Qwen2.5-7B-Instruct (Qwen) to verify generalization (Appendix A). For the search agent backbone, prompt-engineering agents use Qwen2.5-7B-Instruct, and SFT-based agents are

Method	PE-ReSearch ^{PE}		ReSearch ^{RL}		Search-R1 ^{RL}		R1-Searcher ^{RL}		RS-ReSearch ^{SFT}	
	Acc	Rec	Acc	Rec	Acc	Rec	Acc	Rec	Acc	Rec
Greedy	2.29	2.28	2.05	3.05	<u>2.06</u>	2.50	0.72	2.18	3.49	<u>2.88</u>
SC	2.05	2.09	3.98	4.14	1.69	2.44	1.93	<u>2.73</u>	3.37	2.70
GymPRM	1.81	1.64	3.25	4.14	1.69	2.07	<u>2.17</u>	2.31	2.05	1.80
RePRM	0.84	0.57	3.49	<u>4.17</u>	1.57	2.10	1.45	1.30	2.05	1.28
CoPRM	3.37	2.45	4.46	4.54	2.29	2.76	3.98	3.50	3.61	3.10

Table 2: Answer accuracy (%) and evidence recall (%) results on BrowseComp-Plus across five search agents. For each agent and metric, the best result is in **bold**, and the second-best is underlined.

Method	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			Average		
	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
Collaborative Calibration	49.00	61.56	72.20	51.80	58.28	59.40	<u>26.00</u>	38.01	<u>41.40</u>	52.80	63.70	66.40	44.90	55.39	59.85
<i>Single-Signal Reward (Non-Collaborative)</i>															
Immediate	46.60	58.82	68.40	48.00	54.37	55.80	26.40	37.97	41.60	48.00	61.14	63.20	42.25	53.08	57.25
Search MC	45.00	58.17	68.40	45.80	52.83	53.40	25.20	37.94	41.20	41.60	53.48	56.00	39.40	50.61	54.75
Answer F1	44.20	56.05	64.00	46.80	52.77	53.00	21.80	32.43	36.20	44.00	55.25	58.40	39.20	49.13	52.90
Search MC $\cup$ Answer F1	48.20	60.41	69.00	48.00	54.12	55.60	25.80	36.59	39.20	49.60	58.13	62.40	42.90	52.31	56.55
<i>Collaborative Signals (Requiring Consensus)</i>															
Immediate $\wedge$ Answer F1	44.20	56.12	64.60	48.40	54.52	56.40	26.00	37.07	39.00	45.60	58.90	60.80	41.05	51.65	55.20
Immediate $\wedge$ Search MC	<u>48.40</u>	<u>61.44</u>	<u>69.60</u>	<u>50.80</u>	<u>56.74</u>	<u>58.20</u>	24.80	37.10	40.00	<u>50.40</u>	<u>61.57</u>	<u>63.20</u>	<u>43.60</u>	<u>54.21</u>	<u>57.75</u>

Table 3: Ablation study for Llama-based PRMs comparing the performance of PRMs trained with different preference data collection strategies when guiding ReSearch reasoning (Qwen-based results in Appendix B.1).

fine-tuned from this same checkpoint, while RL-based agents use the Qwen2.5-7B variants from their original works. During PRM-guided inference, we retrieve 5 documents per query and generate 10 candidates per step (max 10 iterations). Further details are provided in Appendix I.

5.2 Main Results

As shown in Table 1, CoPRM provides effective guidance for search agents regardless of their underlying training methodology. All relative improvements reported below represent the average performance gains over the strongest existing baseline for each respective agent across EM, F1, and LJ metrics to provide fair comparisons. Our results first confirm CoPRM’s strong performance when guiding its base agent, ReSearch, achieving an average improvement of 8.2% across all metrics. More importantly, CoPRM exhibits remarkable generalization capabilities across entirely different training methods. It successfully guides prompt-engineered agents (achieving an average improvement of 9.6%), SFT-based agents (yielding an average improvement of 3.5%), and agents trained with different RL-based methodologies, improving R1-Searcher by an average of 3.1% and Search-R1 by an average of 5.1%.

CoPRM exhibits strong generalization to out-of-

domain datasets, significantly improving agent performance on unseen benchmarks including 2WikiMultihopQA, MuSiQue, and Bamboogle despite being trained exclusively on HotpotQA. Furthermore, it demonstrates cross-architecture generalizability: as shown in Table 5, CoPRM remains highly effective when trained on a Qwen-based backbone. This consistent effectiveness demonstrates that CoPRM learns transferable principles of reasoning quality rather than overfitting to specific architectures or datasets. Furthermore, as shown in Table 2, CoPRM achieves the best answer accuracy and evidence recall for all five search agents on BrowseComp-Plus.

5.3 Ablation Study

To investigate the necessity of collaborative synergy between immediate quality assessments and outcome-informed evaluations and to justify our *outcome-informed gating* mechanism (Eq. 6), we trained multiple PRMs using different data collection strategies. We distinguish two types of signal integration: **Union** ($\cup$) simply merges preference pairs from different sources without requiring agreement, while **Consensus** ($\wedge$) requires agreement between signals before generating preferences. We compare our **Collaborative Calibration** approach against individual components, their

Method	EM	F1	LJ
Outcome-informed gating (Ours)	44.90	55.39	59.85
Remove Cross-Type Pairs	43.20	54.31	57.85
Directly use LLM Rank	41.15	51.84	56.25
Directly Compare MC & F1	43.45	53.85	57.85
Gating $\wedge$ LLM Rank	42.80	53.49	57.55
(MC vs F1) $\wedge$ LLM Rank	43.65	54.51	58.50

Table 4: Ablation study comparing the performance of PRMs trained with different cross-type preference calculation strategies when guiding ReSearch reasoning (Average results, full results in Appendix B.2).

union, and consensus-based combinations. Here, **Immediate** corresponds to *Immediate Quality Assessment*, **Search MC** to *MC Rollouts for Search Steps*, and **Answer F1** to *GT F1 Scoring for Answer Steps*. All strategies generate 10k preference pairs for training, except for Answer F1 which, due to its sparse reward nature, generates 639 preference pairs. The resulting PRMs are used to guide ReSearch reasoning.

Effectiveness of Collaborative Synergy. As shown in Table 3, the PRM trained with our Collaborative Calibration data consistently achieves the best overall performance compared to all component combinations across most metrics and datasets. This validates that our Collaborative Calibration framework successfully captures the complementary strengths of each component: (1) Immediate assessment ensures locally sound reasoning quality. (2) Search MC provides strategic outcome-oriented verification. (3) Answer F1 serves as an objective quality gate for final answers.

Effectiveness of Outcome-Informed Gating. As shown in Table 4, our results highlight three key findings: (1) Removing these pairs causes a notable performance decline, confirming their role in balancing exploration and exploitation. (2) Relying on immediate quality fails for cross-type decisions due to the difficulty of ranking heterogeneous search and answer steps on a single scale. (3) Direct comparison of MC estimates and F1 scores underperforms due to distribution mismatches, whereas our gating mechanism effectively treats answer quality as a sufficiency threshold.

5.4 Effect of Training Data Size

Our CoPRM framework demonstrates strong data efficiency (see detailed results in Appendix E.1). As shown in Figure 3, CoPRM consistently outperforms both baseline PRMs and the Self-

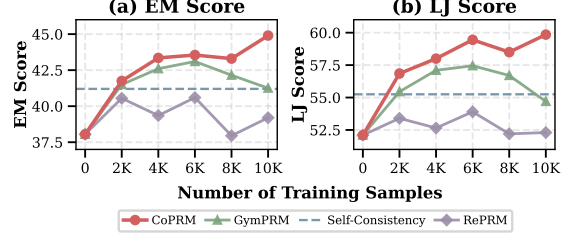


Figure 3: Performance comparison of ReSearch guided by PRMs trained on varying amounts of data.

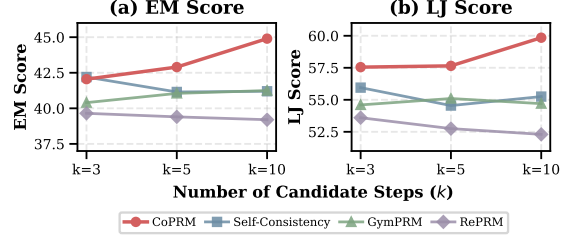


Figure 4: Performance comparison of PRM-guided ReSearch with varying numbers of candidate steps (k).

Consistency baseline across all training data sizes, achieving clear advantages with as few as 2,000 training samples. Baseline methods plateau or even decline with more data, as scaling amplifies the assessment unreliability inherent in single-source supervision rather than improving generalization. In contrast, CoPRM exhibits overall improvements across the evaluated data sizes, as the consensus between complementary supervision signals effectively filters out unreliable labels, resulting in more robust training data than single-source alternatives.

5.5 Impact of Candidate Sampling

To evaluate the impact of candidate sampling on our CoPRM-guided framework, we systematically investigate how the number of candidate steps sampled at each step affects overall performance. As demonstrated in Figure 4, CoPRM exhibits strong discriminative capabilities and consistent performance scaling. CoPRM consistently outperforms all baseline methods across all candidate sampling configurations, demonstrating the effectiveness of our framework. Furthermore, as the number of candidate steps increases from 3 to 10, both CoPRM variants show steady performance improvements (see additional results in Appendix E.2), suggesting that CoPRM effectively leverages expanded action spaces for better guidance.

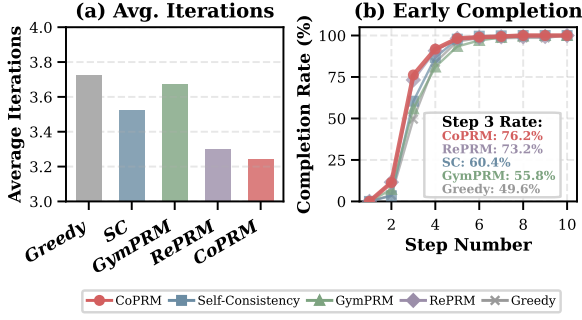


Figure 5: Inference efficiency analysis of PRM-guided ReSearch across 500 HotpotQA samples. CoPRM achieves the lowest average iteration count and highest early completion rate.

5.6 Inference Efficiency

Our inference efficiency analysis reveals that CoPRM’s process-level guidance translates strategic computational investment into superior task completion efficiency. As shown in Figure 5, CoPRM demonstrates two critical efficiency advantages: **(1) Reduced Average Iterations:** CoPRM achieves the lowest average number of iterations required for task completion (3.24 iterations), representing a 12.9% reduction in iteration count compared to the Greedy baseline (3.72 iterations); and **(2) Enhanced Early Completion Rate:** 76.2% of tasks are completed within 3 steps or fewer, substantially outperforming other baselines. These improvements indicate CoPRM’s ability to identify optimal reasoning paths while avoiding unproductive exploration. For more detailed efficiency analysis, please refer to Appendix K.

6 Conclusion

In this paper, we present CoPRM, a novel process reward model that addresses the critical limitations in existing PRMs for search agents: the trade-off between global solvability and local validity. The core innovation lies in our Collaborative Calibration methodology for generating high-quality training data by synthesizing immediate quality signals with outcome-informed signals. This enables CoPRM to deliver strategic and precise guidance, as demonstrated by consistent performance improvements across diverse search agents, exhibiting superior cross-domain generalization and efficiency.

Limitations

Despite CoPRM’s effectiveness in enhancing search agent reasoning, there are limitations to the

current work. While our Collaborative Calibration methodology successfully generates high-quality preference data through the automated synthesis of immediate and outcome-informed signals, the potential role of human-annotated data remains underexplored; incorporating human expert annotations could potentially further refine the reward model’s discriminative ability in highly nuanced scenarios.

Additionally, due to computational resource constraints, our current work focuses primarily on the application of CoPRM in inference-time guidance algorithms. Although Process Reward Models have the potential to offer dense, step-level rewards to enhance Reinforcement Learning (RL) training scenarios, the integration of CoPRM into the RL training loop for search agents remains an open area for future research. Future work could use CoPRM to provide fine-grained process rewards during RL training and extend its calibrated preference signals to multilingual and other preference-sensitive settings (Chen et al., 2025a; Yang et al., 2025a,b; Ye et al., 2026; Yu et al., 2025). Beyond multi-hop QA, another direction is to extend CoPRM to knowledge-grounded generation and evaluate its guidance across hierarchical scientific reasoning settings (Ren et al., 2023; Zhang et al., 2025c).

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant No. 62476271. We sincerely thank the anonymous reviewers for their valuable feedback and constructive suggestions.

References

- Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. [Self-rag: Learning to retrieve, generate, and critique through self-reflection](#). In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*.
- Hyunjoo Chae, Seonghwan Kim, Junhee Cho, Seungone Kim, Seungjun Moon, Gyeom Hwangbo, Dongha Lim, Minjin Kim, Yeonjun Hwang, Minju Gwak, Dongwook Choi, Minseok Kang, Gwanhoon Im, ByeongUng Cho, Hyojun Kim, Jun Hee Han, Taeyoon Kwon, Minju Kim, Beong-woo Kwak, and 2 others. 2025. [Web-shepherd: Advancing prms for reinforcing web agents](#). In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems*.

