

ADASEARCH: BALANCING PARAMETRIC KNOWLEDGE AND SEARCH IN LARGE LANGUAGE MODELS VIA REINFORCEMENT LEARNING

Anonymous authors

Paper under double-blind review

ABSTRACT

Equipping large language models (LLMs) with search engines via reinforcement learning (RL) has emerged as an effective approach for building search agents. However, overreliance on search introduces unnecessary cost and risks exposure to noisy or malicious content, while relying solely on parametric knowledge risks hallucination. The central challenge is to develop agents that adaptively balance parametric knowledge with external search, invoking search only when necessary. Prior work mitigates search overuse by shaping rewards around the number of tool calls. However, these penalties require substantial reward engineering, provide ambiguous credit assignment, and can be exploited by agents that superficially reduce calls. Moreover, evaluating performance solely through call counts conflates necessary and unnecessary search, obscuring the measurement of true adaptive behavior. To address these limitations, we first quantify the self-knowledge awareness of existing search agents via an F1-based decision metric, revealing that methods such as Search-R1 often overlook readily available parametric knowledge. Motivated by these findings, we propose ADASEARCH, a simple two-stage, outcome-driven RL framework that disentangles problem solving from the decision of whether to invoke search, and makes this decision process explicit and interpretable. This transparency is crucial for high-stakes domains such as finance and medical question answering, yet is largely neglected by prior approaches. Experiments across multiple model families and sizes demonstrate that ADASEARCH substantially improves knowledge-boundary awareness, reduces unnecessary search calls, preserves strong task performance, and offers more transparent, interpretable decision behaviors.

1 INTRODUCTION

Large language models (LLMs) have achieved significant advances across various natural language processing tasks (Brown et al., 2020; Hendrycks et al., 2020; Team et al., 2023; Touvron et al., 2023; Wei et al., 2022; Guo et al., 2025). Yet they face inherent limitations: parametric knowledge alone cannot fully capture domain-specific (Lewis et al., 2020; Yang et al., 2024) or rapidly evolving information (Kasai et al., 2023; Vu et al., 2024), and models may hallucinate when queried beyond their knowledge boundary (Ji et al., 2023; Xu et al., 2024; Huang et al., 2025a; Kalai et al., 2025). To overcome these challenges, integrating LLMs with external knowledge sources has become a crucial direction (Karpukhin et al., 2020; Guu et al., 2020; Nakano et al., 2021; Lazaridou et al., 2022; Shi et al., 2024; Press et al., 2023; Nakano et al., 2021; Feng et al., 2024).

Early work has primarily focused on retrieval-augmented generation (RAG) (Lewis et al., 2020; Borgeaud et al., 2022), which searches for relevant passages and appends them to the context before generation (Shuster et al., 2021; Ram et al., 2023; Jiang et al., 2023; Asai et al., 2024; Wei et al., 2024). However, the relatively static pipeline of RAG overlooks the model’s reasoning process, yielding suboptimal results (Trivedi et al., 2023). Moreover, enabling dynamic multi-turn search is challenging due to the requirement of large-scale annotated trajectories and the non-differentiable nature of the search operation (Schick et al., 2023; Asai et al., 2024; Guan et al., 2025). To address these limitations, treating search as a tool and training tool-integrated LLMs via reinforcement learning (RL) has emerged as a promising paradigm in recent works (Chen et al., 2025b; Jin et al., 2025a;

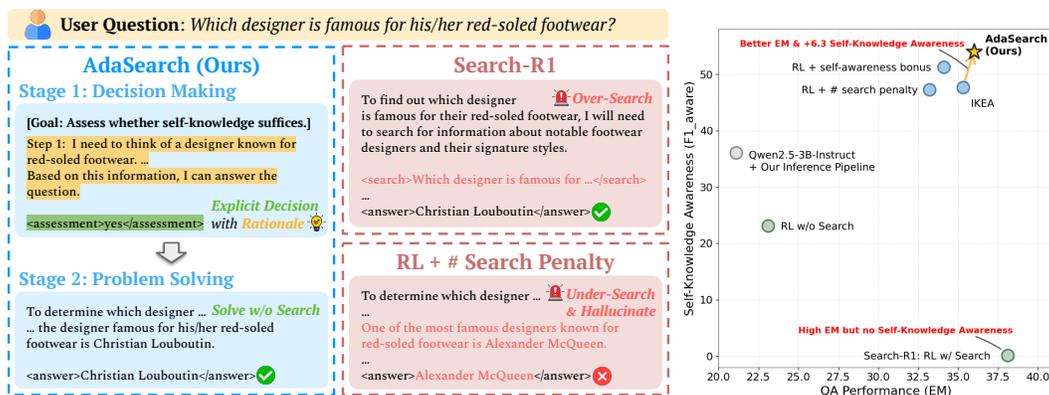


Figure 1: Comparison of RL methods for search agents. *Left*: ADASEARCH provides more transparent and interpretable decisions about whether to search. *Right*: ADASEARCH achieves the best overall self-knowledge awareness while preserving task performance.

Wang et al., 2025; Huang et al., 2025b; Fan et al., 2025). In such setting, models are trained to act as search agents, interleaving reasoning with search in a multi-turn, interactive fashion. This allows LLMs to issue queries adaptively and integrate retrieved information into their reasoning process, outperforming prior prompting-based and supervised approaches.

Nevertheless, building search agents that are truly adaptive remains challenging (Jeong et al., 2024; Qian et al., 2025; Huang et al., 2025b; Wang et al., 2025). Ideally, a search agent should balance parametric knowledge and invoke external search only when necessary. Excessive search calls might raise concerns about efficiency, safety, and privacy, potentially increasing unnecessary costs, exposing the agent to noisy or malicious content, and unintentionally leaking sensitive information. Prior attempts (Huang et al., 2025b; Wang et al., 2025) encourage adaptivity and mitigate overuse by tying rewards and evaluation to the number of search invocations. However, reward shaping around search-call counts requires careful engineering (Table 3), often involving substantial human effort and trial-and-error to determine suitable penalties for redundant search. Furthermore, reward credit assignment (Pignatelli et al., 2023) may be ambiguous: agents may exploit these signals by crafting stronger queries to reduce calls or by superficially avoiding calls even when search is necessary. In addition, naively employing search-call counts as an evaluation metric conflates necessary and unnecessary calls, obscuring the measurement of true adaptive behavior.

Beyond efficiency, a key limitation of prior search-agent frameworks is the lack of *transparency and interpretability* in the decision to invoke search. Because existing methods provide no explicit supervision for this decision, the reasoning process behind “Should I search?” remains implicit and difficult for users to interpret. This limitation becomes critical in real-world, high-stakes environments such as finance, medical question answering, and legal compliance, where users must understand why an agent chooses to rely on parametric knowledge or to consult external information. Consider a case where the agent skips search but then produces an incorrect answer: without an explicit decision rationale, users cannot determine whether the model was overconfident in its knowledge, whether it misinterpreted the query, or whether the failure arose from other factors. Such opacity hinders trust, auditing, and safe deployment of search agents.

In this work, we introduce ADASEARCH, a simple outcome-based RL framework that disentangles and jointly optimizes two abilities: *problem solving* and *deciding whether to invoke search*. Specifically, during training we adopt different prompts to guide LLMs in three modes: (i) solving the problem with internal parametric knowledge, (ii) solving with the problem external search, and (iii) explicitly deciding whether search invocation is needed. At inference time, the model first decides whether the question can be solved using its own knowledge; if so, the model answers the question directly, otherwise search calls will be invoked as part of the model’s reasoning process (Figure 2).

Unlike prior approaches that depend on intricate reward engineering and often suffer from ambiguous credit assignment and implicit decision behavior, ADASEARCH simply leverages task outcomes as rewards with a training framework that provides clearer learning signals, avoids the pitfalls of ambiguous tool-count penalties, and enhances the model’s self-knowledge awareness through ex-

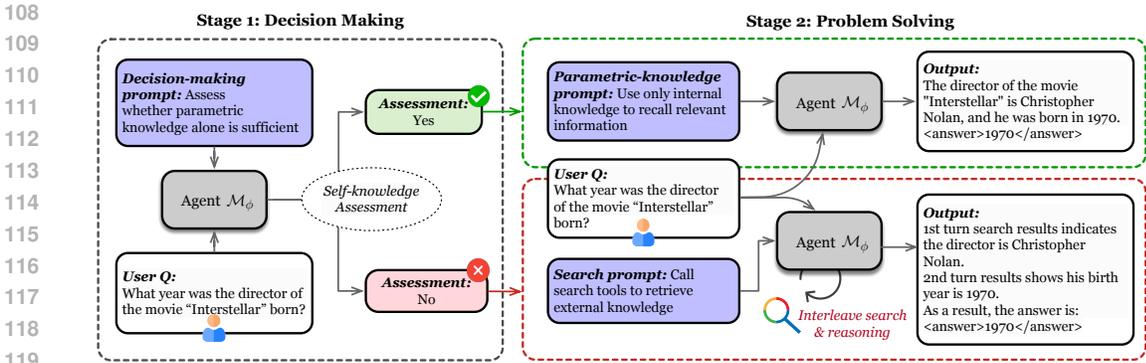


Figure 2: Overview of our proposed ADASEARCH framework. In stage 1, the agent explicitly reasons to decide whether the query can be solved using parametric knowledge. In stage 2, it follows the parametric-knowledge prompt if the knowledge is sufficient; otherwise, it switches to the search prompt to interleave reasoning and search for the final answer.

licit reasoning (Figure 1 left). Experiments and evaluations with our proposed fine-grained self-knowledge awareness F1 metric demonstrate that ADASEARCH fosters better adaptivity, reducing unnecessary search calls while preserving strong task performance (Figure 1 right).

In summary, our contributions are three-fold:

- We propose **ADASEARCH**, a simple yet effective multi-task RL framework that explicitly optimizes both problem solving and decision making to build adaptive search agents.
- **ADASEARCH** improves interpretability by generating explicit reasoning during the decision stage, enhancing the trustworthiness of search agents in real-world scenarios.
- **ADASEARCH** eliminates the need for complex reward engineering while outperforming existing methods in both problem-solving accuracy and self-knowledge awareness, and generalizes across different model sizes and families.

2 EXAMINE SELF-KNOWLEDGE AWARENESS ON VANILLA SEARCH AGENTS

2.1 SELF-KNOWLEDGE AWARENESS F1 SCORE

We evaluate models’ awareness on parametric knowledge by computing the F1 score, where the positive class corresponds to the model deciding that its parametric knowledge is sufficient to answer the input question (equivalently, not invoking search). For brevity, we use the notation $F1_{\text{aware}}$ in the following sections. To determine whether a test instance is a true or false sample, we check whether the model can in fact solve the instance using parametric knowledge alone.

Formally, we define

$$F1_{\text{aware}} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{1}$$

where precision and recall are computed on the binary decision of whether to use search, comparing the model’s choice with an oracle label that indicates whether the question is solvable with parametric knowledge. This metric reflects model’s awareness on its parametric knowledge.

Previous RL-based methods (Jin et al., 2025a; Wang et al., 2025; Huang et al., 2025b) adopt a single prompt that asks the agent to conduct the decision-making process and problem-solving simultaneously. To compute the score for these methods, two cases arise: (1) if the model does not call search, we directly evaluate the EM of its final answer to determine whether the instance is a true or false sample; (2) if the model does call search, we enforce parametric-only reasoning by applying another system prompt s_{param} (Appendix E), and use the correctness of the resulting answer to decide whether the instance is a true or false sample.

2.2 UNDERUSE OF PARAMETRIC KNOWLEDGE IN VANILLA SEARCH AGENTS

In our preliminary experiments, we evaluate prior outcome-based RL methods using Qwen2.5-7B-Base (Team, 2024). Specifically, we include (1) Search-R1 (Jin et al., 2025a), which equips LLMs with search access and optimizes them with outcome rewards, and (2) RL w/o Search, which conducts RL training without search access and fully relies on parametric knowledge. QA performance is measured by exact match (**EM**). We follow the testing setup of Jin et al. (2025a) and report macro-averaged **EM**, **F1_{aware}**, and confusion matrices. We re-evaluate the official Search-R1 checkpoint¹. For RL w/o Search, we report the R1-base results from Jin et al. (2025a).

The results are summarized in Table 1. For Search-R1, although search access improves **EM**, its **F1_{aware}** is 0 because the model invokes search for every query. This overreliance, reflected in many false negatives, exemplifies tool overuse (Qian et al., 2025), where the model searches even when parametric knowledge suffices. In contrast, the model trained without search access achieves a higher **F1_{aware}** but lower **EM** due to limited internal knowledge. These observations highlight the central challenge: enabling greater adaptivity by balancing parametric and external knowledge.

Table 1: Macro-averaged QA performance (**EM**), self-knowledge awareness (**F1_{aware}**), and confusion matrix results on Qwen2.5-7B-Base. **TP**: true positives; **TN**: true negatives; **FP**: false positives; **FN**: false negatives.

Method	EM	F1 _{aware}	TP	TN	FP	FN
Search-R1	39.4	0.0	0.0	72.4	0.0	27.6
RL w/o Search	27.6	27.6	27.6	0.0	72.4	0.0

3 OUR METHOD: ADASEARCH

We introduce ADASEARCH, illustrated in Figure 2. The key idea is to achieve adaptivity without complex reward engineering by disentangling problem solving from the decision of whether to invoke search. Separating these sub-tasks and optimizing them independently enables training policy LLM with simple outcome-based rewards. In this section, we first present the problem formulation (Sec. 3.1), and then describe the ADASEARCH framework (Sec. 3.2). Our main approach is a two-stage training framework, where the first stage targets problem solving and the second stage focuses on decision making. Finally, we conclude with the inference pipeline of ADASEARCH (Sec. 3.3).

3.1 PROBLEM FORMULATION

The problem setting is the same as Jin et al. (2025a). Given an input x , we let policy LLM π_θ interact with the search engine \mathcal{E} to generate a problem solving trajectory τ . Formally,

$$\tau \sim \pi_\theta(\cdot | x; \mathcal{E}). \quad (2)$$

Our goal is to train the policy π_θ to interact with \mathcal{E} by maximizing the following RL objective:

$$J(\theta) = \max_{\pi_\theta} \mathbb{E}_{(x,y) \sim \mathcal{D}, \tau \sim \pi_\theta(\cdot | x; \mathcal{E})} [R(\tau, y)] - \beta \mathbb{D}_{\text{KL}}(\pi_\theta(\cdot | x; \mathcal{E}) \| \pi_{\text{ref}}(\cdot | x; \mathcal{E})), \quad (3)$$

where $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^D$ is the training set with size D , y is the golden answer for input x , $R(\tau, y)$ is the reward function, and π_{ref} is a reference policy. We use GRPO (Shao et al., 2024) as our base RL algorithm.

3.2 ADASEARCH

Stage 1: Problem solving. We focus on incentivizing problem-solving capabilities for the policy LLM. Concretely, we train the policy to utilize (1) parametric knowledge and (2) search call for problem solving. We design two different system prompts: (1) parametric-knowledge prompt s_{param} , which requires the policy to answer using only its internal knowledge, and (2) search prompt s_{search} , which permits the use of search tools. Full prompt details are provided in Appendix E.

The training procedure is depicted in Algorithm 1. For each training instance (x, y) in the training batch, we augment the problem x with both s_{param} and s_{search} , and generate two groups of rollouts:

$$\mathcal{R}_{\text{param}} = \{\tau_{\text{param}}^1, \tau_{\text{param}}^2, \dots, \tau_{\text{param}}^N\} \text{ and } \mathcal{R}_{\text{search}} = \{\tau_{\text{search}}^1, \tau_{\text{search}}^2, \dots, \tau_{\text{search}}^M\}.$$

¹Huggingface: PeterJinGo/SearchR1-nq_hotpotqa_train-qwen2.5-7b-em-ppo

Algorithm 1 ADASEARCH Training

Require: base policy LLM π_{θ_b} ; search engine \mathcal{E} ; training set $\mathcal{D} = (x_i, y_i)$; prompts $s_{\text{param}}, s_{\text{search}}, s_{\text{decision}}$; string concatenation $[\cdot, \cdot]$; solve-rate threshold ρ .

- 1: Initialize policy $\pi_{\theta} \leftarrow \pi_{\theta_b}$
- 2: **for** iteration = 1 to T **do** ▷ Stage-1 Training Loop
- 3: Sample a batch $\mathcal{D}_b \sim \mathcal{D}$
- 4: $\mathcal{R} \leftarrow \{\}; \mathcal{A} \leftarrow \{\}$
- 5: **for each** $(x_i, y_i) \sim \mathcal{D}_b$ **do**
- 6: **for** $g \in \{\text{param}, \text{search}\}$ **do** ▷ Rollout Phase
- 7: **if** g is param **then**
- 8: Generate $\mathcal{R}_g^i = \{\tau_{\text{param}}^n\}_{n=1}^N$, where $\tau_{\text{param}}^n \sim \pi_{\theta}(\cdot \mid [s_{\text{param}}, x_i])$.
- 9: **else**
- 10: Generate $\mathcal{R}_g^i = \{\tau_{\text{search}}^m\}_{m=1}^M$, where $\tau_{\text{search}}^m \sim \pi_{\theta}(\cdot \mid [s_{\text{search}}, x_i]; \mathcal{E})$.
- 11: Compute rewards for all $\tau \in \mathcal{R}_g^i$ with Eq 4.
- 12: Compute group-wise advantages \mathcal{A}_g^i for \mathcal{R}_g^i .
- 13: $\mathcal{R} \leftarrow \mathcal{R} \cup \mathcal{R}_g^i; \mathcal{A} \leftarrow \mathcal{A} \cup \mathcal{A}_g^i$ ▷ Aggregate Rollouts & Advantages
- 14: Update π_{θ} using GRPO with \mathcal{R} and \mathcal{A} .
- 15: Generate dataset $\mathcal{D}_{\text{decision}}$ with $\pi_{\theta}, \mathcal{D}$, and s_{decision} ▷ Detailed in section 3.2
- 16: Stage-2 Training: Update π_{θ} with $\mathcal{D}_{\text{decision}}$ using GRPO
- 17: **return** π_{θ}

For each trajectory $\tau \in \mathcal{R}_{\text{param}} \cup \mathcal{R}_{\text{search}}$, we use regular expressions to extract the final answer \hat{y} from τ and apply exact match $\mathbf{EM}(\hat{y}, y) \rightarrow \{\text{true}, \text{false}\}$, which checks whether the extracted answer is exactly the same as any of the gold answer after normalization, to check the correctness. In the following sections, we use \mathbf{EM} to denote the verification procedure above for simplicity.