- 2025, *NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*.
- Jianghao Chen, Wei Sun, Qixiang Yin, Lingxing Kong, Zhixing Tan, and Jiajun Zhang. 2025a. [ACE-RL: adaptive constraint-enhanced reward for long-form generation reinforcement learning](#). *CoRR*, abs/2509.04903.
- 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. 2025b. [ReSearch: Learning to reason with search for LLMs via reinforcement learning](#). In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*.
- Zijian Chen, Xueguang Ma, Shengyao Zhuang, Ping Nie, Kai Zou, Sahel Sharifymoghaddam, Andrew Liu, Joshua Green, Kshama Patel, Ruoxi Meng, Mingyi Su, Yanxi Li, Haoran Hong, Xinyu Shi, Xuye Liu, Hosna Oyarhoseini, Nandan Thakur, Crystina Zhang, Luyu Gao, and 2 others. 2026. [BrowseComp-plus: A fair and disentangled evaluation benchmark for deep search agents](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 22349–22370, San Diego, California, United States. Association for Computational Linguistics.
- Sanjiban Choudhury. 2025. [Process Reward Models for LLM Agents: Practical Framework and Directions](#). *arXiv preprint arXiv:2502.10325*.
- Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, and 48 others. 2023. [Palm: Scaling language modeling with pathways](#). *J. Mach. Learn. Res.*, 24:240:1–240:113.
- Tristan Cinqun, Geoff Pleiss, and Agustinus Kristiadi. 2025. [Limits of PRM-Guided Tree Search for Mathematical Reasoning with LLMs](#). *arXiv preprint arXiv:2510.20272*.
- Wenfeng Feng, Chuzhan Hao, Yuewei Zhang, Guochao Jiang, and Jingyi Song. 2025. [AirRAG: Autonomous Strategic Planning and Reasoning Steer Retrieval Augmented Generation](#). In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 18934–18953, Suzhou, China. Association for Computational Linguistics.
- Bofei Gao, Zefan Cai, Runxin Xu, Peiyi Wang, Ce Zheng, Runji Lin, Keming Lu, Dayiheng Liu, Chang Zhou, Wen Xiao, Tianyu Liu, and Baobao Chang. 2025a. [LLM Critics Help Catch Bugs in Mathematics: Towards a Better Mathematical Verifier with Natural Language Feedback](#). In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 14588–14604, Vienna, Austria. Association for Computational Linguistics.
- Yunfan Gao, Yun Xiong, Yijie Zhong, Yuxi Bi, Ming Xue, and Haofen Wang. 2025b. [Synergizing RAG and Reasoning: A Systematic Review](#). *arXiv preprint arXiv:2504.15909*.
- Gemini Team. 2025. [Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities](#). *CoRR*, abs/2507.06261.
- Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and Mao Yang. 2025. [rstar-math: Small llms can master math reasoning with self-evolved deep thinking](#). In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*.
- Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. [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, Barcelona, Spain (Online). International Committee on Computational Linguistics.
- Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. 2023. [Survey of hallucination in natural language generation](#). *ACM Comput. Surv.*, 55(12):248:1–248:38.
- Jinhao Jiang, Jiayi Chen, Junyi Li, Ruiyang Ren, Shijie Wang, Wayne Xin Zhao, Yang Song, and Tao Zhang. 2025. [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, Albuquerque, New Mexico. Association for Computational Linguistics.
- Bowen Jin, Jinsung Yoon, Priyanka Kargupta, Sercan O. Arik, and Jiawei Han. 2025a. [An Empirical Study on Reinforcement Learning for Reasoning-Search Interleaved LLM Agents](#). *arXiv preprint arXiv:2505.15117*.
- Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025b. [Search-r1: Training LLMs to reason and leverage search engines with reinforcement learning](#). In *Second Conference on Language Modeling*.
- Jiajie Jin, Yutao Zhu, Zhicheng Dou, Guanting Dong, Xinyu Yang, Chenghao Zhang, Tong Zhao, Zhao

- Yang, and Ji-Rong Wen. 2025c. [Flashrag: A modular toolkit for efficient retrieval-augmented generation research](#). In *Companion Proceedings of the ACM on Web Conference 2025, WWW 2025, Sydney, NSW, Australia, 28 April 2025 - 2 May 2025*, pages 737–740. ACM.
- Leslie Pack Kaelbling, Michael L. Littman, and Andrew W. Moore. 1996. [Reinforcement learning: A survey](#). *J. Artif. Intell. Res.*, 4:237–285.
- Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. [Dense Passage Retrieval for Open-Domain Question Answering](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.
- Haitao Li, Qian Dong, Junjie Chen, Huixue Su, Yujia Zhou, Qingyao Ai, Ziyi Ye, and Yiqun Liu. 2024a. [LLMs-as-Judges: A Comprehensive Survey on LLM-based Evaluation Methods](#). *arXiv preprint arXiv:2412.05579*.
- Junyi Li, Jie Chen, Ruiyang Ren, Xiaoxue Cheng, Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2024b. [The Dawn After the Dark: An Empirical Study on Factuality Hallucination in Large Language Models](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10879–10899, Bangkok, Thailand. Association for Computational Linguistics.
- Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. [Search-o1: Agentic Search-Enhanced Large Reasoning Models](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5420–5438, Suzhou, China. Association for Computational Linguistics.
- Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. [Let’s verify step by step](#). In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*.
- Minhua Lin, Zongyu Wu, Zhichao Xu, Hui Liu, Xianfeng Tang, Qi He, Charu Aggarwal, Hui Liu, Xiang Zhang, and Suhang Wang. 2025. [A Comprehensive Survey on Reinforcement Learning-based Agentic Search: Foundations, Roles, Optimizations, Evaluations, and Applications](#). *arXiv preprint arXiv:2510.16724*.
- Yinhong Liu, Han Zhou, Zhijiang Guo, Ehsan Shareghi, Ivan Vulić, Anna Korhonen, and Nigel Collier. 2024. [Aligning with human judgement: The role of pairwise preference in large language model evaluators](#). In *First Conference on Language Modeling*.
- Llama Team. 2024. [The llama 3 herd of models](#). *CoRR*, abs/2407.21783.
- Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, and Abhinav Rastogi. 2024. [Improve Mathematical Reasoning in Language Models by Automated Process Supervision](#). *arXiv preprint arXiv:2406.06592*.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. [Training language models to follow instructions with human feedback](#). In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.
- Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. 2023. [Measuring and Narrowing the Compositionality Gap in Language Models](#). In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711, Singapore. Association for Computational Linguistics.
- Qwen Team. 2025a. [Qwen3 technical report](#). *CoRR*, abs/2505.09388.
- Qwen Team. 2025b. [Qwq-32b: Embracing the power of reinforcement learning](#).
- Zixuan Ren, Yang Zhao, and Chengqing Zong. 2023. [Towards informative open-ended text generation with dynamic knowledge triples](#). In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3189–3203, Singapore. Association for Computational Linguistics.
- Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. [Toolformer: Language models can teach themselves to use tools](#). In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.
- Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. [Enhancing Retrieval-Augmented Large Language Models with Iterative Retrieval-Generation Synergy](#). In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9248–9274, Singapore. Association for Computational Linguistics.
- Aditi Singh, Abul Ehtesham, Saket Kumar, Tala Talaei Khoei, and Athanasios V. Vasilakos. 2026. [Agentic retrieval-augmented generation: A survey on agentic rag](#). *arXiv preprint arXiv:2501.09136*.
- Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2025. [Scaling LLM test-time compute](#)

- optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*.
- Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Jirong Wen. 2025. [R1-Searcher: Incentivizing the Search Capability in LLMs via Reinforcement Learning](#). *arXiv preprint arXiv:2503.05592*.
- Wei Sun, Qianlong Du, Fuwei Cui, and Jiajun Zhang. 2025a. [An efficient and precise training data construction framework for process-supervised reward model in mathematical reasoning](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4292–4305, Vienna, Austria. Association for Computational Linguistics.
- Zhongxiang Sun, Qipeng Wang, Haoyu Wang, Xiao Zhang, and Jun Xu. 2025b. [Detection and Mitigation of Hallucination in Large Reasoning Models: A Mechanistic Perspective](#). *arXiv preprint arXiv:2505.12886*.
- Zhongxiang Sun, Qipeng Wang, Weijie Yu, Xiaoxue Zang, Kai Zheng, Jun Xu, Xiao Zhang, Yang Song, and Han Li. 2025c. [ReARTeR: Retrieval-Augmented Reasoning with Trustworthy Process Rewarding](#). In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '25*, pages 1251–1261, New York, NY, USA. Association for Computing Machinery.
- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. [Musique: Multi-hop questions via single-hop question composition](#). *Trans. Assoc. Comput. Linguistics*, 10:539–554.
- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. [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 (Volume 1: Long Papers)*, pages 10014–10037, Toronto, Canada. Association for Computational Linguistics.
- Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. 2022. [Solving math word problems with process- and outcome-based feedback](#). *arXiv preprint arXiv:2211.14275*.
- Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. [Text embeddings by weakly-supervised contrastive pre-training](#). *CoRR*, abs/2212.03533.
- Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. [Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439, Bangkok, Thailand. Association for Computational Linguistics.
- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. [Self-consistency improves chain of thought reasoning in language models](#). In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*.
- Zhiheng Xi, Chenyang Liao, Guanyu Li, Zhihao Zhang, Wenxiang Chen, Binghai Wang, Senjie Jin, Yuhao Zhou, Jian Guan, Wei Wu, Tao Ji, Tao Gui, Qi Zhang, and Xuanjing Huang. 2026. [Agentprm: Process reward models for LLM agents via step-wise promise and progress](#). In *Proceedings of the ACM Web Conference 2026, WWW 2026, Dubai, United Arab Emirates, originally scheduled for April 13-17, 2026, rescheduled for June 29 - July 3, 2026*, pages 4184–4195. ACM.
- Lu Xiang, Yang Zhao, Yaping Zhang, and Chengqing Zong. 2025. [A survey of large language models in discipline-specific research: Challenges, methods and opportunities](#). *CoRR*, abs/2507.08425.
- Guangzhi Xiong, Qiao Jin, Xiao Wang, Yin Fang, Haolin Liu, Yifan Yang, Fangyuan Chen, Zhixing Song, Dengyu Wang, Minjia Zhang, Zhiyong Lu, and Aidong Zhang. 2026. [Supervising the search process produces reliable and generalizable information-seeking agents](#). *arXiv preprint arXiv:2502.13957*.
- An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. [Qwen2.5 technical report](#). *CoRR*, abs/2412.15115.
- Wen Yang, Junhong Wu, Chen Wang, Chengqing Zong, and Jiajun Zhang. 2025a. [Implicit cross-lingual rewarding for efficient multilingual preference alignment](#). In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 21125–21147, Vienna, Austria. Association for Computational Linguistics.
- Wen Yang, Junhong Wu, Chen Wang, Chengqing Zong, and Jiajun Zhang. 2025b. [Language imbalance driven rewarding for multilingual self-improving](#). In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. [HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages

- 2369–2380, Brussels, Belgium. Association for Computational Linguistics.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023. [React: Synergizing reasoning and acting in language models](#). In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*.
- Jing Ye, Xinpei Zhao, Lu Xiang, Yaping Zhang, and Chengqing Zong. 2026. [Listening to the echo: User-reaction aware policy optimization via scalar-verbal hybrid reinforcement learning](#). *CoRR*, abs/2603.15434.
- Donglei Yu, Yang Zhao, Jie Zhu, Yangyifan Xu, Yu Zhou, and Chengqing Zong. 2025. [Simulpl: Aligning human preferences in simultaneous machine translation](#). In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*.
- Duzhen Zhang, Zhong-Zhi Li, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Xiuyi Chen, Yingying Zhang, Fei Yin, Jiahua Dong, Zhijiang Guo, Le Song, and Cheng-Lin Liu. 2026. [From system 1 to system 2: A survey of reasoning large language models](#). *IEEE Trans. Pattern Anal. Mach. Intell.*, 48(3):3335–3354.
- Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. 2025a. [Generative verifiers: Reward modeling as next-token prediction](#). In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*.
- Yao Zhang, Yu Wu, Haowei Zhang, Weiguo Li, Haokun Chen, Jingpei Wu, Guohao Li, Zhen Han, and Volker Tresp. 2025b. [GroundedPRM: Tree-Guided and Fidelity-Aware Process Reward Modeling for Step-Level Reasoning](#). *arXiv preprint arXiv:2510.14942*.
- Yaping Zhang, Qixuan Zhang, Xingquan Zhang, Zhiyuan Chen, Wenwen Zhuang, Yupu Liang, Lu Xiang, Yang Zhao, Jiajun Zhang, Yu Zhou, and Chengqing Zong. 2025c. [Hiscibench: A hierarchical multi-disciplinary benchmark for scientific intelligence from reading to discovery](#). *CoRR*, abs/2512.22899.
- Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025d. [The lessons of developing process reward models in mathematical reasoning](#). In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 10495–10516, Vienna, Austria. Association for Computational Linguistics.
- Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Zican Dong, Yupeng Hou, Beichen Zhang, Yingqian Min, Junjie Zhang, Peiyu Liu, Xiaolei Wang, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, and 4 others. 2026. [A survey of large language models](#). *Frontiers Comput. Sci.*, 20(12):2012627.
- Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025a. [ProcessBench: Identifying Process Errors in Mathematical Reasoning](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1009–1024, Vienna, Austria. Association for Computational Linguistics.
- Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. 2025b. [DeepResearcher: Scaling Deep Research via Reinforcement Learning in Real-world Environments](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 414–431, Suzhou, China. Association for Computational Linguistics.

A Cross-architecture Generalizability

To assess the cross-architecture generalizability of our proposed method, we further evaluate CoPRM using Qwen2.5-7B-Instruct (Yang et al., 2024) as the base model for the process reward model. As shown in Table 5, Qwen-CoPRM consistently outperforms both the strongest baselines and other PRM variants across diverse search agents and datasets. These results demonstrate that CoPRM’s effectiveness is not limited to a specific model architecture but rather stems from the high-quality preference data collected through our Collaborative Calibration methodology.

B Full Ablation Study Results

B.1 Qwen-based Ablation Study Results

In Table 6, we present the complete ablation study results for Qwen2.5-7B-Instruct-based PRMs. These results complement the Llama-3.1-8B-Instruct results shown in the main text, demonstrating the effectiveness of collaborative synergy, where our methodology consistently outperforms individual components and their combinations across different base models.

B.2 Full Cross-Type Ablation Study Results

Table 7 presents the complete cross-type ablation study results across all four benchmarks, complementing the averaged results shown in the main text. These detailed results demonstrate that our *outcome-informed gating* mechanism consistently outperforms alternative cross-type preference calculation strategies.

B.3 Additional Ablation Results

To further evaluate our preference data construction process, we consider three controlled variants while keeping all other training and inference settings unchanged. We replace Answer F1 in the outcome-informed gating mechanism with semantic correctness labels produced by Qwen3-32B (Qwen Team, 2025a), replace the QwQ-32B ranker in Stage 1 with Qwen3-8B, or replace rank-guided state advancement with random advancement.

Compared with the default CoPRM results in Table 7, LJ-based gating yields the smallest average performance decrease ($\Delta = -0.80/-0.71/-1.40$ points in EM/F1/LJ), indicating limited sensitivity to replacing lexical F1 with semantic correctness labels. Replacing the Stage 1 ranker with Qwen3-8B and using random state advancement lead to

larger decreases ($\Delta = -2.15/-2.06/-2.40$ and $-2.80/-2.51/-2.85$ points, respectively), suggesting that both ranker strength and rank-guided advancement contribute to guidance quality.

C Sensitivity Analysis of Hyperparameters

To understand the impact of key hyperparameters on our method’s performance, we conduct a comprehensive sensitivity analysis focusing on three critical parameters: the rollout evaluation threshold τ for Monte Carlo outcome assessment, the preference filtering threshold θ for outcome-gated preference decisions, and the rollout count N for Monte Carlo sampling frequency. We evaluate τ and θ across both Llama and Qwen models, while the rollout count analysis is conducted on the Llama model, with an expanded hyperparameter range to ensure robust and generalizable findings.

C.1 Monte Carlo Rollout Threshold (τ) Analysis

We fix $\theta = 0.5$ and vary $\tau \in \{0.25, 0.5, 0.75, 1.0\}$ across both Llama and Qwen models. As shown in Table 9, $\tau = 0.75$ consistently achieves the best overall performance across both architectures. Lenient thresholds ($\tau \leq 0.5$) introduce noisy training signals, while the strict threshold ($\tau = 1.0$) excludes valuable intermediate reasoning steps that lead to partially correct answers. The moderate threshold effectively filters low-quality steps while preserving nuanced reasoning patterns.

C.2 Outcome-Gated Preference Threshold (θ) Analysis

We fix $\tau = 0.75$ and vary $\theta \in \{0.25, 0.5, 0.75, 1.0\}$ across both Llama and Qwen models. The cross-type preference follows the gating mechanism defined in Eq. 6. As shown in Table 10, $\theta = 0.5$ yields the best overall performance across both architectures. A lenient threshold ($\theta = 0.25$) causes premature answer acceptance, while strict thresholds ($\theta \geq 0.75$) reject high-quality but imperfect answers in favor of less informative search steps.

C.3 Monte Carlo Rollout Count Analysis

We fix $\tau = 0.75$ and $\theta = 0.5$, and vary $N \in \{6, 8, 10, 16\}$ on the Llama model. As shown in Table 11, $N = 8$ achieves the best overall performance. While $N = 6$ yields competitive results,

Method	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			Average		
	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
PE-ReSearch^{PE}															
+greedy	24.00	37.32	59.40	23.20	34.35	46.00	13.80	22.68	29.60	21.60	35.35	48.80	20.65	32.43	45.95
+Self-Consistency	37.40	46.99	57.40	38.60	46.39	49.00	17.40	26.60	30.20	36.80	47.54	50.40	32.55	41.88	46.75
+Qwen-GymPRM	30.60	42.29	54.80	31.40	39.23	41.80	13.60	23.69	28.00	42.40	54.38	<u>58.40</u>	29.50	39.90	45.75
+Qwen-RePRM	22.60	35.78	53.80	24.60	34.11	39.60	13.60	21.48	25.60	35.20	47.47	60.00	24.00	34.71	44.75
+Qwen-CoPRM	<u>34.80</u>	<u>45.32</u>	57.20	39.60	48.14	49.60	18.00	27.68	32.00	<u>40.00</u>	<u>53.50</u>	57.60	33.10	43.66	49.10
ReSearch^{RL}															
+greedy	45.40	56.77	66.80	48.80	54.56	56.20	23.80	35.47	38.20	44.80	57.41	58.40	40.70	51.05	54.90
+Self-Consistency	44.20	56.71	65.20	49.00	55.14	<u>56.60</u>	<u>26.00</u>	<u>37.92</u>	<u>40.80</u>	45.60	56.45	58.40	<u>41.20</u>	51.56	55.25
+Qwen-GymPRM	44.80	<u>58.02</u>	<u>67.20</u>	49.40	55.80	56.80	23.20	34.98	38.60	47.20	61.33	62.40	41.15	52.53	56.25
+Qwen-RePRM	<u>46.00</u>	<u>57.70</u>	<u>66.80</u>	<u>47.80</u>	54.15	54.00	22.40	35.18	37.80	44.80	55.89	58.40	40.25	50.73	54.25
+Qwen-CoPRM	46.60	59.70	70.80	50.60	56.65	56.80	28.00	38.80	42.60	50.40	<u>60.11</u>	63.20	43.90	53.82	58.35
Search-R1^{RL}															
+greedy	47.60	59.40	67.80	42.00	47.32	46.80	23.80	33.90	36.20	42.40	54.58	56.80	38.95	48.80	51.90
+Self-Consistency	45.20	57.59	65.40	40.80	45.98	46.60	24.40	33.35	<u>36.80</u>	44.00	56.03	57.60	38.60	48.24	51.60
+Qwen-GymPRM	45.60	57.66	64.80	42.60	48.81	<u>49.40</u>	24.00	33.34	<u>36.00</u>	<u>45.60</u>	<u>57.04</u>	<u>58.40</u>	39.45	<u>49.21</u>	<u>52.15</u>
+Qwen-RePRM	<u>46.60</u>	<u>58.02</u>	<u>65.60</u>	42.00	47.41	48.20	25.20	<u>33.86</u>	36.00	44.00	55.11	56.80	<u>39.45</u>	48.60	51.65
+Qwen-CoPRM	45.00	57.71	65.20	45.80	50.98	51.20	<u>24.60</u>	33.84	38.40	46.40	57.99	60.00	40.45	50.13	53.70
R1-Searcher^{RL}															
+greedy	44.00	56.24	62.60	46.60	52.76	53.00	<u>25.20</u>	33.64	<u>37.60</u>	45.60	56.52	60.00	40.35	49.79	53.30
+Self-Consistency	44.40	57.15	63.20	47.80	53.19	53.60	<u>25.20</u>	33.29	<u>36.80</u>	43.20	55.04	58.40	40.15	49.67	53.00
+Qwen-GymPRM	45.00	<u>57.95</u>	<u>65.60</u>	46.00	51.86	52.40	23.00	30.90	34.40	40.00	52.98	56.80	38.50	48.42	52.30
+Qwen-RePRM	<u>46.20</u>	57.41	64.20	48.00	53.76	54.80	23.20	31.55	35.00	46.40	<u>57.14</u>	<u>60.80</u>	40.95	49.97	53.70
+Qwen-CoPRM	47.80	60.64	67.40	49.40	56.02	57.20	26.40	36.35	39.00	44.00	57.87	61.60	41.90	52.72	56.30
RS-ReSearch^{SFT}															
+greedy	46.60	58.99	68.20	<u>51.80</u>	58.14	59.20	26.00	36.48	40.40	<u>47.20</u>	58.82	59.20	42.90	<u>53.11</u>	<u>56.75</u>
+Self-Consistency	46.00	58.11	67.20	52.40	58.97	<u>58.60</u>	24.60	34.80	36.60	46.40	<u>59.00</u>	<u>59.20</u>	42.35	52.72	55.40
+Qwen-GymPRM	<u>47.80</u>	<u>60.60</u>	<u>69.80</u>	50.20	57.81	57.60	23.00	33.96	38.00	46.40	57.22	58.40	41.85	52.40	55.95
+Qwen-RePRM	46.20	57.12	65.00	47.20	53.05	53.40	24.20	33.38	37.60	44.80	55.39	57.60	40.60	49.74	53.40
+Qwen-CoPRM	49.20	62.01	70.20	50.20	57.43	<u>58.60</u>	26.40	37.47	42.00	48.80	59.68	61.60	43.65	54.15	58.10

Table 5: CoPRM’s cross-architecture generalizability: Qwen-based PRMs guiding search agents. EM (%), F1 (%), and LJ (%) results on multi-hop QA benchmarks. The best result in each section is in **bold**, the second-best is underlined. ^{PE} = Prompt Engineering, ^{SFT} = Supervised Fine-Tuning, ^{RL} = Reinforcement Learning.

Method	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			Average		
	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
Qwen2.5-7B-Instruct															
Collaborative Calibration	46.60	59.70	70.80	50.60	<u>56.65</u>	<u>56.80</u>	28.00	38.80	42.60	<u>50.40</u>	60.11	63.20	43.90	53.82	58.35
<i>Single-Signal Reward (Non-Collaborative)</i>															
Immediate	<u>45.60</u>	58.33	67.60	47.20	53.98	55.00	<u>24.80</u>	<u>36.79</u>	<u>40.40</u>	52.00	63.09	64.80	<u>42.40</u>	<u>53.05</u>	56.95
Search MC	43.20	55.70	64.20	44.40	50.24	50.80	22.80	33.01	37.60	43.20	54.21	56.80	38.40	48.29	52.35
Answer F1	41.80	54.17	63.00	45.00	50.84	51.80	22.80	33.21	36.20	40.80	52.52	52.00	37.60	47.69	50.75
Search MC $\cup$ Answer F1	44.20	56.55	66.20	47.20	52.90	53.80	24.60	36.24	39.60	52.00	<u>61.92</u>	64.80	42.00	51.90	56.10
<i>Collaborative Signals (Requiring Consensus)</i>															
Immediate $\wedge$ Answer F1	43.40	56.09	64.80	<u>50.20</u>	55.59	<u>56.80</u>	23.60	34.65	38.40	48.80	58.90	60.80	41.50	51.31	55.20
Immediate $\wedge$ Search MC	<u>45.60</u>	<u>58.51</u>	<u>68.60</u>	50.60	56.90	57.40	24.20	35.63	38.80	48.00	60.21	<u>64.00</u>	42.10	52.81	<u>57.20</u>

Table 6: Full ablation study results for Qwen2.5-7B-Instruct-based PRMs comparing the performance of PRMs trained with different preference data collection strategies when guiding ReSearch reasoning. Best results are in **bold**, the second-best is underlined.

performance degrades at $N \geq 10$. This degradation likely occurs because excessive rollouts cause steps with intrinsically different quality levels to achieve similar success rates through multiple attempts, thereby reducing the distinctiveness of quality signals for preference learning.

Notably, across all hyperparameter settings examined, CoPRM-guided agents consistently outperform the strongest baseline Self-Consistency (41.20 EM, 51.56 F1, 55.25 LJ). Even with sub-optimal configurations (e.g., Qwen with $\theta = 1.0$: 42.95 EM, 52.44 F1, 55.90 LJ), our method sur-

Method	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			Average		
	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
Llama-3.1-8B-Instruct															
Outcome-informed gating (Ours)	49.00	61.56	72.20	51.80	<u>58.28</u>	59.40	26.00	38.01	41.40	52.80	63.70	66.40	44.90	55.39	59.85
Remove Cross-Type Pairs	<u>48.20</u>	<u>61.18</u>	<u>70.40</u>	<u>51.60</u>	58.43	58.20	27.40	39.60	42.00	45.60	58.04	60.80	43.20	54.31	57.85
Directly use LLM Rank	46.20	58.77	69.00	47.80	54.06	55.80	23.40	35.43	39.40	47.20	59.09	60.80	41.15	51.84	56.25
Directly Compare MC & F1	45.80	59.22	68.80	50.60	57.08	<u>58.80</u>	<u>26.20</u>	37.69	40.60	<u>51.20</u>	61.41	<u>63.20</u>	43.45	53.85	57.85
Gating $\wedge$ LLM Rank	47.40	59.90	68.80	48.80	55.50	57.60	<u>27.00</u>	38.08	41.40	<u>48.00</u>	60.48	<u>62.40</u>	42.80	53.49	57.55
(MC vs F1) $\wedge$ LLM Rank	47.20	60.08	69.40	50.20	56.74	58.00	26.80	<u>39.18</u>	43.40	50.40	<u>62.05</u>	<u>63.20</u>	<u>43.65</u>	<u>54.51</u>	<u>58.50</u>
Qwen2.5-7B-Instruct															
Outcome-informed gating (Ours)	46.60	59.70	70.80	50.60	56.65	56.80	28.00	38.80	42.60	50.40	<u>60.11</u>	63.20	43.90	53.82	58.35
Remove Cross-Type Pairs	45.60	58.04	67.20	51.60	58.43	59.80	23.40	35.93	40.40	47.20	57.32	57.60	41.95	52.43	56.25
Directly use LLM Rank	47.00	59.06	<u>69.60</u>	46.40	53.15	54.40	25.80	37.25	41.00	<u>49.60</u>	61.15	<u>62.40</u>	42.20	52.65	56.85
Directly Compare MC & F1	<u>48.20</u>	<u>60.36</u>	69.40	48.80	55.83	57.60	<u>26.80</u>	37.24	40.00	48.80	59.55	63.20	43.15	<u>53.25</u>	<u>57.55</u>
Gating $\wedge$ LLM Rank	46.40	58.71	67.00	<u>51.00</u>	<u>57.78</u>	<u>59.00</u>	24.20	35.27	39.80	47.20	59.07	60.00	42.20	52.71	56.45
(MC vs F1) $\wedge$ LLM Rank	48.60	61.06	69.40	51.60	56.97	57.80	26.40	<u>37.59</u>	<u>41.60</u>	46.40	57.36	59.20	<u>43.25</u>	<u>53.25</u>	57.00

Table 7: Full cross-type ablation study results comparing the performance of PRMs trained with different cross-type preference calculation strategies across all four benchmarks. Best results are in **bold**, the second-best is underlined.