The reward design in this stage is simply the binary correctness reward. Formally,

$$R(\tau, y) = \begin{cases} 1.0 & \text{if } \mathbf{EM} = \text{true}, \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

Finally, we compute group-wise advantages as in Shao et al. (2024) and update policy with Eq 3.

Stage 2: Decision making. The goal of stage 2 is to incentivize the decision making of search invocations. Before training, we use the stage 1 policy π_{θ_1} and the parametric-knowledge prompt s_{param} to generate pseudo labels. Specifically, for each training instance (x, y) , we sample K responses from π_{θ_1} and compute the empirical solve rate p using substring exact match (**SubEM**), which checks whether any gold answer is a substring of the extracted answer after normalization:

$$p = \frac{1}{K} \sum_{k=1}^K \mathbb{1}[\mathbf{SubEM}(\hat{y}_k, y) = \text{true}], \quad (5)$$

where \hat{y}_k is the final answer extracted from the k -th response, and $\mathbb{1}[\cdot]$ is the indicator function. We use **SubEM** instead of the stricter **EM** because the objective here is not to evaluate exact answer formatting, but to estimate whether the model can solve the problem using its parametric knowledge. **SubEM** provides a more tolerant measure that avoids penalizing semantically correct answers with minor formatting differences, making it better suited for generating decision-making supervision.

Then, we define a threshold ρ and label instances with $p \geq \rho$ as solvable with parametric knowledge. We use the token `yes` as a label for such instances and the token `no` otherwise. With the pseudo labels, we craft the dataset $\mathcal{D}_{\text{decision}} = \{(x_i, \ell_i)\}_{i=1}^D$, where $\ell_i \in \{\text{yes}, \text{no}\}$, for stage-2 training.

We design a decision-making system prompt s_{decision} that requires explicit reasoning before producing the final decision ℓ , as detailed in Appendix E. [Such explicit reasoning increases transparency and interpretability of the decision-making process, making it clearer why the agent chooses to invoke search or rely solely on parametric knowledge.](#) The reward function in this stage remains a simple binary outcome reward, identical to that used in stage 1 (Equation 4).

3.3 ADASEARCH INFERENCE

We adopt a two-stage inference pipeline (Figure 2). In the first stage, the model is prompted with the system instruction s_{decision} to decide whether it can answer the query using parametric knowledge through explicit reasoning. In the second stage, the model receives the corresponding problem-solving instruction: if it decides it can answer with parametric knowledge, we use s_{param} ; otherwise, we use s_{search} . Note that the stage-1 history is not prepended in stage 2.

4 EXPERIMENTS

4.1 EXPERIMENT SETUP

Models. We conduct experiments on Qwen2.5 (Team, 2024) and Llama-3.2 (Grattafiori et al., 2024) to ensure generalizability. Due to compute constraints, our main experiments use the 3B Instruct variants, with scaling results presented in Section 4.3.

Search environment. Following Jin et al. (2025a), we use E5 (Wang et al., 2022) as the retriever over a 2018 Wikipedia dump, with the implementation from Griggs et al. (2025). For each query, we return the top-3 documents and limit the number of search calls to at most three.

Baselines. We consider three categories of baselines. For **prompting baselines**, we select Direct inference, Chain-of-Thought (CoT) Reasoning (Wei et al., 2022), and RAG (Lewis et al., 2020). For **outcome-RL baselines**, we apply GRPO (Shao et al., 2024) to train models on the parametric prompt s_{param} (RL w/o search) and on the search prompt s_{search} (akin to Search-R1). For **reward-shaping baselines**, we include IKEA (Huang et al. (2025b), Table 3) as a representative method, and also design two baselines. The first one is denoted as ‘‘Naive Shaping’’. Its reward function is defined as

$$R(\tau, y) = \begin{cases} 1.0 - \lambda \cdot (\# \text{ search calls}) & \text{if } \mathbf{EM} = \text{true}, \\ 0 & \text{otherwise,} \end{cases} \quad (6)$$

which is conceptually identical to OTC (Wang et al., 2025) but easier to implement, assigning higher rewards to correct rollouts with fewer search calls. We set $\lambda = 0.05$.

In addition, we introduce another reward-shaping baseline designed to enforce prompt-level search decisions, which closely resembles the objective of ADASEARCH. We refer to this variant as ‘‘Awareness Shaping’’. Although it directly encourages knowledge-aware search behavior, it still underperforms compared to ADASEARCH (Table 2). Its reward function is defined as

$$R(\tau, y) = \mathbb{1}[\mathbf{EM} = \text{true}] + \alpha \cdot \begin{cases} \mathbb{1}[\tau \text{ has search}] & \text{if } p < \rho, \\ \mathbb{1}[\tau \text{ has no search}] & \text{otherwise,} \end{cases} \quad (7)$$

where p is the empirical solve rate defined in Eq 5, ρ is the solve-rate threshold, and α a hyper-parameter controlling the bonus for correct self-knowledge awareness. This baseline encourages awareness by rewarding the model for making the appropriate search decision. In our experiments, we set both α and ρ to 0.5. We estimate p for each sample using the base policy before RL training. Finally, the prompts used for each baseline are detailed in Appendix E.

Training. To ensure fair comparison across training-based baselines, we construct a difficulty-balanced training set of 8,192 samples following the procedure of Huang et al. (2025b). We estimate the solve rate of each problem using the base policy and split the data into easy and hard subsets using a threshold $\rho = 0.5$, then sample equal-sized portions from each subset. A validation set is constructed using the same procedure and used for checkpoint selection. Full details for dataset construction are presented in Appendix B.1. For ADASEARCH stage 2, we use the same threshold $\rho = 0.5$. Full training details are provided in Appendix B.2.

Evaluation. Following Jin et al. (2025a), we use **EM** as the task metric, apply greedy decoding, and evaluate on standard QA benchmarks. For single-hop QA, we use Natural Questions (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), and PopQA (Mallen et al., 2023). For multi-hop QA, we use HotpotQA (Yang et al., 2018), 2WikiMultihopQA (Ho et al., 2020),

Table 2: Results of different model across benchmarks. **EM** denotes exact match. $F1_{\text{aware}}$ denotes the self-knowledge awareness score (see Section 2). **Avg** is computed by averaging **EM** and $F1_{\text{aware}}$ over all benchmarks. The best score is shown in **bold** and the second best in underline.

Method	General QA						Multi-Hop QA						Avg.			
	NQ		TQ		PopQA		HotpotQA		2Wiki		MuSiQue		Bamboogle		EM	$F1_{\text{aware}}$
	EM	$F1_{\text{aware}}$	EM	$F1_{\text{aware}}$	EM	$F1_{\text{aware}}$	EM	$F1_{\text{aware}}$	EM	$F1_{\text{aware}}$	EM	$F1_{\text{aware}}$	EM	$F1_{\text{aware}}$		
Qwen2.5-3B-Instruct																
<i>Prompting Baselines</i>																
Direct Answer	8.1	8.1	24.1	24.1	7.8	7.8	13.2	13.2	19.5	19.5	1.4	1.4	5.6	5.6	11.4	11.4
CoT	14.1	14.1	38.2	38.2	13.5	13.5	17.3	17.3	22.8	22.8	3.9	3.9	24.0	24.0	19.1	19.1
RAG	31.7	0.0	52.7	0.0	38.1	0.0	24.6	0.0	20.1	0.0	4.4	0.0	8.0	0.0	25.7	0.0
<i>Outcome RL Baselines</i>																
RL w/o search	21.7	21.7	44.9	44.9	17.3	17.3	20.5	20.5	28.0	28.0	5.9	5.9	23.2	23.2	23.1	23.1
Search-R1	42.7	0.0	60.1	0.3	<u>43.9</u>	0.0	36.9	0.1	<u>37.3</u>	0.2	16.1	0.0	29.6	0.0	38.1	0.1
<i>Reward-Shaping Baselines</i>																
Naive Shaping	37.8	41.6	54.8	66.9	42.2	55.3	29.8	49.3	33.1	58.4	9.9	15.8	24.8	43.8	33.2	47.3
Awareness Shaping	37.7	<u>45.9</u>	56.1	<u>70.3</u>	41.8	<u>61.3</u>	32.1	<u>57.5</u>	35.9	<u>61.1</u>	12.5	<u>21.7</u>	22.4	41.1	34.1	<u>51.3</u>
IKEA	<u>41.4</u>	37.7	<u>59.5</u>	62.4	44.2	55.2	32.0	58.1	35.0	60.2	11.5	14.2	23.2	<u>46.0</u>	35.3	47.7
ADASEARCH	37.9	49.0	56.8	71.7	42.8	58.8	33.4	<u>57.5</u>	38.5	65.5	14.4	25.2	28.0	50.0	36.0	54.0
Llama-3.2-3B-Instruct																
<i>Prompting Baselines</i>																
Direct Answer	18.8	18.8	42.3	42.3	16.4	16.4	12.5	12.5	16.2	16.2	3.0	3.0	8.8	8.8	16.9	16.9
CoT	25.0	25.0	45.6	45.6	15.8	15.8	16.3	16.3	11.3	11.3	4.5	4.5	35.2	35.2	22.0	22.0
RAG	29.4	0.0	53.0	0.0	35.7	0.0	20.6	0.0	8.3	0.0	4.1	0.0	12.8	0.0	23.4	0.0
<i>Outcome RL Baselines</i>																
RL w/o search	38.2	38.2	54.6	54.6	23.1	23.1	23.7	23.7	26.7	26.7	6.7	6.7	34.4	34.4	29.6	29.6
Search-R1	46.1	1.7	64.4	2.5	44.8	1.5	38.2	6.6	<u>34.8</u>	1.0	14.6	1.3	42.4	0.0	40.7	2.1
<i>Reward-Shaping Baselines</i>																
Naive Shaping	41.2	57.6	59.7	75.3	39.3	56.3	31.2	52.6	33.5	54.8	11.5	21.1	36.8	55.9	36.2	53.4
Awareness Shaping	41.4	<u>58.8</u>	61.5	75.4	<u>41.2</u>	<u>61.4</u>	33.7	57.5	33.7	48.1	13.0	31.4	32.0	58.1	36.6	55.8
IKEA	42.2	58.6	62.0	<u>76.2</u>	41.1	59.9	32.3	<u>58.2</u>	32.9	51.3	11.2	<u>31.9</u>	<u>40.8</u>	<u>63.1</u>	37.5	<u>57.0</u>
ADASEARCH	<u>43.5</u>	62.7	<u>62.9</u>	79.2	40.4	62.3	<u>34.4</u>	59.2	36.2	60.1	<u>13.2</u>	36.5	42.4	64.2	<u>39.0</u>	60.6

MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023). We also report $F1_{\text{aware}}$ (Section 2) to measure self-knowledge awareness.