Method	HotpotQA			2Wiki			MuSiQue			Bamboogle			Avg		
	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
LJ-based Gating (Qwen3-32B)	46.60	60.23	70.00	51.00	57.22	58.00	27.60	39.10	42.60	51.20	62.19	63.20	44.10	54.68	58.45
Qwen3-8B Stage 1 Ranker	47.00	59.58	69.40	50.00	57.07	58.00	24.40	37.03	40.80	49.60	59.63	61.60	42.75	53.33	57.45
Random State Advancement	45.40	59.29	69.00	48.80	56.31	57.60	23.80	35.77	38.20	50.40	60.16	63.20	42.10	52.88	57.00

Table 8: Additional ablation study results comparing Llama-based PRMs trained with different preference data construction variants when guiding ReSearch reasoning across all four benchmarks. Results for the default CoPRM configuration are reported in Table 7.

Model	τ	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			Average		
		EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
Llama	0.25	47.00	59.77	69.40	49.60	56.47	57.80	26.80	37.67	41.00	48.00	59.97	61.60	42.85	53.47	57.45
	0.5	46.00	59.16	69.00	50.00	56.50	58.40	25.60	36.93	40.00	50.40	61.86	62.40	43.00	53.61	57.45
	0.75	49.00	61.56	72.20	51.80	58.28	59.40	26.00	38.01	41.40	52.80	63.70	66.40	44.90	55.39	59.85
	1.0	48.60	61.67	71.00	48.80	55.46	56.40	24.40	36.50	39.00	50.40	61.95	64.80	43.05	53.90	57.80
Qwen	0.25	46.00	58.72	67.20	48.00	54.13	55.40	24.20	35.86	39.60	48.80	59.20	61.60	41.75	51.98	55.95
	0.5	44.60	56.94	65.80	48.20	54.59	55.40	27.40	37.68	39.40	49.60	60.80	62.40	42.45	52.50	55.75
	0.75	46.60	59.70	70.80	50.60	56.65	56.80	28.00	38.80	42.60	50.40	60.11	63.20	43.90	53.82	58.35
	1.0	46.20	59.61	69.40	50.40	56.64	57.40	25.60	36.99	41.00	45.60	57.28	58.40	41.95	52.63	56.55

Table 9: Performance comparison of CoPRM trained on data collected with different rollout F1 thresholds (τ) with fixed $\theta = 0.5$ across Llama and Qwen models.

passes this baseline, demonstrating robustness to hyperparameter variations.

D PRM Scalability Analysis

To address potential concerns regarding the computational overhead of CoPRM’s additional evaluation mechanism during inference, we investigate two mitigation strategies: (1) employing smaller (3B) models as the PRM backbone to reduce the cost of reward calculation, and (2) using CoPRM to guide larger, more capable search agents (32B), where the relative overhead of the PRM becomes negligible compared to the agent’s generation cost.

D.1 Performance of 3B PRMs

We first evaluate the performance of using 3B-parameter models as the backbone for CoPRM. We train CoPRM on Qwen2.5-3B-Instruct and Llama-3.2-3B-Instruct using our collaborative preference data and use them to guide the ReSearch agent.

As shown in Table 12, both Qwen2.5-3B and Llama-3.2-3B based CoPRMs successfully guide the ReSearch agent, consistently outperforming both Greedy and Self-Consistency (SC) baselines. Notably, Llama-3.2-3B-CoPRM achieves an average EM of 43.25% and F1 of 53.34%, demonstrating that even smaller PRMs can capture the necessary process supervision signals to improve search agent inference. This suggests that deploying smaller PRMs is a viable strategy to miti-

Model	θ	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			Average		
		EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
Llama	0.25	48.60	61.64	70.80	49.00	55.92	57.20	26.80	39.28	43.00	45.60	58.29	59.20	42.50	53.78	57.55
	0.5	49.00	61.56	72.20	51.80	58.28	59.40	26.00	38.01	41.40	52.80	63.70	66.40	44.90	55.39	59.85
	0.75	47.20	60.19	70.00	49.80	55.75	56.20	27.20	37.27	41.20	48.00	59.51	62.00	43.05	53.18	57.35
	1.0	47.40	59.97	69.60	47.40	54.32	55.40	25.60	37.76	39.80	45.60	58.64	59.20	41.50	52.67	56.00
Qwen	0.25	47.40	59.47	68.80	47.60	53.42	53.60	26.20	37.03	40.60	47.20	60.66	62.40	42.10	52.65	56.35
	0.5	46.60	59.70	70.80	50.60	56.65	56.80	28.00	38.80	42.60	50.40	60.11	63.20	43.90	53.82	58.35
	0.75	46.60	58.67	68.60	49.40	56.58	57.80	27.20	37.45	40.20	52.00	63.55	65.60	43.80	54.06	58.05
	1.0	46.20	57.93	66.60	49.40	55.66	57.00	26.60	37.21	39.20	49.60	58.96	60.80	42.95	52.44	55.90

Table 10: Performance of ReSearch agent guided by CoPRM trained on data collected with different preference F1 thresholds (θ) with fixed $\tau = 0.75$ across Llama and Qwen models.

Rollout Count	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			Average		
	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
6	46.00	58.64	75.00	50.00	57.21	58.80	25.40	38.78	41.80	52.80	62.29	64.80	43.55	54.23	60.10
8	49.00	61.56	72.20	51.80	58.28	59.40	26.00	38.01	41.40	52.80	63.70	66.40	44.90	55.39	59.85
10	46.40	58.88	69.40	51.80	55.06	59.40	25.00	36.77	40.40	48.80	62.26	63.20	43.00	53.24	58.10
16	48.40	60.78	71.80	47.20	53.76	55.60	24.80	36.16	39.00	46.40	57.68	59.20	41.70	52.10	56.40

Table 11: Performance of ReSearch agent guided by Llama-CoPRM trained on data collected with different Monte Carlo rollout counts with fixed $\tau = 0.75$ and $\theta = 0.5$, measured by EM, F1, and LJ metrics across four datasets.

Method	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			Average		
	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
ReSearch (Greedy)	45.40	56.77	66.80	48.80	54.56	56.20	23.80	35.47	38.20	44.80	57.41	58.40	40.70	51.05	54.90
ReSearch (SC)	44.20	56.71	65.20	49.00	55.14	56.60	26.00	37.92	40.80	45.60	56.45	58.40	41.20	51.56	55.25
3B PRMs															
Qwen2.5-3B-CoPRM	45.60	58.85	67.80	49.80	56.69	57.40	24.20	35.71	39.80	49.60	59.88	62.40	42.30	52.78	56.85
Llama-3.2-3B-CoPRM	45.60	58.03	67.00	50.20	56.93	57.40	26.80	38.25	42.20	50.40	60.16	61.60	43.25	53.34	57.05

Table 12: Performance of 3B PRMs when guiding ReSearch-7B reasoning. Even with significantly reduced parameters, 3B CoPRMs consistently outperform baselines, offering a lightweight alternative for effective guidance.

gate computational overhead without compromising guidance quality.

D.2 Guiding Larger Agents (32B)

We further investigate the effectiveness of CoPRM when guiding a significantly larger and more capable search agent, ReSearch-Qwen2.5-32B-Instruct. This setup mimics a practical scenario where a smaller, specialized PRM guides a large general-purpose reasoning model.

Table 13 presents the results of using CoPRM to guide a 32B parameter search agent. CoPRM consistently outperforms the Greedy, Major Vote, and other PRM baselines (GymPRM, RePRM) across most metrics. Specifically, CoPRM achieves the highest average EM (49.20%), F1 (59.52%), and LJ (64.30%) scores. This demonstrates that our CoPRM, trained on high-quality collaborative preference data, generalizes well to larger model architectures and continues to provide valuable strategic guidance, effectively enhancing the capabilities of even powerful foundational models.

E Detailed Experiment Results

E.1 Effect of Training Data Size

The detailed results of training data size experiments for Llama-CoPRM across 4 datasets with EM, F1, and LJ metrics are shown in Table 14. The detailed results for Qwen-CoPRM are shown in Table 15. Additionally, the results for Llama-RePRM and Llama-GymPRM under varying training data sizes are presented in Table 16 and Table 17, respectively.

E.2 Impact of Candidate Sampling

The detailed results of candidate sampling experiments across 4 datasets with EM, F1, and LJ metrics are shown in Table 18.

E.3 Impact of Foundation Model Type

To understand the influence of foundation model type on CoPRM performance, we compare the effectiveness of base models versus instruction-tuned models using both Qwen-2.5-7B-base and Qwen-2.5-7B-instruct variants. Both models are trained

Method	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			Average		
	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
ReSearch-32B (Greedy)	49.00	61.87	71.20	54.00	59.51	59.80	32.40	41.88	46.40	55.20	69.11	70.40	47.65	58.09	61.95
ReSearch-32B (SC)	49.40	62.42	71.80	52.80	58.92	60.00	33.20	42.91	46.20	57.60	68.75	71.20	48.25	58.25	62.30
GymPRM	50.60	62.51	71.00	54.00	59.29	60.80	28.80	39.66	44.40	60.00	71.17	73.60	48.35	58.16	62.45
RePRM	44.40	57.13	65.60	50.20	55.25	56.80	29.40	39.23	42.60	59.20	70.01	72.00	45.80	55.41	59.25
CoPRM	51.40	64.39	74.00	55.80	61.27	63.00	31.20	41.53	45.80	58.40	70.89	74.40	49.20	59.52	64.30

Table 13: Performance of CoPRM guiding ReSearch-Qwen2.5-32B-Instruct. CoPRM provides robust guidance for larger models, achieving the highest average scores across metrics.

Training Samples	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			AVG		
	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
10,000	49.00	61.56	72.20	51.80	58.28	59.40	26.00	38.01	41.40	52.80	63.70	66.40	44.90	55.39	59.85
8,000	45.80	58.51	69.40	50.60	57.29	58.40	27.20	39.92	43.00	49.60	59.84	63.20	43.30	53.89	58.50
6,000	47.60	60.84	70.80	50.00	56.55	57.40	27.00	39.15	45.60	49.60	62.18	64.00	43.55	54.68	59.45
4,000	49.00	62.29	71.60	49.00	54.40	55.20	26.60	38.34	42.00	48.80	59.84	63.20	43.35	53.72	58.00
2,000	46.00	58.50	68.20	49.40	56.06	57.80	26.00	37.55	41.40	45.60	56.90	60.00	41.75	52.25	56.85
0	43.60	55.50	64.40	43.60	51.01	52.80	23.40	33.46	36.80	41.60	52.72	54.40	38.05	48.17	52.10

Table 14: Performance of ReSearch agent guided by Llama-CoPRM trained on varying amounts of training data, measured by EM, F1, and LJ metrics across four datasets.

Training Samples	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			AVG		
	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
10,000	46.60	59.70	70.80	50.60	56.65	56.80	28.00	38.80	42.60	50.40	60.11	63.20	43.90	53.82	58.35
8,000	46.00	58.36	69.40	49.20	55.78	57.60	25.60	36.54	43.00	47.20	58.83	60.80	42.00	52.38	57.70
6,000	43.20	56.42	65.60	48.60	54.99	56.00	25.40	36.53	40.80	48.80	61.09	63.20	41.50	52.26	56.40
4,000	46.60	59.70	69.00	50.40	56.16	58.20	26.80	38.25	41.60	46.40	56.98	61.60	42.55	52.77	57.60
2,000	43.60	56.14	66.20	47.40	53.14	53.40	26.80	36.78	40.80	49.60	61.36	62.40	41.85	51.86	55.70
0	44.20	55.55	64.40	43.80	49.37	50.20	22.80	32.55	36.60	43.20	56.83	57.60	38.50	48.58	52.20

Table 15: Performance of ReSearch agent guided by Qwen-CoPRM trained on varying amounts of training data, measured by EM, F1, and LJ metrics across four datasets.

Training Samples	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			AVG		
	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
10,000	41.60	53.49	62.00	44.20	49.45	50.40	22.20	33.08	36.00	48.80	57.70	60.80	39.20	48.43	52.30
8,000	41.60	54.63	64.00	44.00	50.28	51.20	23.80	34.89	37.60	42.40	53.98	56.00	37.95	48.45	52.20
6,000	41.60	53.68	62.60	46.80	52.36	53.60	22.80	33.37	37.00	51.20	62.08	62.40	40.60	50.37	53.90
4,000	45.60	56.53	65.80	45.80	52.24	53.20	23.60	34.20	37.20	42.40	52.56	54.40	39.35	48.88	52.65
2,000	42.80	55.14	64.60	44.60	50.49	51.00	24.40	34.75	37.20	50.40	60.27	60.80	40.55	50.16	53.40
0	43.60	55.50	64.40	43.60	51.01	52.80	23.40	33.46	36.80	41.60	52.72	54.40	38.05	48.17	52.10

Table 16: Performance of ReSearch agent guided by Llama-RePRM trained on varying amounts of training data, measured by EM, F1, and LJ metrics across four datasets.

on the same 10,000 preference data samples to ensure fair comparison.

As shown in Table 19, the results demonstrate that instruction-tuned models generally provide superior performance compared to base models for ReSearch agent guidance. The Qwen-2.5-7B-instruct model consistently outperforms its base counterpart across most evaluation metrics and datasets. Specifically, the instruct model achieves better performance on 10 out of 12 metric-dataset combinations, with particularly notable improvements on HotpotQA, 2WikiMultihopQA, and MuSiQue datasets.

The instruction-tuned model’s superior performance can be attributed to its enhanced ability to understand and follow reasoning evaluation instructions. Pre-training with instruction-following data enables the model to better align with the step-by-step reasoning evaluation requirements inherent in CoPRM training.

Overall, the results indicate that both base and instruction-tuned foundation models significantly outperform the Self-Consistency baseline when used for CoPRM training. However, instruction-tuned models demonstrate more consistent and robust performance across diverse reasoning tasks,

Training Samples	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			AVG		
	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
10,000	46.80	58.43	67.20	47.40	54.32	54.60	24.40	35.82	38.60	46.40	57.89	58.40	41.25	51.62	54.70
8,000	45.20	57.93	67.00	51.20	57.45	59.00	24.20	36.00	40.00	48.00	59.55	60.80	42.15	52.73	56.70
6,000	47.40	59.18	68.80	46.80	53.91	55.00	25.40	36.96	41.20	52.80	62.40	64.80	43.10	53.11	57.45
4,000	46.80	58.78	68.80	48.00	54.80	55.60	26.00	37.31	40.00	49.60	62.08	64.00	42.60	53.24	57.10
2,000	46.40	58.64	68.00	46.00	52.84	53.00	25.60	36.98	40.80	48.00	58.18	60.00	41.50	51.66	55.45
0	43.60	55.50	64.40	43.60	51.01	52.80	23.40	33.46	36.80	41.60	52.72	54.40	38.05	48.17	52.10

Table 17: Performance of ReSearch agent guided by Llama-GymPRM trained on varying amounts of training data, measured by EM, F1, and LJ metrics across four datasets.

Method	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			AVG		
	$k=3$	$k=5$	$k=10$	$k=3$	$k=5$	$k=10$	$k=3$	$k=5$	$k=10$	$k=3$	$k=5$	$k=10$	$k=3$	$k=5$	$k=10$
<i>EM Score</i>															
Self-Consistency	45.60	45.00	44.20	50.80	48.40	49.00	24.40	24.80	26.00	48.00	46.40	45.60	42.20	41.15	41.20
Llama-GymPRM	45.20	46.80	46.80	47.20	48.00	47.40	24.40	24.60	24.40	44.80	44.80	46.40	40.40	41.05	41.25
Llama-RePRM	42.60	42.80	41.60	47.40	47.00	44.20	23.80	23.00	22.20	44.80	44.80	48.80	39.65	39.40	39.20
Llama-CoPRM	47.20	46.60	49.00	48.00	47.60	51.80	25.80	27.00	26.00	47.20	50.40	52.80	42.05	42.90	44.90
Qwen-CoPRM	45.00	48.20	46.60	49.00	50.20	50.60	25.40	25.20	28.00	49.60	48.80	50.40	42.25	43.10	43.90
<i>F1 Score</i>															
Self-Consistency	57.70	57.61	56.71	57.86	53.93	55.14	35.45	35.66	37.92	60.83	58.02	56.45	52.96	51.31	51.56
Llama-GymPRM	57.57	58.39	58.43	54.24	53.79	54.32	36.33	36.60	35.82	55.49	56.18	57.89	50.91	51.24	51.62
Llama-RePRM	54.99	54.21	53.49	52.88	52.17	49.45	34.41	34.15	33.08	57.31	55.15	57.70	49.90	48.92	48.43
Llama-CoPRM	60.19	60.26	61.56	55.11	53.95	58.28	36.84	38.54	38.01	59.92	62.90	63.70	53.02	53.91	55.39
Qwen-CoPRM	58.68	59.86	59.70	54.82	54.98	56.65	36.53	35.54	38.80	62.56	59.04	60.11	53.15	52.36	53.82
<i>LJ Score</i>															
Self-Consistency	66.00	66.40	65.20	58.60	54.60	56.60	37.60	38.80	40.80	61.60	58.40	58.40	55.95	54.55	55.25
Llama-GymPRM	68.20	67.80	67.20	54.80	55.00	54.60	39.40	40.00	38.60	56.00	57.60	58.40	54.60	55.10	54.70
Llama-RePRM	63.40	63.40	62.00	54.60	54.40	50.40	38.00	35.60	36.00	58.40	57.60	60.80	53.60	52.75	52.30
Llama-CoPRM	70.40	69.60	72.20	57.00	55.00	59.40	41.20	42.00	41.40	61.60	64.00	66.40	57.55	57.65	59.85
Qwen-CoPRM	69.40	69.20	70.80	56.20	56.20	56.80	40.40	39.20	42.60	66.40	60.80	63.20	58.10	56.35	58.35

Table 18: Performance comparison of different reward models with varying numbers of candidate steps ($k=3, 5, 10$). CoPRM methods (highlighted) consistently outperform baseline methods across all sampling configurations, with performance generally improving as k increases.

achieving superior results on 10 out of 12 metric-dataset combinations, making them the preferred choice for CoPRM initialization.

E.4 Impact of Answer Similarity Evaluation Criteria in Monte Carlo Rollout Assessment

This experiment investigates the impact of using different answer similarity criteria in the Monte Carlo rollout evaluation (Eq. 4). We compare two criteria: (1) **F1 Threshold (Original)**: rollouts are successful if F1 score exceeds $\tau = 0.75$, providing partial credit; (2) **Exact Match (EM) Threshold**: rollouts are successful if EM score exceeds $\tau = 0.75$. Both methods use $N = 8$ rollouts per search step with ReSearch-Qwen-7B-Instruct.

Table 20 presents the results across four datasets for both Llama and Qwen architectures.

CoPRM trained with F1 threshold rollout evaluation consistently outperforms those trained with EM threshold rollout evaluation across both model

architectures. For Llama-CoPRM, using F1 threshold for rollout evaluation achieves average improvements of 0.2 EM points, 0.87 F1 points, and 1.35 LJ points. For Qwen-CoPRM, the F1 threshold approach yields average gains of 0.95 EM points, 0.99 F1 points, and 1.65 LJ points.

The stricter EM threshold produces more conservative rollout success assessments, leading to sparser positive training signals during preference data collection. This sparsity reduces the diversity of successful reasoning patterns captured in the training data, limiting CoPRM’s ability to learn nuanced quality distinctions. In contrast, the F1 threshold captures a broader spectrum of partially successful reasoning paths, enabling more robust step evaluation capabilities.

These findings demonstrate that overly stringent evaluation criteria can paradoxically reduce training signal quality by creating artificial sparsity in preference data. The success of F1 threshold rollout evaluation shows that incorporating partial

Model Type	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			AVG		
	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
Base	45.40	57.91	68.00	49.60	56.14	56.60	26.00	37.75	42.20	49.60	62.98	64.80	42.65	53.70	57.90
Instruct	46.60	59.70	70.80	50.60	56.65	56.80	28.00	38.80	42.60	50.40	60.11	63.20	43.90	53.82	58.35

Table 19: Performance of ReSearch agent guided by Qwen-CoPRM with different foundation model types (base vs. instruct), measured by EM, F1, and LJ metrics across four datasets.

Model	Criterion	HotpotQA			2WikiMultihopQA			MuSiQue			Bamboogle			Average		
		EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ	EM	F1	LJ
Llama	EM	48.6	60.59	71.0	52.0	57.67	59.8	27.0	38.21	40.8	51.2	61.6	62.4	44.7	54.52	58.5
	F1	49.0	61.56	72.2	51.8	58.28	59.4	26.0	38.01	41.4	52.8	63.7	66.4	44.9	55.39	59.85
Qwen	EM	47.6	59.58	68.6	48.0	54.5	55.0	28.2	38.15	41.6	48.0	59.09	61.6	42.95	52.83	56.7
	F1	46.6	59.7	70.8	50.6	56.65	56.8	28.0	38.8	42.6	50.4	60.11	63.2	43.9	53.82	58.35

Table 20: Performance of ReSearch agent guided by CoPRM trained on preference data collected using different rollout evaluation criteria, measured by EM, F1, and LJ metrics across four datasets.

credit mechanisms in rollout evaluation leads to more effective PRMs for guiding search agent reasoning.

F Data Construction Cost Analysis

We provide a quantitative analysis of the data construction cost for CoPRM compared to baseline PRMs. All CoPRM components utilize locally deployable open-source models (e.g., Qwen-7B, QwQ-32B) and are fully compatible with consumer-grade hardware such as the RTX 3090.

Data Construction Cost. Generating 10,000 training samples requires approximately 19.4 H100 GPU-hours in total, comprising 10.0 H100 GPU-hours for Stage 1 ranking and 9.4 H100 GPU-hours for parallelized Stage 2 batched rollouts. Stage 3 relies exclusively on CPU processing and incurs negligible overhead. In comparison, generating equivalent data volumes requires 15.9 H100 GPU-hours for GymPRM and 20.1 H100 GPU-hours for RePRM.

Amortized Efficiency and Data Quality. Although GymPRM requires less upfront compute, CoPRM provides superior efficiency across two practical dimensions:

- **Amortized Efficiency:** CoPRM’s construction serves as a scalable one-time investment. Trained on a single agent’s trajectories, CoPRM provides plug-and-play guidance to all five evaluated search agents without agent-specific retraining (Table 1), yielding an amortized cost of ~ 3.9 H100 GPU-hours per agent.
- **Data Utilization Efficiency:** As demonstrated in Figure 3, baseline PRMs plateau or even degrade

with increasing data volume, as scaling without quality control amplifies assessment noise. CoPRM trained on merely 2,000 samples already surpasses the 10k-sample performance of both GymPRM and RePRM, confirming that collaborative calibration concentrates resources on high-signal preference pairs rather than sheer volume.

G Baseline Details

This section introduces baseline models and datasets used in our evaluation to ensure full reproducibility of our experimental results.

G.1 Foundational Baselines

We first establish performance baselines using traditional approaches that rely solely on carefully designed prompts without specialized training. These baselines serve as lower bounds and help contextualize the value of search-based reasoning:

- **Naive Generation:** Direct answer generation without external search, serving as a lower bound baseline that relies solely on parametric knowledge.
- **Naive RAG:** Standard retrieval-augmented generation with single-step search and answer generation, representing the simplest form of retrieval-augmented reasoning.
- **Iter-RetGen** (Shao et al., 2023): An iterative retrieval and generation approach that alternates between search and reasoning steps, providing a more sophisticated retrieval strategy.
- **IRCoT** (Trivedi et al., 2023): Interleaving retrieval with chain-of-thought reasoning specifically designed for multi-hop question answering tasks.

Implementation Details: For all foundational baselines, we use the FLASHRAG (Jin et al., 2025c) implementation.

G.2 Search Agent Baselines.

Moving beyond foundational approaches, we evaluate CoPRM’s guidance effectiveness across search agents implemented through different training paradigms. This evaluation design demonstrates that our CoPRM can enhance reasoning regardless of how the underlying agents were trained:

Prompt-Engineering Based Agents. We include prompt-based agents that rely solely on carefully designed prompts without specialized training:

- **PE-ReSearch:** Our implementation of a prompt-engineering based search agent, using prompts consistent with ReSearch (Chen et al., 2025b) but without any fine-tuning, establishing a strong prompt-engineering baseline.

Implementation Details: This baseline utilizes Qwen2.5-7B-Instruct as the foundation model and relies solely on the ReSearch system prompt (detailed in Section M) without any parameter updates.

Supervised Fine-Tuning (SFT) Based Agents. To represent learning from expert demonstrations, we include agents trained on successful reasoning trajectories:

- **RS-ReSearch:** This agent is created by fine-tuning the base model exclusively on successful reasoning paths, which are selected via rejection sampling from the data generated in our initial filtering stage (as described in Section H.1).

Implementation Details: Specifically, from all reasoning trajectories generated during the question filtering rollout process, we apply rejection sampling to retain only trajectories whose final answers achieve F1 scores greater than 0.5 when compared to ground truth answers. After trajectory deduplication, we mask the content between <result> and </result> tags since these segments represent external retrieval environment feedback rather than model-generated content. The resulting curated trajectories are used to conduct supervised fine-tuning on Qwen2.5-7B-Instruct, creating an agent trained exclusively on successful reasoning demonstrations.

Reinforcement Learning (RL) Based Agents.