4.2 MAIN RESULTS

Comparison against prompting and outcome-RL baselines. Table 2 summarizes the results of prompting and outcome-RL baselines. In terms of task performance **EM**, Search-R1 achieves the best overall results across benchmarks, and the RAG baseline consistently outperforms all prompting-based methods. This pattern reflects the characteristics of these benchmarks, where retrieval generally offers an advantage over purely parametric reasoning; the performance gap between RL w/o Search and Search-R1 further supports this observation. Regarding self-knowledge awareness $F1_{\text{aware}}$, however, both the RAG baseline and Search-R1 obtain scores close to zero across benchmarks due to their always-search behavior. The results of Search-R1 highlight the deficiency of naive QA correctness rewards in balancing parametric and external knowledge. Non-search baselines achieve higher $F1_{\text{aware}}$, but because they never explicitly assess their own knowledge boundaries, their awareness remains suboptimal. In comparison, our ADASEARCH maintains competitive **EM** while substantially improving self-knowledge awareness across different model families, achieving 54% to 60% relative gain in $F1_{\text{aware}}$ over Search-R1. These results highlight that our method can effectively elicit self-knowledge awareness without compromising task performance.

Comparison against reward-shaping baselines. As shown in Table 2, our ADASEARCH consistently outperforms intricate reward-shaping baselines in both **EM** and $F1_{\text{aware}}$. First, Naive Shaping generally underperforms other baselines in **EM**, suggesting that naively penalizing tool usage might hurt performance. On the other hand, although Awareness Shaping generally shows improvements in both **EM** and $F1_{\text{aware}}$ over Naive Shaping, the results are still suboptimal. We suspect that this may be due to distribution shifts: we use the base policy before training to compute the empirical solve rate p , but as training proceeds, the solve rate may change as well. Training on such offline-generated labels may therefore introduce noise. Lastly, IKEA (Huang et al., 2025b) generally outperforms the other reward-shaping baselines in both **EM** and $F1_{\text{aware}}$ (except on Qwen2.5-3B-Instruct). Its reward design can be viewed as combining both Naive Shaping (e.g., penalizing tool usage on correct trajectories) and Awareness Shaping (e.g., adding bonuses on incorrect trajectories that involve search calls). However, ADASEARCH still consistently outperforms all reward-shaping

baselines across model families, especially on challenging multi-hop QA where search is more genuinely required. This suggests that reward shaping often over-optimizes for reducing tool calls and fails to adapt to query difficulty. In contrast, ADASEARCH strengthens self-knowledge awareness while achieving better task performance through explicit binary supervision.

4.3 ANALYSIS

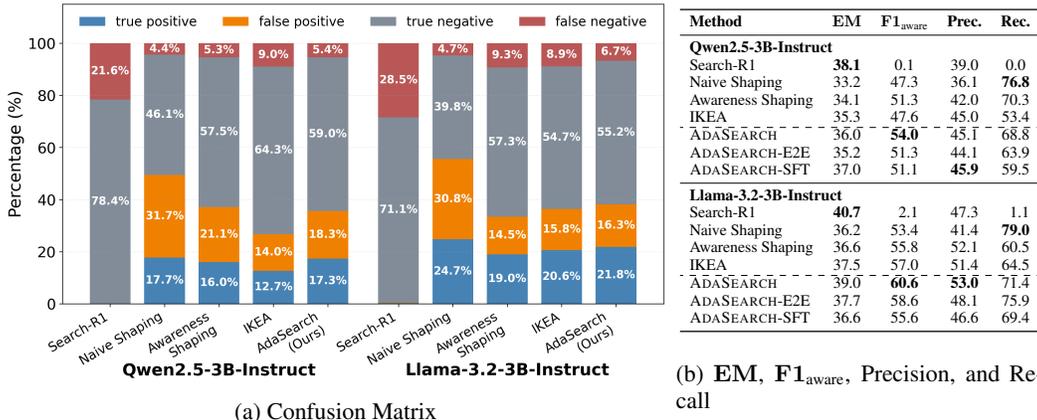


Figure 3: Analysis on self-knowledge awareness. (a) Confusion matrix of different RL methods. (b) Averaged EM, F1_{aware}, precision (Prec.), and recall (Rec.) across benchmarks.

ADASEARCH achieves better self-knowledge awareness. Figure 3 presents the confusion matrices and precision–recall–F1 scores across different methods and model families. For Naive Shaping, we observe the lowest false negative rates but substantially higher false positive rates for both Qwen and Llama (Figure 3a). This pattern suggests that naively penalizing search usage causes the model to underuse search: it spuriously avoids invoking search even when its parametric knowledge is insufficient, thereby drifting away from adaptive behavior. In contrast, ADASEARCH achieves comparable true positive and false negative rates while significantly reducing the false positive rate.

For the other reward-shaping baselines, although they show more balanced false positive and false negative rates, our ADASEARCH still outperforms them in terms of both F1_{aware} and true positive rate. A closer examination shows that ADASEARCH attains precision on par with Awareness Shaping and IKEA, while achieving noticeably higher recall (Figure 3b).

Two-stage training vs joint optimization. To validate the benefit of two-stage training over joint optimization, we implement an end-to-end variant, denoted as **ADASEARCH-E2E**, which jointly optimizes problem solving and self-knowledge awareness. This design enables on-the-fly label generation for self-awareness by using the empirical solve rate of parametric knowledge at each step to produce pseudo labels. Further details are provided in Appendix D.

The results are shown in Figure 3b. ADASEARCH-E2E consistently underperforms ADASEARCH on both EM and F1_{aware}, suggesting that the two-stage approach is more effective than joint training.

RL vs SFT in stage-2 training. A common approach for teaching models to express uncertainty is supervised fine-tuning (SFT) (Lin et al., 2022; Zhang et al., 2024). In stage 2, we implement an SFT variant, **ADASEARCH-SFT**. After computing solve rates as described in Section 3.2, we directly construct target responses: for problems with solve rate $p \geq \rho$, we assign “<assessment>yes</assessment>,” and “<assessment>no</assessment>” otherwise, with $\rho = 0.5$ in our experiments. Additional details are provided in Appendix B.

The results are shown in Figure 3b. While ADASEARCH-SFT improves F1_{aware} over Search-R1 and achieves slightly higher EM on Qwen, it still underperforms ADASEARCH on Llama across both metrics. A closer look reveals that the F1_{aware} gap is especially large on challenging benchmarks such as MuSiQue (Trivedi et al., 2022) and Bamboogle (Press et al., 2023), echoing findings from Chu et al. (2025) that RL generalizes better than SFT to out-of-distribution tasks.

432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485

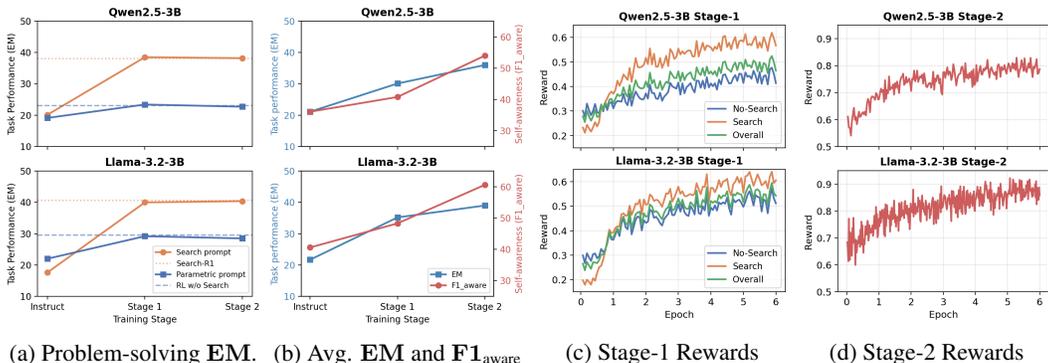


Figure 4: Averaged performance and training rewards across stages. (a) Stage 1 substantially improves problem solving, and Stage 2 does not degrade it. (b) Both test EM and self-knowledge awareness ($F1_{\text{aware}}$) improve throughout training. (c) (d) Stage 1 and Stage 2 respectively incentivize problem-solving and self-knowledge awareness on the training set.