We evaluate against state-of-the-art RL-trained search agents that learn search strategies through environmental interaction, representing the best performing search agent approaches in current research:

- **ReSearch (Chen et al., 2025b):** A reinforcement learning approach that treats search operations as integral components of chain-like reasoning processes, training models from scratch to learn interactive reasoning-search patterns using GRPO without supervised reasoning data.
- **R1-Searcher (Song et al., 2025):** A two-stage RL framework that first uses retrieve-reward to teach correct retrieval invocation format, then applies answer-reward to learn effective knowledge utilization, enabling autonomous exploration without distillation or cold-start training.
- **Search-R1 (Jin et al., 2025b):** A training framework that models search engines as environmental components with retrieved token masking for optimization stability, supporting multi-turn interleaved reasoning through structured token formatting.

Implementation Details: For ReSearch, R1-Searcher, and Search-R1, we use the same prompts as their original work, with detailed prompts provided in Section M. For ReSearch, we use the model trained with reinforcement learning on Qwen2.5-7B-Instruct base model from their implementation. For R1-Searcher and Search-R1, we use the models trained with reinforcement learning on Qwen2.5-7B-base model from their respective implementations.

G.3 Inference-Time Guidance Baselines

Finally, to directly evaluate CoPRM’s effectiveness as an inference-time guidance mechanism, we compare against existing inference-time guidance methods. This comparison isolates the contribution of our CoPRM guidance from the underlying agent capabilities:

- **Greedy Decoding:** Selects the highest-probability action at each step, representing the default inference strategy without any guidance mechanism.
- **Self-Consistency (SC) (Wang et al., 2023):** Generates multiple candidate steps and selects the

most frequent action at each step, providing a strong ensemble-based baseline for comparative evaluation.

- **GymPRM** (Xiong et al., 2026): An LLM-based process reward modeling approach that trains a critic r_ϕ to evaluate state-action pairs, distinguishing preferred from less-preferred intermediate actions.
- **RePRM** (Sun et al., 2025c): An MC-based process reward modeling approach that utilizes Monte Carlo estimation to derive supervision signals for training the reward model.

GymPRM Implementation Details: To ensure fair comparison with our CoPRM, we implement GymPRM using QwQ-32B with identical ranking prompts (as detailed in Section M.1) for state evaluation and preference pair generation. Following the original authors’ implementation, we train the GymPRM model using preference pairs derived from these rankings, maintaining experimental consistency by utilizing 10,000 preference pairs to match the training dataset size used for our CoPRM model.

RePRM Implementation Details: We implement RePRM using the source code provided by the authors. To ensure a fair comparison, we use the same ReSearch-Qwen-7B-Instruct as CoPRM to serve as the trajectory generator. Following the MC-based data collection algorithm described in their paper, we generate 10,000 PRM training data samples to train the model.

This comprehensive baseline design enables us to evaluate CoPRM framework across multiple dimensions: its effectiveness compared to traditional QA approaches (foundational baselines), its generalizability across agents trained through different paradigms (search agent baselines), and its direct superiority over existing inference-time guidance methods (inference-time guidance baselines).

H Details of Training Data Collection

This section provides additional details of our Collaborative Calibration methodology described in Section 4.1.

H.1 Question Filtering for Quality Assurance

To ensure that preference pairs exhibit clear and substantive quality differences between positive

and negative samples, we begin with a careful question filtering process. Starting from 2,000 questions in HotpotQA (Yang et al., 2018), we employ ReSearch-Qwen-7B-Instruct (Chen et al., 2025b) with the system prompt detailed in Section M to perform 8 rollouts per question. Each rollout generates a complete problem-solving trajectory leading to a final answer.

We then filter questions based on answer diversity: questions where all 8 rollouts produce either entirely correct or entirely incorrect final answers are removed, as they fail to provide meaningful quality distinctions for preference learning. The correctness of final answers is determined by computing the F1 score between the predicted answer and the ground truth answer, with answers achieving F1 scores greater than 0.5 considered correct. This filtering retains 654 questions that demonstrate mixed success rates during the sampling process, ensuring that the subsequent preference data collection occurs over questions with meaningful quality variations.

H.2 State Generation and Immediate Quality Assessment

At each time step t , we use ReSearch-Qwen-7B-Instruct with temperature=1.0 to generate $k = 10$ candidate steps. After deduplication to remove redundant candidates, all unique candidate steps are evaluated using our immediate quality assessment framework. The highest-ranked candidate is selected for state advancement.

Our immediate quality assessment employs QwQ-32B (Qwen Team, 2025b) as an LLM ranker to evaluate candidate steps based on reasoning soundness and action appropriateness. Rather than absolute scoring, we use comparative ranking because it enables more effective assessment of relative quality differences between candidates (Liu et al., 2024).

The ranking process evaluates each candidate step across three key criteria: (1) **Reasonableness and Sufficiency of Thought:** Whether the reasoning process demonstrates logical coherence and adequate analysis of available information; (2) **Necessity, Utility, and Accuracy of Action:** Whether the proposed action (search query or answer) is well-motivated, useful for progress, and accurately formulated; (3) **Redundancy Check:** Whether the action avoids unnecessary repetition of previous search efforts. The detailed prompt for this ranking process is provided in Section M.1.

To ensure computational efficiency and focus our outcome-informed evaluation on questions with meaningful learning potential, we implement an additional filtering step after the immediate quality assessment stage. For each question, after generating and ranking all states through the ranker-guided iterative process, we examine the quality distribution of answer candidates across all generated states.

Specifically, if all answer candidates p_t^a generated across all states for a given question yield F1 scores below the threshold τ , we exclude this question from subsequent Monte Carlo rollout evaluation. This pre-rollout filtering mechanism serves two key purposes: (1) **Computational Efficiency**: It eliminates expensive rollout computations for questions where the search agent consistently fails to generate viable answer candidates, and (2) **Learning Quality**: For such questions, all candidate steps would likely yield similar Monte Carlo estimates near zero, making quality distinctions between steps indiscernible and thus providing limited learning signal for preference modeling.

H.3 Example Preference Pair

The following example illustrates a typical preference pair from our collected dataset:

Example Preference Pair

Question: What is the broadcasting location of the program on which the music video for the song "Out of the Woods" premiered?

Previous State Context (s_t):

```
<think>
To find the broadcasting location of the program on which the music video for "Out of the Woods" premiered, I will first search for information about the music video's premiere.
</think>
<search>
Which program premiered the music video for "Out of the Woods"?
</search>
<result>
"Out of the Woods" serves as the sixth official single with an accompanying music video that premiered on December 31, 2015, during "Dick Clark's New Year's Rockin' Eve" on ABC... [additional search results omitted for brevity]
</result>
```

Preferred Step (p_t^+):

```
<think>
After analyzing the information from the sources, it is clear that the relevant program in question is "Dick Clark's New Year's Rockin' Eve" on ABC which premiered the music video for the song "Out of the Woods". Now I will look for the location where this program is broadcasted to finalize the question.
```

```
</think>
<search>
Broadcasting location of "Dick Clark's New Year's Rockin' Eve"
</search>
```

Less Preferred Step (p_t^-):

```
<think>
I've gathered information on the program that aired the music video for "Out of the Woods" as it premiered on December 31, 2015. It was during "Dick Clark's New Year's Rockin' Eve" on ABC. I now know the broadcasting location of the program which aired the music video.
</think>
<search>
On which channel was "Dick Clark's New Year's Rockin' Eve" aired?
</search>
```

In this example, the preferred step demonstrates more focused reasoning toward the specific question about broadcasting location and formulates a more targeted search query, while the less preferred step shows redundant information gathering that has already been established.

H.4 Data Construction Pipeline Algorithm

This section provides the detailed algorithmic description of our CoPRM training data construction pipeline. The complete algorithm is presented in Algorithm 1.

I Implementation Details

This section provides comprehensive technical implementation details for our CoPRM framework, including training configurations, inference settings, retrieval configurations, evaluation metrics, relative improvement calculation methodology, computational infrastructure, and detailed prompts used throughout the implementation, ensuring reproducibility and transparency of our methodology.

AI Use Statement. Generative AI tools were used solely for coding assistance and language polishing.

I.1 Training Configuration

The detailed hyperparameters for CoPRM training are summarized in Table 21. All PRM models are trained for one epoch using the specified hyperparameters.

I.2 Inference Configuration

During PRM-guided inference, we generate 10 candidate steps at each reasoning step with temperature=1.0. The CoPRM selects the highest-scoring candidate according to our collaborative consensus

Hyperparameters	Value
Learning rate	2×10^{-6}
LR scheduler	Cosine
Weight decay	0.001
Gradient clip	1.0
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.999$)
Warm-up ratio	0.01
Training data quantity	10K
Batch size	16
Sequence length	4096
Mixed precision training	BF16
Training epochs	1

Table 21: Detailed hyperparameters of our CoPRM training.

framework for step-by-step guidance throughout the reasoning process.

I.3 Retrieval Configuration

Following ReSearch methodology, our retrieval environment employs the FLASHRAG toolkit (Jin et al., 2025c) with E5-base-v2 (Wang et al., 2022) as the dense retriever and Wikipedia (December 2018 version) (Karpukhin et al., 2020) as the knowledge base. All the corpus indexing and embedding has been preprocessed by FlashRAG. For BrowseComp-Plus, we use BM25 retrieval over its fixed corpus.

I.4 Evaluation Metrics

Exact Match (EM) Binary accuracy metric measuring whether the predicted answer exactly matches the ground truth answer after normalization.

F1 Score Token-level F1 score measuring the overlap between predicted and ground truth answers using precision and recall.

LLM-as-a-Judge (LJ) Semantic evaluation using Qwen2.5-72B-Instruct (Yang et al., 2024) to assess answer correctness through natural language understanding, capturing semantic equivalence beyond exact string matching. The evaluation follows the prompt template detailed in Section M, which instructs the judge to determine correctness based on semantic similarity rather than exact textual match.

BrowseComp-Plus Evaluation. Following the benchmark protocol, we employ Qwen3-32B (Qwen Team, 2025a) as a judge to evaluate the correctness of final answers and report answer accuracy and evidence recall.

I.5 Relative Improvement Calculation

All relative improvements reported in the main paper are computed as $I_{avg} = \frac{1}{3}(I_{EM} + I_{F1} + I_{LJ})$, where each $I_m = (C_m - B_m)/B_m \times 100\%$ compares CoPRM’s average performance C_m against the strongest existing baseline B_m for each agent across all four datasets. Table 22 summarizes these results.

Search Agent	Average Imp.	Strongest Baseline
<i>Llama-based PRMs</i>		
PE-ReSearch	9.6%	Self-Consistency
ReSearch	8.2%	Self-Consistency
Search-R1	5.1%	GymPRM
R1-Searcher	3.1%	GymPRM
RS-ReSearch	3.5%	Greedy
<i>Qwen-based PRMs</i>		
PE-ReSearch	3.7%	Self-Consistency
ReSearch	4.3%	GymPRM
Search-R1	2.5%	GymPRM
R1-Searcher	4.2%	RePRM
RS-ReSearch	2.0%	Greedy

Table 22: Summary of relative improvements and strongest baselines for each search agent.

I.6 Computing Infrastructure and Software Environment

All experiments were conducted on a server equipped with dual Intel Xeon Platinum 8558 CPUs, 2TB system RAM, and $4 \times$ NVIDIA H100 80GB GPUs running Ubuntu 22.04.5 LTS. The experimental framework utilizes the following libraries: torch=2.5.1, CUDA version=12.4, transformers=4.49.0, accelerate=1.5.2, trl=0.13.0, vllm=0.6.5, datasets=3.3.2, faiss-gpu=1.8.0, deepspeed=0.16.3.

J CoPRM-guided inference Algorithm

This section provides the detailed algorithmic description of our CoPRM-guided inference process, complementing the methodology presented in the main paper. The complete algorithm is presented in Algorithm 2.

K Inference Efficiency Analysis

To address concerns about the computational overhead of CoPRM-guided inference, we provide a detailed analysis of absolute inference time metrics. Our experiments were conducted under the following configuration:

Hardware and Software Configuration.

- **GPU:** Single NVIDIA H100 (80GB)
- **Inference Framework:** vLLM 0.6.5
- **Policy Model:** ReSearch-Qwen-7B-Instruct
- **Reward Model:** Llama-CoPRM (Llama-3.1-8B-Instruct based)
- **Evaluation Dataset:** 500 HotpotQA samples

Absolute Time Metrics. Table 23 presents the detailed breakdown of inference time components.

Metric	Value
Total Inference Time	2723.34 seconds (45.39 minutes)
Average Time per Sample	5.447 seconds
Reward Model Total Time	562.38 seconds
Reward Model Time per Sample	1.125 seconds
Reward Model Time Ratio	20.65%
Search Engine Total Time	53.10 seconds
Search Engine Time per Sample	0.106 seconds
Search Engine Time Ratio	1.95%

Table 23: Absolute inference time metrics for CoPRM-guided inference on 500 HotpotQA samples using a single H100 GPU.

Practical Deployment Considerations. Our analysis reveals several important findings for practical deployment:

1. **Acceptable Absolute Latency:** The average processing time of ~ 5.45 seconds per sample is practical for many real-world applications, particularly in knowledge-intensive QA scenarios where users expect thoughtful, well-researched responses.
2. **Modest Overhead:** The reward model contributes only 20.65% of the total inference time, and search engine latency is negligible (1.95%), indicating that the majority of computational cost comes from the policy model’s generation of candidate steps.

Potential Optimizations. The current implementation can be further optimized through several approaches:

- **Parallel Candidate Generation:** Generating and scoring multiple candidates in parallel can reduce wall-clock time.

- **Reward Model Caching:** Caching reward scores for similar reasoning states can eliminate redundant computations.

These optimizations represent promising directions for future work to further enhance the practical efficiency of PRM-guided inference.

L Additional Related Work

In this section, we expand on the related works discussed in Section 2. Our work is closely related to two specific categories of research: Process Reward Models (PRMs) and PRM-guided test-time scaling. Below, we discuss each category and position our work within the broader landscape.

L.1 Process Reward Models

Process Reward Models (PRMs) have emerged as a pivotal approach to enhancing LLM reasoning through step-wise supervision (Uesato et al., 2022). Initially validated in mathematical reasoning, PRMs provide granular feedback to verify intermediate logic steps (Lightman et al., 2024; Wang et al., 2024; Sun et al., 2025a). To mitigate the cost of human annotation, recent advancements have transitioned toward automated supervision, employing methods such as Monte Carlo estimation (Wang et al., 2024; Luo et al., 2024) or LLM-as-a-Judge (Zhang et al., 2025a; Gao et al., 2025a) to synthesize process labels.

Beyond mathematics, the application of PRMs has recently expanded into general agentic tasks. Notable works include Web-Shepherd (Chae et al., 2025), which targets web navigation, and Agent-PRM (Xi et al., 2026), which introduces “Promise” and “Progress” metrics for decision-making benchmarks. Similarly, (Choudhury, 2025) validate process rewards on embodied tasks. However, applying these frameworks to search agents presents fundamental ontological and evaluative mismatches.

Existing agent PRMs primarily address tasks governed by grounded actions intended to modify the physical or environmental state (e.g., “buy an item”, “move to a grid”, or “craft a tool”). In these scenarios, rewards typically quantify the physical proximity to a concrete goal state. In contrast, search agents perform **epistemic actions** aimed at altering the agent’s *internal knowledge state* without necessarily changing the external environment. Standard agent metrics like “Progress” fail to capture the nuances of *information gain* or resolve the critical epistemic dilemma unique to

Aspect	Zhang et al. (2025d)	CoPRM	Implication
Step type	Homogeneous deduction	Heterogeneous (Search/Answer)	Cross-type decisions are unavoidable
Supervision	Binary correctness	Calibrated preferences for strategic utility	Valid-but-redundant steps require finer granularity
Cross-type	N/A	Outcome-informed gating (Eq. 6)	Prevents premature answering or redundant search
Data	Trajectory-level generation, then post-hoc evaluation	Rank-guided iterative step generation	Produces relative supervision aligned with step selection
Inference	Post-hoc (BoN)	Online step-wise (Alg. 2)	Avoids error propagation in early search steps

Table 24: Key differences between Math PRMs (specifically Zhang et al. (2025d)) and CoPRM, with implications for search agents.

search: the trade-off between acquiring more information (Search) and utilizing current knowledge (Answer). Our Collaborative Calibration methodology is specifically designed to bridge this gap by measuring sufficiency rather than physical proximity.

Furthermore, CoPRM differs fundamentally from Math PRMs in both supervision target and methodological design. Recent work such as Zhang et al. (2025d) has proposed consensus filtering that combines LLM-as-a-judge with MC estimation to construct process reward data for mathematical reasoning. While CoPRM shares the high-level principle of synthesizing complementary signals, our contribution addresses the distinct structural challenges of search agents. Table 24 summarizes the key methodological differences and their implications.

These search-specific designs yield distinctive empirical advantages: CoPRM trained on a single agent’s trajectories from only HotpotQA generalizes to guide diverse search agents across unseen datasets without retraining (Table 1) and remains robust across backbone families (Table 5). Ablation studies confirm the necessity of the two core components of collaborative calibration: same-type consensus filtering (Table 3) and cross-type outcome-informed gating (Table 4).

L.2 PRM Guided Test-Time Scaling

PRMs are widely utilized for test-time scaling to enhance inference performance without retraining the base model. The most prevalent strategy is Best-of-N (BoN), where the model generates

N complete solutions, and the PRM ranks them post-hoc to select the optimal trajectory (Lightman et al., 2024). To achieve more granular control, other approaches integrate PRMs directly into tree search algorithms—such as Monte Carlo Tree Search (MCTS) (Guan et al., 2025), beam search (Cinquin et al., 2025), or greedy search (Zhang et al., 2025b). In these paradigms, partial solutions are expanded at each decoding step, and the PRM evaluates the utility of the new step to guide subsequent exploration (Snell et al., 2025).

However, heavy scaling strategies like BoN and tree search face critical limitations when applied to search agents. The standard BoN approach suffers from what we term “full trajectory generation lag,” as it necessitates generating complete trajectories prior to evaluation. In a search context, this requires the agent to perform all retrieval interactions and process the accumulated long-context information for every candidate trajectory, even if the initial steps are fundamentally flawed. This results in substantial computational overhead and latency, rendering it impractical for real-time deployment. Similarly, while tree search algorithms such as MCTS and beam search enable intermediate evaluation, they often impose prohibitive computational costs. MCTS requires extensive rollouts at every node, while beam search necessitates maintaining multiple active candidate paths (beams) throughout the generation process. For search agents, where each step involves retrieving and processing external knowledge, these approaches significantly multiply the computational burden of inference and retrieval, rendering them inefficient for real-time deployment.

To address these challenges, our work implements a PRM-guided step-wise greedy search strategy that balances exploration with computational efficiency. This lightweight and effective decoding strategy allows us to apply CoPRM across diverse search agents, ensuring a fair and scalable comparison of guidance effectiveness.

M Prompt Details

This section provides detailed prompts used in our CoPRM framework implementation.

M.1 LLM Ranking Prompt for Immediate Quality Assessment

The following prompt is used for LLM-based ranking of candidate steps in the immediate quality

assessment.

System and User Prompts for LLM Ranking in Immediate Quality Assessment

System Prompt:

You are a decision evaluation assistant. Your task is to rank the proposed next actions for an AI agent that solves problems step-by-step using a Wikipedia search tool. The agent's workflow is: Think → Search → Get Results → Think Again → Decide Next Action (new search or provide final answer). A "proposed next action" includes the agent's thought process (<think>...</think>) and the specific action it proposes based on that thought (<search>...</search> or <answer>...</answer>). You will be given: the original question, the existing search result history, and a list of one or more "proposed next actions". Based on this information, please rank the proposed actions from most to least appropriate.

User Prompt:

Original Question:
{question}

Existing Search Result History:
{curr_history}

List of Proposed Next Actions:
{actions_text_with_think}

Important Assumptions

1. The agent has no prior knowledge of the subject matter and must rely solely on the "Existing Search Result History" to evaluate and answer the "Original Question".
2. The "Existing Search Result History" is the objective information obtained by the agent.
3. The "<think>" part in a "proposed next action" shows the agent's analysis and inference process before proposing a specific search or answer. This part may contain misunderstandings or insufficient use of historical information.

Appropriateness Evaluation Criteria

1. Check for Reasonableness and Sufficiency of Thought
2. Check for Necessity, Utility, and Accuracy of Action
3. Redundancy Check
4. ... (detailed criteria omitted for brevity)

Expected Output Format

- Output the indices of the ranked actions in JSON format:
“json{“ranked_indices”: [list of indices]}“
For example, a valid output is:
“json{“ranked_indices”: [0, 1, 2]}“
Another example: “json{“ranked_indices”: [3, 0, 2, 1]}“
Your entire output must be precisely this JSON format and nothing else.
- Rank the actions from most to least appropriate according to the evaluation criteria above.
- Do not provide additional explanations or justifications.

M.2 LLM-as-a-Judge Evaluation Prompt

The prompt used for semantic evaluation (LJ metric) in our experimental evaluation, as described in the Experiments section.

Prompt for LLM-as-a-Judge Semantic Evaluation

You will be given a question and its ground truth answer list where each item can be a ground truth answer. Provided a pred_answer, you need to judge if the pred_answer correctly answers the question based on the ground truth answer list. You should first give your rationale for the judgement, and then give your judgement result (i.e., correct or incorrect).

Here is the criteria for the judgement:

1. The pred_answer doesn't need to be exactly the same as any of the ground truth answers, but should be semantically same for the question.
2. Each item in the ground truth answer list can be viewed as a ground truth answer for the question, and the pred_answer should be semantically same to at least one of them.

question: {question}
ground truth answers: {gt_answer}
pred_answer: {pred_answer}

The output should in the following json format:

```
“json
{
  "rationale": "your rationale for the judgement,
as a text",
  "judgement": "your judgement result, can only
be 'correct' or 'incorrect'"
}
“
```

Your output:

M.3 Search Agent Reasoning Prompts

The system prompts used by different search agents in our evaluation framework.

ReSearch Agent System Prompt

You are a helpful assistant that can solve the given question step by step with the help of the wikipedia search tool. Given a question, you need to first think about the reasoning process in the mind and then provide the answer. During thinking, you can invoke the wikipedia search tool to search for fact information about specific topics if needed. The reasoning process and answer are enclosed within <think> </think> and <answer> </answer> tags respectively, and the search query and result are enclosed within <search> </search> and <result> </result> tags respectively. For example, <think> This is the reasoning process. </think> <search> search query here </search> <result> search result here </result> <think> This is the

reasoning process. `</think>` `<answer>` The final answer is `\boxed{answer here}` `</answer>`. In the last part of the answer, the final exact answer is enclosed within `\boxed{}` with latex format.

Search-R1 Agent System 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}}`

R1-Searcher Agent System Prompt

The User asks a question, and the Assistant solves it. The Assistant first thinks about the reasoning process in the mind and then provides the User with the final answer. The output format of reasoning process and final answer are enclosed within `<think>` `</think>` and `<answer>` `</answer>` tags, respectively, i.e., "`<think>` reasoning process here `</think>` `<answer>` final answer here `</answer>`". During the thinking process, the Assistant can perform searching for uncertain knowledge if necessary with the format of "`<|begin_of_query|>` search query (only keywords) here `<|end_of_query|>`". A query must involve only a single triple. Then, the system will provide the Assistant with helpful information with the format of "`<|begin_of_documents|>` ...search results... `<|end_of_documents|>`".

User: `{{question}}`
Assistant: `<think>`

N Notation and Symbols

This section provides a comprehensive reference of all symbols and notations used throughout the paper. Table 25 summarizes the key symbols organized by category.

O Case Study Analysis

This section presents detailed case study analyses to empirically validate the limitations discussed in Section 1: assessment unreliability in existing PRMs. Through a systematic comparison of CoPRM against these baselines, we illustrate how CoPRM mitigates these biases to provide precise strategic guidance.

O.1 Case Study 1: Mitigating Speculative Selection and Hallucination in Historical Reasoning

Historical multi-hop reasoning requires maintaining a rigorous genealogical chain. Table 26 compares the reasoning trajectories for the question: *"Who is Gaius Rubellius Blandus's father-in-law?"*

- **CoPRM:** CoPRM correctly identifies the logical dependency: to find a father-in-law, one must first identify the spouse. It selects the step targeting the marriage relationship (*"who was gaius rubellius blandus married to?"*), successfully bridging the knowledge gap.
- **RePRM:** In Step 2, RePRM gets distracted by the prominent entity "Tiberius" found in the retrieved context ("granddaughter of Tiberius"), ignoring the true search target (the spouse's father). Consequently, it attempts a speculative shortcut: it speculatively assumes the answer is directly related to this famous figure, thus querying "Tiberius's father" and skipping the necessary logical step of identifying Julia Livia's father first.
- **GymPRM:** GymPRM suffers from specious selection. It performs valid retrieval steps, correctly identifying the spouse (Julia Livia) and querying her father. However, despite retrieving the correct answer ("daughter of Drusus Julius Caesar"), it exhibits specious hallucination driven by plausibility bias in the final reasoning. It mimics the behavior of selecting a superficially plausible prominent entity in the context ("Tiberius") as the father-in-law, ignoring the retrieved evidence.

O.2 Case Study 2: Overcoming Distractors in Compositional Reasoning

Complex questions often contain "distractor" entities that can trigger strategic derailment. Table 27 illustrates a scenario requiring query decomposition: *"Who is the child of the Italian navigator... where Christian Sánchez was born?"*

- **CoPRM:** CoPRM employs strategic simplification, correctly prioritizing the main subject (the navigator) while ignoring the nested distractor until necessary.
- **RePRM:** RePRM demonstrates speculative selection. It formulates an under-specified query by omitting the "sailed for England" constraint.

Instead of resolving this ambiguity, it bets on a "prominent entity" path, asking "who is the son of amerigo vespucci?", leading to the incorrect answer "lorenzo vespucci".

- **GymPRM:** GymPRM exhibits specious selection. It effectively resolves the distractor ("El Salvador") but then gets distracted by this specious local detail. Instead of adhering to the global constraint ("sailed for England"), it incorrectly narrows the scope to "Central America", leading to an inefficient path and an erroneous conclusion.