Learning curves. The learning curves are shown in Figure 4. In stage 1, the training reward increases steadily (Figure 4c), and the model’s performance on both search-based and parametric problem solving matches the single-task baselines, Search-R1 and RL w/o Search (Figure 4a). In stage 2, the training reward also increases smoothly (Figure 4d), while self-knowledge awareness improves substantially and task performance continues to rise (Figure 4b). Moreover, Stage 2 does not degrade problem-solving ability under either prompt type (Figure 4a), consistent with recent findings that on-policy RL mitigates forgetting (Shenfeld et al., 2025; Chen et al., 2025a).

Stage-2 labeling hyperparameters. To understand the effect of labeling hyperparameters for stage-2 training, we conduct ablation studies on (1) the number of responses K used to estimate the empirical solve rate in stage 2 and (2) the solve-rate threshold ρ for stage-2 labeling.

The results are shown in Figure 5. For (1) (Figure 5a), increasing K improves both $F1_{\text{aware}}$ and EM by providing a more reliable estimate of the solve rate. Notably, even $K = 1$ performs well (35.1% EM and 52.1% $F1_{\text{aware}}$), likely because RL sharpens output distributions (Yue et al., 2025; He et al., 2025), allowing a single sample to produce effective signals. For (2) (Figure 5b), ρ has limited effect when $\rho \leq 0.5$, aside from a drop in EM at $\rho = 0.3$. As ρ increases further, $F1_{\text{aware}}$ declines while EM rises, since higher ρ encourages more no assessments, increasing search usage.

Retriever choices. To test the generalizability of ADASEARCH across different retrievers, we compare ADASEARCH when train and inference with E5-base (Wang et al., 2022) and BM25 (Robertson & Walker, 1994). As shown in Figure 5c, ADASEARCH continues to improve self-knowledge awareness even with the weaker BM25. While EM drops due to poorer retrieval, the self-awareness signal remains stable, indicating that ADASEARCH is robust to retriever choice.

Model scaling. To assess the robustness of our method across model scales, we train Qwen2.5-7B-Instruct and compare ADASEARCH with Search-R1 and Naive Shaping. As shown in Figure 5c, ADASEARCH boosts self-knowledge awareness on Qwen2.5-7B-Instruct by roughly 60% in $F1_{\text{aware}}$

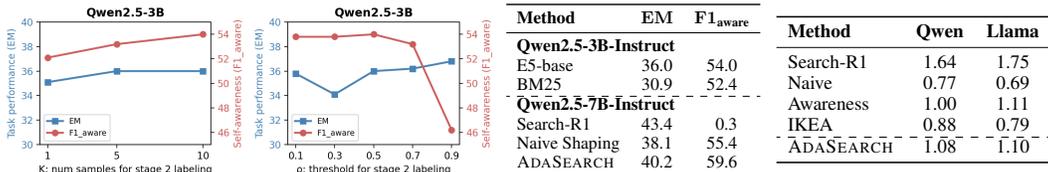


Figure 5: Ablations of ADASEARCH.

486 compared to Search-R1. Relative to Naive Shaping, ADASEARCH improves both EM (+2.1%) and
487 $F1_{\text{aware}}$ (+4.2%). These results indicate that ADASEARCH generalizes well to larger model scales.
488

489 **Average number of searches.** We report the average search calls across all benchmarks.
490 ADASEARCH reduces unnecessary search usage by 34–38% compared to Search-R1, and its fre-
491 quency is similar to Awareness Shaping since both rely on self-knowledge for prompt-level decision
492 making. ADASEARCH uses slightly more searches than Naive Shaping and IKEA, which is expected
493 because it does not directly optimize for minimizing tool calls.
494

495 5 RELATED WORK

497 **Retrieval-augmented generation.** Retrieval-augmented generation (RAG) has become a widely
498 adopted paradigm for equipping LLMs with external knowledge (Shuster et al., 2021; Ram et al.,
499 2023; Jiang et al., 2023; Asai et al., 2024; Wei et al., 2024). By integrating retrieval into the gener-
500 ation process, RAG has demonstrated strong potential to mitigate hallucinations and improve output
501 accuracy across a range of real-world applications (Jin et al., 2025b; Lu et al., 2022; Tan et al., 2024;
502 Xiong et al., 2025). Early approaches follow a static retrieve-and-read pipeline (Lewis et al., 2020;
503 Guu et al., 2020; Izacard et al., 2023), which is effective for factoid queries but relatively limited
504 for multi-step reasoning. Architectural variants such as RETRO (Borgeaud et al., 2022) deliver
505 substantial gains but require model modifications and retraining, whereas in-context RAG (Ram
506 et al., 2023) simply prepends retrieved passages to the prompt, enabling practical off-the-shelf use.
507 More recent works emphasize reasoning-aware retrieval: IRCot (Trivedi et al., 2023) interleaves
508 Chain-of-Thought reasoning (Wei et al., 2022) with retrieval to improve multi-hop question answer-
509 ing, Adaptive-RAG (Jeong et al., 2024) adapts retrieval strategies to query complexity, and Deep-
510 RAG (Guan et al., 2025) formulates retrieval as a step-wise decision process to balance parametric
511 knowledge with external evidence. **Our work shares a similar spirit but focuses on developing search
512 agents via RL to adaptively decide whether to search through explicit reasoning.**

513 **Reinforcement learning for search agents.** Reinforcement Learning (RL) has emerged as a pow-
514 erful paradigm for augmenting LLMs with the ability to invoke external tools during reasoning, ad-
515 dressing tasks that require information beyond the model’s internal knowledge (Guo et al., 2025;
516 Jaech et al., 2024; Li et al., 2025). By treating search call as tools, LLMs are trained to act as
517 search agents, interleaving reasoning with search in a multi-turn, interactive fashion. Representative
518 methods include Search-R1 (Jin et al., 2025a) and ReSearch (Chen et al., 2025b). Following this
519 line of work, recent studies have focused on explicitly optimizing for search adaptivity and effi-
520 ciency. Huang et al. (2025b) train agents to delineate knowledge boundaries and synergize internal
521 and external knowledge; Wang et al. (2025) seek to enhance tool productivity by reducing redundant
522 calls through optimal tool call-controlled policy. However, these approaches rely on intricate reward
523 shaping around search-call counts, requiring substantial manual tuning and trial-and-error. They
524 also face ambiguous credit assignment, where agents exploit reward signals by superficially lower-
525 ing call frequency instead of genuinely improving search behavior. **Moreover, whether to invoke
526 search remains an implicit decision, limiting transparency and hindering deployment in real-world,
527 high-stakes settings.** In contrast, we adopt a simple outcome-based RL framework that separates
528 search decision making from problem solving, avoiding complex reward engineering while provid-
529 ing clearer learning signals. **At inference time, ADASEARCH offers explicit decision rationales,
530 improving transparency and making the agent’s behavior easier to inspect.**

531 6 CONCLUSION

533 In this work, we first conduct analysis and reveal that existing search agents often struggle to recog-
534 nize the limits of their parametric knowledge, leading to unnecessary or excessive search.
535 Motivated by this observation, we propose ADASEARCH, a simple two-stage, outcome-driven RL
536 framework that avoids complex reward engineering while improving both decision quality and task
537 performance. ADASEARCH achieves the strongest self-knowledge awareness on both Qwen and
538 Llama models, reduces unnecessary search calls, preserves task accuracy, and offers more transpar-
539 ent and interpretable decisions about when to search. These results demonstrate that ADASEARCH
is an effective and generalizable approach for building adaptive, trustworthy search agents.

REFERENCES

- 540
541
542 Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to
543 retrieve, generate, and critique through self-reflection. 2024.
- 544 Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Milli-
545 can, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al.
546 Improving language models by retrieving from trillions of tokens. In *International conference on*
547 *machine learning*, pp. 2206–2240. PMLR, 2022.
- 548 Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal,
549 Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are
550 few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- 551
552 Howard Chen, Noam Razin, Karthik Narasimhan, and Danqi Chen. Retaining by doing: The role of
553 on-policy data in mitigating forgetting. *arXiv preprint arXiv:2510.18874*, 2025a.
- 554 Mingyang Chen, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z Pan,
555 Wen Zhang, Huajun Chen, Fan Yang, et al. Learning to reason with search for llms via reinforce-
556 ment learning. *arXiv preprint arXiv:2503.19470*, 2025b.
- 557
558 Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V
559 Le, Sergey Levine, and Yi Ma. SFT memorizes, RL generalizes: A comparative study of founda-
560 tion model post-training. In *Forty-second International Conference on Machine Learning*, 2025.
561 URL <https://openreview.net/forum?id=dYur3yabMj>.
- 562 Yuchen Fan, Kaiyan Zhang, Heng Zhou, Yuxin Zuo, Yanxu Chen, Yu Fu, Xinwei Long, Xuekai
563 Zhu, Che Jiang, Yuchen Zhang, et al. Ssr: Self-search reinforcement learning. *arXiv preprint*
564 *arXiv:2508.10874*, 2025.
- 565 Shangbin Feng, Weijia Shi, Yuyang Bai, Vidhisha Balachandran, Tianxing He, and Yulia Tsvetkov.
566 Knowledge card: Filling llms’ knowledge gaps with plug-in specialized language models. In
567 *ICLR*, 2024.
- 568
569 Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad
570 Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd
571 of models. *arXiv preprint arXiv:2407.21783*, 2024.
- 572 Tyler Griggs, Sumanth Hegde, Eric Tang, Shu Liu, Shiyi Cao, Dacheng Li, Charlie Ruan, Philipp
573 Moritz, Kourosh Hakhamaneshi, Richard Liaw, Akshay Malik, Matei Zaharia, Joseph E. Gonza-
574 lez, and Ion Stoica. Evolving skylr into a highly-modular rl framework, 2025. Notion Blog.
- 575
576 Xinyan Guan, Jiali Zeng, Fandong Meng, Chunlei Xin, Yaojie Lu, Hongyu Lin, Xianpei Han,
577 Le Sun, and Jie Zhou. Deeprag: Thinking to retrieve step by step for large language models.
578 *arXiv preprint arXiv:2502.01142*, 2025.
- 579 Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu,
580 Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms
581 via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.
- 582 Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented
583 language model pre-training. In *International conference on machine learning*, pp. 3929–3938.
584 PMLR, 2020.
- 585
586 Andre Wang He, Daniel Fried, and Sean Welleck. Rewarding the unlikely: Lifting GRPO beyond
587 distribution sharpening. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and
588 Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Lan-
589 guage Processing*, pp. 25559–25571, Suzhou, China, November 2025. Association for Compu-
590 tational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1298. URL
591 <https://aclanthology.org/2025.emnlp-main.1298/>.
- 592 Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and
593 Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint*
arXiv:2009.03300, 2020.