Symbol	Description
<i>State and Action Space</i>	
Q	The initial query or question.
H_t	The history of prior thoughts, searches, and retrieved results up to time t .
$\mathcal{S}$	The state space of the search agent.
s_t	The state at time step t , represented as a tuple (Q, H_t) .
$\mathcal{A}$	The action space, containing search and answer actions.
a_t	The action at time step t , either a search action or an answer action.
<i>Process Units (Steps)</i>	
$\mathcal{P}$	The space of process units (steps).
p_t	A process unit at time step t , represented as a tuple (T_t, a_t) .
p_t^i	The i -th candidate step at time step t .
T_t	The reasoning (think) component of a process unit at time step t .
P_t	The candidate step set at time step t , $P_t = \{p_t^1, \dots, p_t^k\}$.
P_t^S	The subset of search steps in P_t .
P_t^A	The subset of answer steps in P_t .
p_t^+	The preferred (positive) step in a preference pair.
p_t^-	The less preferred (negative) step in a preference pair.
p_t^*	The optimal step selected by CoPRM at time step t .
<i>Preference Functions</i>	
$G(p_t^i, p_t^j)$ or G_{ij}	Immediate quality preference from LLM ranking. Returns $\{+1, -1, 0\}$.
$O(p_t^i, p_t^j)$ or O_{ij}	Outcome-informed preference. Returns $\{+1, -1, 0\}$.
O_{ij}^{same}	Outcome-informed preference for same-type step pairs.
O_{ij}^{mixed}	Outcome-informed preference for cross-type (search vs. answer) pairs.
Pref_{ij}	Final preference after collaborative consensus. Returns $\{+1, -1, 0\}$.
<i>Evaluation Metrics and Functions</i>	
$V(s_t, p_t^s)$	Monte Carlo estimated value of a search step p_t^s from state s_t .
$\text{F1}(p_t^a, \text{ans}^*)$	F1 score between the answer in step p_t^a and ground truth ans^* .
ans^*	The ground truth answer.
ans_j	The final answer from rollout j .
<i>Model and Training</i>	
r_ϕ	The CoPRM reward model parameterized by ϕ .
D	The preference dataset, $D = \{(s_t, p_t^+, p_t^-)\}$.
$L_{PRM}(\phi)$	The pairwise ranking loss for PRM training.
<i>Hyperparameters</i>	
k	Number of candidate steps generated per time step.
N	Number of Monte Carlo rollouts for search step evaluation.
τ	F1 threshold for determining rollout success.
θ	Gating threshold for cross-type preference determination.
$T_{\max}$	Maximum number of iterations allowed during inference.
<i>Condition Notations</i>	
$\mathcal{C}_{ij}^S$	Condition that both p_t^i and p_t^j are search steps: $(p_t^i, p_t^j \in P_t^S)$.
$\mathcal{C}_{ij}^A$	Condition that both p_t^i and p_t^j are answer steps: $(p_t^i, p_t^j \in P_t^A)$.
<i>Operators and Functions</i>	
$\oplus$	Concatenation operator for updating history or state.
$\text{sign}(\cdot)$	Sign function returning $\{-1, 0, +1\}$.
$\mathbf{1}_{[\cdot]}$	Indicator function (1 if condition holds, 0 otherwise).
$\sigma(\cdot)$	Sigmoid function.

Table 25: Summary of notation and symbols used throughout this paper.

Algorithm 1 CoPRM Training Data Construction Pipeline

Input: Questions Q , candidates k , MC rollouts N , thresholds τ, θ

Output: Preference dataset $D = \{(s_t, p_t^+, p_t^-)\}$

```

1: Question Filtering
2:  $D \leftarrow \emptyset, \mathcal{B} \leftarrow \emptyset, Q' \leftarrow \emptyset$ 
3: for each  $Q \in \mathcal{Q}$  do
4:   Generate complete trajectories for  $Q$ 
5:   if their final answers include successes and failures then
6:      $Q' \leftarrow Q' \cup \{Q\}$ 
7:   end if
8: end for
9: Stage 1: State Generation and LLM Ranking
10: for each  $Q \in Q'$  do
11:    $s_0 \leftarrow (Q, \emptyset), \mathcal{B}_Q \leftarrow \emptyset$ 
12:   while  $s_t$  is not terminal do
13:      $\tilde{P}_t \leftarrow \text{GENERATE}(s_t, k)$ 
14:      $P_t \leftarrow \text{DEDUPLICATE}(\tilde{P}_t)$ 
15:     Rank  $P_t$  to obtain  $G_{ij} = G(p_t^i, p_t^j)$ 
16:      $\mathcal{B}_Q \leftarrow \mathcal{B}_Q \cup \{(s_t, P_t, G)\}$ 
17:      $s_{t+1} \leftarrow \text{ADVANCE}(s_t, \text{TOP}(P_t, G))$ 
18:   end while
19:   if  $\text{HASVIABLEANSWER}(\mathcal{B}_Q, \tau)$  then
20:      $\mathcal{B} \leftarrow \mathcal{B} \cup \mathcal{B}_Q$ 
21:   end if
22: end for
23: Stage 2: Outcome-informed Evaluation
24:  $\mathcal{E} \leftarrow \emptyset$ 
25: for each  $(s_t, P_t, G) \in \mathcal{B}$  do
26:   Estimate  $V(s_t, p_t^s)$  for all  $p_t^s \in P_t^S$  by Eq. 4
27:   Evaluate  $\text{F1}(p_t^a, \text{ans}^*)$  for all  $p_t^a \in P_t^A$ 
28:   Compute  $O_{ij}^{\text{same}}$  and  $O_{ij}^{\text{mixed}}$  by Eqs. 5–6
29:    $\mathcal{E} \leftarrow \mathcal{E} \cup \{(s_t, P_t, G, O)\}$ 
30: end for
31: Stage 3: Collaborative Preference Construction
32: for each  $(s_t, P_t, G, O) \in \mathcal{E}$  do
33:   for each pair  $(p_t^i, p_t^j)$ , with search first if mixed do
34:     if  $C_{ij}^S$  or  $C_{ij}^A$  then
35:        $\text{Pref}_{ij} \leftarrow G_{ij}$  if  $G_{ij} = O_{ij}^{\text{same}}$ , else 0
36:     else
37:        $\text{Pref}_{ij} \leftarrow O_{ij}^{\text{mixed}}$ 
38:     end if
39:     if  $\text{Pref}_{ij} = +1$  then
40:        $D \leftarrow D \cup \{(s_t, p_t^i, p_t^j)\}$ 
41:     else if  $\text{Pref}_{ij} = -1$  then
42:        $D \leftarrow D \cup \{(s_t, p_t^j, p_t^i)\}$ 
43:     end if
44:   end for
45: end for
46: return  $D$ 

```

Algorithm 2 CoPRM-Guided Search Agent Inference

Input: Initial question Q , Search engine SE , Trained CoPRM r_ϕ , Max iterations T_{max} , Number of candidates k

Output: Final answer A_{final}

```

1: Initialize history  $H \leftarrow \text{InitialContext}(Q)$ 
2: Initialize state  $s_0 \leftarrow (Q, H)$ 
3:  $t \leftarrow 0$ 
4: while  $t < T_{max}$  do
5:   Generate diverse candidate steps
6:   Generate candidate set  $P_t = \{p_t^i\}_{i=1}^k$ , where  $p_t^i = (T_t^i, a_t^i)$ 
7:   if  $\text{IsEmpty}(P_t)$  then
8:      $A_{final} \leftarrow \text{"No solution found"}$ 
9:     return  $A_{final}$ 
10:  end if
11:  CoPRM scoring and selection
12:  Compute reward scores for each candidate:
13:     $\text{scores} \leftarrow \{r_\phi(s_t, p) : p \in P_t\}$ 
14:  Select optimal candidate:
15:     $p_t^* \leftarrow \arg \max_{p \in P_t} r_\phi(s_t, p)$ 
16:  Execute step and update state
17:  Update history:  $H \leftarrow H \oplus T_t^*$ 
18:  if  $p_t^*$  contains search action  $a_t^{s,*}$  then
19:    Execute search:  $SR \leftarrow SE(a_t^{s,*})$ 
20:    Update history:  $H \leftarrow H \oplus a_t^{s,*} \oplus SR$ 
21:    Update state:  $s_{t+1} \leftarrow (Q, H)$ 
22:     $t \leftarrow t + 1$ 
23:  else if  $p_t^*$  contains answer action  $a_t^{a,*}$  then
24:    Extract final answer:  $A_{final} \leftarrow a_t^{a,*}$ 
25:    return  $A_{final}$ 
26:  end if
27: end while
28:  $A_{final} \leftarrow \text{"Maximum iterations reached"}$ 
29: return  $A_{final}$ 

```

Question: Who is Gaius Rubellius Blandus’s father-in-law? Ground Truth: Drusus Julius Caesar
Step 1: Initial Strategic Approach CoPRM <i>Selected Query:</i> “who was gaius rubellius blandus married to?” <i>Strategic Focus:</i> Marriage relationship (directly targets the connection needed for father-in-law identification)
GymPRM <i>Selected Query:</i> “who was gaius rubellius blandus wife?” <i>Strategic Focus:</i> Correctly identifies the need to find the spouse.
RePRM <i>Selected Query:</i> “who was gaius rubellius blandus and did he have a spouse?” <i>Strategic Focus:</i> Correctly identifies the need to find the spouse.
Step 2: Information Processing and Strategic Focus CoPRM <i>Retrieved Information:</i> “...In AD 33, he married Julia Livia, granddaughter of the Roman emperor Tiberius...” <i>Selected Query:</i> “who was the father of julia livia, granddaughter of emperor tiberius?” <i>Strategic Focus:</i> Correct lineage (targets Julia Livia’s father, the actual father-in-law)
GymPRM <i>Retrieved Information:</i> “...In AD 33, he married Julia Livia, granddaughter of the Roman emperor Tiberius...” <i>Selected Query:</i> “who is the father of julia livia mentioned with gaius rubellius blandus?” <i>Strategic Focus:</i> Correctly targets the father of the spouse.
RePRM <i>Retrieved Information:</i> “...In AD 33, he married Julia Livia, granddaughter of the Roman emperor Tiberius.” <i>Selected Query:</i> “who was the father of roman emperor tiberius?” <i>Strategic Focus:</i> Speculative Selection: Skips the direct father and targets the grandfather’s father. <i>Strategic Issue:</i> Speculative shortcut driven by the prominent entity "Tiberius".
Step 3: Final Answer Determination CoPRM <i>Retrieved Information:</i> “...Julia Livia was the daughter of Drusus Julius Caesar and Livilla..” <i>Final Answer:</i> “drusus julius caesar” ✓ CORRECT <i>Strategic Success:</i> Maintains correct genealogical chain throughout reasoning process
GymPRM <i>Retrieved Information:</i> “...Julia was the daughter of Livilla and Drusus Julius Caesar...” <i>Final Answer:</i> “tiberius” × INCORRECT <i>Strategic Issue:</i> Despite finding the correct answer (Drusus), it selects the specious prominent entity "Tiberius".
RePRM <i>Final Answer:</i> “octavian” × INCORRECT <i>Strategic Issue:</i> Speculative skip led to the wrong lineage (Tiberius’s father).

Table 26: Case Study 1: Step-by-step comparison of three guidance methods on historical multi-hop reasoning. CoPRM maintains correct genealogical focus, whereas GymPRM suffers from hallucination and RePRM exhibits speculative selection.

Question: Who's the son of the Italian navigator who sailed for England and explored the eastern coast of the continent where Christian Sánchez was born?
Ground Truth: Sebastian Cabot

Step 1: Strategic Decomposition of Complex Question

CoPRM

Selected Query: "who was the italian navigator that sailed for england and explored the eastern coast?"

Decomposition Strategy: Strategic Simplification: Retains "sailed for England" to uniquely identify the navigator.

GymPRM

Selected Query: "where was christian sánchez born?"

Strategic Focus: Correctly attempts to resolve the nested entity location.

RePRM

Selected Query: "who was an italian navigator who explored the eastern coast of a specific continent?"

Strategic Focus: Ambiguous Query: Drops the critical constraint "sailed for England".

Strategic Issue: Speculative Selection: Query is under-specified (fits multiple navigators), leading to a guess.

Step 2: Information Processing and Strategic Focus

CoPRM

Retrieved Information: "...England's claims in North America were based on the voyages of the Italian explorer John Cabot (Giovanni Caboto) in the early 16th century..."

Selected Query: "who is the child of john cabot, the italian explorer who sailed for england?"

Strategic Focus: Single focused query (directly targets the child relationship needed)

GymPRM

Retrieved Information: "Christian Sánchez... born... in Sonsonate, El Salvador."

Selected Query: "what italian navigator explored the eastern coast of central america?"

Strategic Focus: Specious Selection: Focuses on specious local detail "Central America" instead of "Sailed for England".

Strategic Issue: Specious Selection: Follows the specious "Central America" path leading to incorrect navigator.

RePRM

Retrieved Information: Documents about Vespucci, Columbus, Cabot...

Selected Query: "who is the son of amerigo vespucci?"

Strategic Focus: Speculative Bet: "Gambles" on Vespucci being the target without evidence.

Strategic Issue: Reward the "shortcut" to a name.

Step 3: Final Answer Determination

CoPRM

Retrieved Information: "Sebastian Cabot was the son of Italian explorer John Cabot (Giovanni Caboto) and his Venetian wife Mattea."

Final Answer: "sebastian cabot" ✓ CORRECT

Strategic Success: Successfully maintains focus on the main question throughout the reasoning chain

GymPRM

Retrieved Information: Information about Columbus discovering Central/South America...

Final Answer: "ferdinand columbus" × INCORRECT

Strategic Issue: Fail: Derailment due to following plausible but wrong details (Central America).

RePRM

Final Answer: "lorenzo vespucci" × INCORRECT

Strategic Issue: Fail: Speculative skip led to the wrong lineage (Vespucci).

Table 27: Case Study 2: Step-by-step comparison showing strategic derailment and error propagation in complex compositional reasoning. CoPRM maintains focus, GymPRM gets distracted by specious local details (Central America), and RePRM speculatively skips necessary steps.