- 594 Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-
595 hop QA dataset for comprehensive evaluation of reasoning steps. In Donia Scott, Nuria Bel,
596 and Chengqing Zong (eds.), *Proceedings of the 28th International Conference on Computational*
597 *Linguistics*, pp. 6609–6625, Barcelona, Spain (Online), December 2020. International Com-
598 mittee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.580. URL <https://aclanthology.org/2020.coling-main.580/>.
599
- 600 Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong
601 Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. A survey on hallucination in large language
602 models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information*
603 *Systems*, 43(2):1–55, 2025a.
- 604 Ziyang Huang, Xiaowei Yuan, Yiming Ju, Jun Zhao, and Kang Liu. Reinforced internal-
605 external knowledge synergistic reasoning for efficient adaptive search agent. *arXiv preprint*
606 *arXiv:2505.07596*, 2025b.
- 607
608 Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane
609 Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning
610 with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):
611 1–43, 2023.
- 612
613 Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec
614 Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv*
615 *preprint arXiv:2412.16720*, 2024.
- 616 Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C Park. Adaptive-rag:
617 Learning to adapt retrieval-augmented large language models through question complexity. In
618 *Proceedings of the 2024 Conference of the North American Chapter of the Association for Com-*
619 *putational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7029–7043,
620 2024.
- 621 Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang,
622 Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM*
623 *computing surveys*, 55(12):1–38, 2023.
- 624 Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang,
625 Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *Proceedings of the*
626 *2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7969–7992, 2023.
- 627
628 Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O Arik, Dong Wang, Hamed Za-
629 mani, and Jiawei Han. Search-r1: Training LLMs to reason and leverage search engines
630 with reinforcement learning. In *Second Conference on Language Modeling*, 2025a. URL
631 <https://openreview.net/forum?id=Rwhi9lideu>.
- 632
633 Jiajie Jin, Yutao Zhu, Zhicheng Dou, Guanting Dong, Xinyu Yang, Chenghao Zhang, Tong Zhao,
634 Zhao Yang, and Ji-Rong Wen. Flashrag: A modular toolkit for efficient retrieval-augmented
635 generation research. In *Companion Proceedings of the ACM on Web Conference 2025*, pp. 737–
636 740, 2025b.
- 637 Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly
638 supervised challenge dataset for reading comprehension. In Regina Barzilay and Min-Yen Kan
639 (eds.), *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*
640 *(Volume 1: Long Papers)*, pp. 1601–1611, Vancouver, Canada, July 2017. Association for Com-
641 putational Linguistics. doi: 10.18653/v1/P17-1147. URL [https://aclanthology.org/](https://aclanthology.org/P17-1147/)
642 [P17-1147/](https://aclanthology.org/P17-1147/).
- 643 Adam Tauman Kalai, Ofir Nachum, Santosh S Vempala, and Edwin Zhang. Why language models
644 hallucinate. *arXiv preprint arXiv:2509.04664*, 2025.
- 645
646 Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick SH Lewis, Ledell Wu, Sergey Edunov, Danqi
647 Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *EMNLP*
(1), pp. 6769–6781, 2020.

- 648 Jungo Kasai, Keisuke Sakaguchi, Ronan Le Bras, Akari Asai, Xinyan Yu, Dragomir Radev, Noah A
649 Smith, Yejin Choi, Kentaro Inui, et al. Realtime qa: What’s the answer right now? *Advances in*
650 *neural information processing systems*, 36:49025–49043, 2023.
- 651 Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Al-
652 bert, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N.
653 Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav
654 Petrov. Natural questions: a benchmark for question answering research. *Transactions of the*
655 *Association of Computational Linguistics*, 2019.
- 656
657 Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E.
658 Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model
659 serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating*
660 *Systems Principles*, 2023.
- 661 Angeliki Lazaridou, Elena Gribovskaya, Wojciech Stokowiec, and Nikolai Grigorev. Internet-
662 augmented language models through few-shot prompting for open-domain question answering.
663 *arXiv preprint arXiv:2203.05115*, 2022.
- 664
665 Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal,
666 Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented gener-
667 ation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:
668 9459–9474, 2020.
- 669 Xuefeng Li, Haoyang Zou, and Pengfei Liu. Torl: Scaling tool-integrated rl. *arXiv preprint*
670 *arXiv:2503.23383*, 2025.
- 671
672 Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in
673 words. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL <https://openreview.net/forum?id=8s8K2UZGTZ>.
- 674
675 Shuai Lu, Nan Duan, Hojae Han, Daya Guo, Seung-won Hwang, and Alexey Svyatkovskiy. Reacc:
676 A retrieval-augmented code completion framework. In *Proceedings of the 60th Annual Meeting*
677 *of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6227–6240, 2022.
- 678
679 Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi.
680 When not to trust language models: Investigating effectiveness of parametric and non-parametric
681 memories. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of*
682 *the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long*
683 *Papers)*, pp. 9802–9822, Toronto, Canada, July 2023. Association for Computational Linguis-
684 tics. doi: 10.18653/v1/2023.acl-long.546. URL <https://aclanthology.org/2023.acl-long.546/>.
- 685
686 Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christo-
687 pher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted
688 question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.
- 689
690 Eduardo Pignatelli, Johan Ferret, Matthieu Geist, Thomas Mesnard, Hado van Hasselt, Olivier
691 Pietquin, and Laura Toni. A survey of temporal credit assignment in deep reinforcement learning.
692 *arXiv preprint arXiv:2312.01072*, 2023.
- 693
694 Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring
695 and narrowing the compositionality gap in language models. In *Findings of the Association for*
696 *Computational Linguistics: EMNLP 2023*, pp. 5687–5711, 2023.
- 697
698 Cheng Qian, Emre Can Acikgoz, Hongru Wang, Xiushi Chen, Avirup Sil, Dilek Hakkani-Tür,
699 Gokhan Tur, and Heng Ji. Smart: Self-aware agent for tool overuse mitigation. *arXiv preprint*
700 *arXiv:2502.11435*, 2025.
- 701
702 Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and
Yoav Shoham. In-context retrieval-augmented language models. *Transactions of the Association*
for Computational Linguistics, 11:1316–1331, 2023.

- 702 Stephen E Robertson and Steve Walker. Some simple effective approximations to the 2-poisson
703 model for probabilistic weighted retrieval. In *SIGIR'94: Proceedings of the Seventeenth Annual*
704 *International ACM-SIGIR Conference on Research and Development in Information Retrieval,*
705 *organised by Dublin City University*, pp. 232–241. Springer, 1994.
- 706
707 Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro,
708 Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can
709 teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36:68539–
710 68551, 2023.
- 711 Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang,
712 Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathemati-
713 cal reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.
- 714 Idan Shenfeld, Jyothish Pari, and Pulkit Agrawal. RL’s razor: Why online reinforcement learning
715 forgets less. *arXiv preprint arXiv:2509.04259*, 2025.
- 716
717 Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke
718 Zettlemoyer, and Wen-tau Yih. Replug: Retrieval-augmented black-box language models. In
719 *Proceedings of the 2024 Conference of the North American Chapter of the Association for Com-*
720 *putational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 8364–8377,
721 2024.
- 722 Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. Retrieval augmentation
723 reduces hallucination in conversation. *arXiv preprint arXiv:2104.07567*, 2021.
- 724
725 Hanzhuo Tan, Qi Luo, Ling Jiang, Zizheng Zhan, Jing Li, Haotian Zhang, and Yuqun Zhang.
726 Prompt-based code completion via multi-retrieval augmented generation. *ACM Transactions on*
727 *Software Engineering and Methodology*, 2024.
- 728 Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut,
729 Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly
730 capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.
- 731
732 Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL [https://qwenlm.](https://qwenlm.github.io/blog/qwen2.5/)
733 [github.io/blog/qwen2.5/](https://qwenlm.github.io/blog/qwen2.5/).
- 734 Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Niko-
735 lay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruiti Bhosale, et al. Llama 2: Open founda-
736 tion and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- 737
738 Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Mul-
739 ti-hop questions via single-hop question composition. *Transactions of the Association for*
740 *Computational Linguistics*, 10:539–554, 2022. doi: 10.1162/tacl.a.00475. URL [https://](https://aclanthology.org/2022.tacl-1.31/)
741 aclanthology.org/2022.tacl-1.31/.
- 742
743 Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving re-
744 trieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Pro-*
745 *ceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1:*
Long Papers), pp. 10014–10037, 2023.
- 746
747 Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan
748 Sung, Denny Zhou, Quoc Le, et al. Freshllms: Refreshing large language models with search
749 engine augmentation. In *Findings of the Association for Computational Linguistics ACL 2024*,
pp. 13697–13720, 2024.
- 750
751 Hongru Wang, Cheng Qian, Wanjun Zhong, Xiusi Chen, Jiahao Qiu, Shijue Huang, Bowen Jin,
752 Mengdi Wang, Kam-Fai Wong, and Heng Ji. Acting less is reasoning more! teaching model to
753 act efficiently. *arXiv preprint arXiv:2504.14870*, 2025.
- 754
755 Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Ma-
jumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv*
preprint arXiv:2212.03533, 2022.

- 756 Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny
757 Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in*
758 *neural information processing systems*, 35:24824–24837, 2022.
- 759 Zhepei Wei, Wei-Lin Chen, and Yu Meng. Instructrag: Instructing retrieval-augmented generation
760 via self-synthesized rationales. *arXiv preprint arXiv:2406.13629*, 2024.
- 761 Guangzhi Xiong, Qiao Jin, Xiao Wang, Yin Fang, Haolin Liu, Yifan Yang, Fangyuan Chen, Zhixing
762 Song, Dengyu Wang, Minjia Zhang, et al. Rag-gym: Optimizing reasoning and search agents
763 with process supervision. *arXiv preprint arXiv:2502.13957*, 2025.
- 764 Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. Hallucination is inevitable: An innate limitation of
765 large language models. *arXiv preprint arXiv:2401.11817*, 2024.
- 766 Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze
767 Gui, Ziran Jiang, Ziyu Jiang, et al. Crag-comprehensive rag benchmark. *Advances in Neural*
768 *Information Processing Systems*, 37:10470–10490, 2024.
- 769 Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov,
770 and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question
771 answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun’ichi Tsujii (eds.), *Proceed-*
772 *ings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–
773 2380, Brussels, Belgium, October–November 2018. Association for Computational Linguistics.
774 doi: 10.18653/v1/D18-1259. URL <https://aclanthology.org/D18-1259/>.
- 775 Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao
776 Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond
777 the base model? In *2nd AI for Math Workshop @ ICML 2025*, 2025. URL <https://openreview.net/forum?id=upehLVgq1b>.
- 778 Hanning Zhang, Shizhe Diao, Yong Lin, Yi Fung, Qing Lian, Xingyao Wang, Yangyi Chen, Heng
779 Ji, and Tong Zhang. R-tuning: Instructing large language models to say ‘I don’t know’. In
780 Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of*
781 *the North American Chapter of the Association for Computational Linguistics: Human Lan-*
782 *guage Technologies (Volume 1: Long Papers)*, pp. 7113–7139, Mexico City, Mexico, June
783 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.394. URL
784 <https://aclanthology.org/2024.naacl-long.394/>.
- 785 Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyang Luo, Zhangchi Feng, and
786 Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Pro-*
787 *ceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume*
788 *3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguis-
789 tics. URL <http://arxiv.org/abs/2403.13372>.

794 A COMPARISON ON REWARD FUNCTIONS

795 We summarize the reward designs of different methods in Table 3. **OTC** (Wang et al., 2025) is
796 designed to minimize search calls and maximize task performance. **IKEA** (Huang et al., 2025b)
797 shares a similar spirit, but focus more on adaptivity by adding a knowledge-boundary bonus (r_{kb}^-)
798 to encourage agents using search on difficult problems. **Naive Shaping** is conceptually similar to
799 OTC but much easier to implement, aiming to encourage agents to use fewer search calls while
800 obtaining the correct answer. **Awareness Shaping** aims to sharply penalize inconsistent transitions
801 between no-search and search behaviors. In contrast, our ADASEARCH use a simple binary rewards
802 to optimize both problem solving and decision making on whether to search.

804 B IMPLEMENTATION DETAILS

805 B.1 TRAINING DATASET CONSTRUCTION

806 To ensure a fair comparison across training-based baselines, we follow the procedure of Huang et al.
807 (2025b) to construct a difficulty-balanced training set. Specifically, we use the base policy with the
808

Table 3: Comparison of different reward designs. Unlike prior methods that rely on complex reward engineering, such as OTC (Wang et al., 2025) and IKEA (Huang et al., 2025b), ADASEARCH uses a simple binary outcome reward to explicitly optimize both problem solving and decision making.

810	
811	
812	
813	
814	Search-R1 (Jin et al., 2025a) & ADASEARCH (Ours)
815	$R(\tau, y) = \begin{cases} 1.0 & \text{if } \mathbf{EM} = \text{true}, \\ 0 & \text{otherwise.} \end{cases}$
816	<hr/>
817	OTC-GRPO (Wang et al., 2025)
818	$R(\tau, y) = \alpha \cdot R_{\text{tool}} \cdot \mathbb{1}[\mathbf{EM} = \text{true}]$, where $R_{\text{tool}} = \begin{cases} 1.0 & \text{if } f(m, n) = n = 0, \\ \cos\left(\frac{m\pi}{2m+c}\right) & \text{if } n = 0, \\ \sin\left(\frac{f(m,n)\pi}{2n}\right) & \text{otherwise.} \end{cases}$ $f(m, n) = \begin{cases} 0 & \text{if } m = n = 0, \\ m & \text{if } n = 0, \\ \frac{2nm}{m+n} & \text{otherwise.} \end{cases}$
819	
820	
821	α : hyperparameter, m : # search calls in τ , n : min # search calls for input x during training.
822	<hr/>
823	IKEA (Huang et al., 2025b)
824	$R(\tau, y) = \begin{cases} -1 & \text{if incorrect format,} \\ \mathbb{1}[\mathbf{EM} = \text{true}] + R_{\text{kb}} & \text{otherwise.} \end{cases}$ $R_{\text{kb}} = \begin{cases} r_{\text{kb}^+} \left(1 - \frac{\text{RT}}{\text{RT}_{\text{max}}}\right) & \text{if } \mathbf{EM} = \text{true}, \\ 0 & \text{if } \mathbf{EM} = \text{false} \ \& \ \text{RT} = 0, \\ r_{\text{kb}^-} & \text{otherwise.} \end{cases}$
825	RT: # search calls; RT_{max} : max # search allowed; $r_{\text{kb}^+}, r_{\text{kb}^-}$: hyperparameters.
826	<hr/>
827	Naive Shaping
828	$R(\tau, y) = \begin{cases} 1.0 - \lambda \cdot (\# \text{ search calls}) & \text{if } \mathbf{EM} = \text{true}, \\ 0 & \text{otherwise,} \end{cases}$
829	where λ is the hyperparameter controlling penalties for search calls.
830	<hr/>
831	Awareness Shaping
832	$R(\tau, y) = \mathbb{1}[\mathbf{EM} = \text{true}] + \alpha \cdot \begin{cases} \mathbb{1}[\tau \text{ has search}] & \text{if } p < \rho, \\ \mathbb{1}[\tau \text{ has no search}] & \text{otherwise,} \end{cases}$
833	where α : self-awareness bonus, p : solve rate for the input problem, and ρ : threshold for solve rate p .
834	<hr/>

835 system prompt s_{probe} (Table 11), which encourages the model to answer directly without accessing
 836 additional information, to generate $K = 10$ responses for each problem in the training set of Jin
 837 et al. (2025a), drawn from Natural Questions (Kwiatkowski et al., 2019) and HotpotQA (Yang et al.,
 838 2018). We then compute solve rates using substring exact matching (SubEM) for verification.
 839 Using a threshold $\rho = 0.5$, we split the dataset into easy and hard subsets and sample 4,096 examples
 840 from each to form the final training set. The same procedure is used to construct a validation set of
 841 2,048 examples, which is used for checkpoint selection during evaluation.

842 We create datasets for each model independently, and all methods evaluated on the same model use
 843 the same dataset to ensure fair comparison.

845 B.2 TRAINING DETAILS

846 Most RL training experiments are conducted on a compute node equipped with 4 NVIDIA A100-
 847 80GB GPUs. The only exception is the outcome-based RL baselines for the 3B models, which are
 848 trained on a machine with 2 NVIDIA H100-94GB GPUs, as these baselines require relatively fewer
 849 computational resources. We use full parameter fine-tuning for all experiments. To optimize training
 850 efficiency, we adopted DeepSpeed with Zero3 offload, gradient checkpointing, FlashAttention-2,
 851 and bfloat16 mixed precision training.

852 We adopt LlamaFactory (Zheng et al., 2024) for SFT. We conduct hyperparameter search on batch
 853 size $\in \{16, 32\}$ and learning rate $\in \{1\text{e-}6, 5\text{e-}6\}$. We only train for one epoch as the training loss
 854 quickly converged to nearly 0. We select the checkpoint that maximizes the product of validation
 855 **EM** and **F1**_{aware}.

856 For RL training, we use SkyRL (Griggs et al., 2025) as our base framework. Hyperparameters for
 857 RL baselines and for ADASEARCH are shown in Table 4 and Table 5, respectively. Most hyperpa-
 858 rameters follow Jin et al. (2025a) and Wang et al. (2025). For reward-shaping baselines, we increase
 859 the group size to 15 to match the compute used in problem-solving training (ADASEARCH stage 1).
 860

861 For the reward-function hyperparameters, we set λ in Naive Shaping (Equation (6)) to 0.05; both α
 862 and ρ in Awareness Shaping to 0.5; and r_{kb^+} and r_{kb^-} in IKEA (Table 3) to 0.2 and 0.05, following
 863 the original IKEA paper (Huang et al., 2025b). In addition, we analyze the reward-function hyper-

parameters for Naive Shaping and compare our training hyperparameters with those in the original IKEA implementation in Appendix C.2.

For ADASEARCH stage 2, we search over the combinations of (batch size, mini-batch size) in (128, 64), (256, 128). For Llama-3.2-3B-Instruct, we use the (128, 64) setting, while for the Qwen2.5 series models, we use (256, 128).

For efficient rollouts, we use vLLM (Kwon et al., 2023) and set the temperature to 1.0, top-p to 1.0, and top-k to -1, following Jin et al. (2025a).

For baselines that involve non-search generation (e.g., RL w/o Search, rollouts for s_{param} in ADASEARCH stage 1, and ADASEARCH stage 2), we set the maximum number of turns to 2, with a maximum generation length of 500 tokens in the second turn. In most cases, the full trajectory of these baselines contains only one turn unless the model fails to follow the required output format (e.g., the answer is not wrapped in `<answer>...</answer>` or the assessment is not wrapped in `<assessment>...</assessment>`). Similar to Search-R1 (Jin et al., 2025a), which appends a nudge prompt (Table 20) when the model violates the required format, we design nudge prompts (Table 21 and 22) to enforce correct formatting. This formatting-correction procedure is applied to all reward-shaping baselines as well to ensure fairness.

We save checkpoints at the end of each epoch. For RL baselines, we select the checkpoint that achieves the highest validation reward. For ADASEARCH stage 1, we use the final checkpoint before collapse. For ADASEARCH stage 2, we select the checkpoint that maximizes the product of validation EM and $F1_{\text{aware}}$.

Table 4: Hyperparameters for RL baselines. *: in the second turn, the maximum generation length is set to 500 (See Appendix B.2).

Hyperparameter	Search-R1	RL w/o Search	Reward-Shaping Baselines
(batch size, mini-batch size)	(512, 256)	(512, 256)	(512, 256)
learning rate	1e-6	1e-6	1e-6
epoch	6	6	6
group size	5	10	15
KL loss coefficient β	0.001	0.001	0.001
PPO clip ratio	0.2	0.2	0.2
warm-up step ratio	0.285	0.285	0.285
max generation length	500	2048*	500
max number of turns	4	2	4
max input length	4096	3072	4096

Table 5: Hyperparameters for ADASEARCH. *: in the second turn, the maximum generation length is set to 500 (See Appendix B.2).

Hyperparameter	ADASEARCH stage 1	ADASEARCH stage 2
(batch size, mini-batch size)	(512, 256)	{(128, 64), (256, 218)}
learning rate	1e-6	1e-6
epoch	6	6
group size	$(s_{\text{search}}, s_{\text{param}}) = (5, 10)$	5
KL loss coefficient β	0.001	0.001
PPO clip ratio	0.2	0.2
warm-up step ratio	0.285	0.285
max generation length	$(s_{\text{search}}, s_{\text{param}}) = (500, 2048^*)$	2048*
max number of turns	$(s_{\text{search}}, s_{\text{param}}) = (4, 2)$	2
max input length	$(s_{\text{search}}, s_{\text{param}}) = (4096, 3072)$	3072

B.3 INFERENCE DETAILS

We perform inference with vLLM (Kwon et al., 2023) and bfloat16 precision on the same compute nodes used for training. For each method, the maximum generation length, maximum number of turns, and maximum input length are set to match the rollout configuration used during training for consistency. Following Jin et al. (2025a), we use greedy decoding for evaluation.

C ADDITIONAL ANALYSIS

In this section, we analyze the effects of hyperparameters across different methods and examine efficiency in terms of the number of search calls and the latency. All analyses are conducted on Qwen2.5-3B-Instruct.

C.1 ADASEARCH STAGE 2: THRESHOLD ρ VS NUMBER OF SEARCH CALLS

We analyze the effect of the solve-rate threshold ρ on the number of search calls, with results shown in Table 6. As ρ increases, the number of search calls rises, which is expected since a higher threshold encourages the model to invoke search more often. However, when ρ becomes too large (e.g., 0.9), $F1_{\text{aware}}$ drops substantially. This suggests overuse of search, where the model chooses to search even when its parametric knowledge is sufficient, as reflected in the decrease in recall and the increase in precision.

Table 6: Analysis on the effect of ρ for ADASEARCH stage 2 (Section 3.2).

ρ	EM	$F1_{\text{aware}}$	Precision	Recall	Avg. Search
0.1	35.8	53.8	44.5	69.3	0.99
0.3	34.1	53.8	43.4	72.5	1.00
0.5	36.0	54.0	45.1	68.8	1.08
0.7	36.2	53.2	48.9	59.6	1.12
0.9	36.8	46.2	55.9	41.0	1.21

C.2 HYPERPARAMETERS OF REWARD-SHAPING BASELINES

We analyze the reward-function hyperparameter of Naive Shaping and Awareness Shaping, and the training hyperparameters of IKEA (Huang et al., 2025b) on Qwen2.5-3B-Instruct.

Naive shaping. We vary the value of λ in Equation 6 over 0.05, 0.1, 0.25, and report EM, $F1_{\text{aware}}$, precision, recall, and the micro-averaged number of search calls. The results are shown in Table 7. As expected, the average number of search calls decreases as λ increases, due to the stronger penalty on invoking search. $F1_{\text{aware}}$ increases, which can be attributed to the increase in recall, while precision remains relatively stable. Lastly, EM decreases with larger λ , possibly because the model underuses search, as reflected by the lower recall and reduced number of search calls.

Table 7: Analysis on the effect of λ for Naive Shaping baseline (Eq 6).

λ	EM	$F1_{\text{aware}}$	Precision	Recall	Avg. Search
0.05	33.2	47.3	36.1	76.8	0.77
0.1	33.0	47.8	35.7	81.3	0.74
0.25	32.1	48.7	35.9	85.6	0.67

Awareness shaping. We vary the value of α in Equation 7 over 0.1, 0.5, 1.0 and report EM, $F1_{\text{aware}}$, precision, recall, and the micro-averaged number of search calls. The results are shown in Table 8. Increasing α improves $F1_{\text{aware}}$ and recall, as expected from assigning a larger bonus to correct decisions. In contrast, precision, average search-call counts, and task EM decrease. These trends suggest that the model may underuse search, reflected in higher recall but lower precision and fewer search calls.

Table 8: Analysis on the effect of α for Awareness Shaping baseline (Eq 7).

α	EM	F1 _{aware}	Precision	Recall	Avg. Search
0.1	36.2	49.5	44.7	57.5	1.12
0.5	34.1	51.3	42.0	70.3	1.00
1.0	33.8	52.5	41.6	76.4	0.98

IKEA. We compare our training hyperparameters for IKEA (Huang et al., 2025b) (Table 4) with those used in the original IKEA paper. In the original work, the authors train with both batch size and mini-batch size set to 256, a smaller learning rate of 5e-7, a much larger warm-up ratio of 0.75, a slightly larger group size of 16, and a total of 120 training steps (approximately 4 epochs). The performance comparison is shown in Table 9. We find that using the original hyperparameters results in IKEA being undertrained. Therefore, we adopt the hyperparameters in Table 4 for our experiments.

Table 9: Comparison of Hyperparameters for IKEA Huang et al. (2025b)

Hyperparameter Setting	EM	F1 _{aware}	Precision	Recall	Avg. Search
Original (Huang et al., 2025b)	32.3	12.4	42.0	8.0	1.06
Ours (Table 4)	35.3	47.6	45.0	59.5	0.88

C.3 COMPARISON ON EFFICIENCY

In this section, we compare the efficiency of each method in terms of the average number of search calls and the average latency, as shown in Table 10. All averages are computed over the full set of testing samples. In addition, we estimate the total search cost on the evaluation benchmarks using the pricing of the Google Search API (5 USD per 1,000 queries)².

Regarding the average number of search calls, ADASEARCH substantially reduces unnecessary searches and costs compared to Search-R1, achieving a 34% reduction. ADASEARCH also incurs a similar number of search calls as Awareness Shaping, as both methods decide whether to invoke search on prompt-level based on self-knowledge. Compared to methods that explicitly minimize search calls (Naive Shaping and IKEA), ADASEARCH uses slightly more searches. This behavior is expected, since ADASEARCH does not directly optimize for minimizing tool calls. Instead, it focuses on avoiding unnecessary searches when parametric knowledge is sufficient, while still achieving a significant reduction relative to Search-R1.

In terms of average latency, although ADASEARCH introduces an additional inference stage that explicitly decides whether to invoke search, its latency remains lower than that of Search-R1 by 20%. These results demonstrate that ADASEARCH not only reduces the average number of searches (and thus API cost) but also decreases inference latency, while providing transparent and interpretable decision rationales that are valuable in real-world applications. Compared to reward-shaping baselines, ADASEARCH incurs a slightly higher latency, but it achieves higher task performance (EM) and demonstrates better discrimination between necessary and unnecessary search calls, as reflected by F1_{aware} (Table 2).

Finally, our framework is orthogonal to reward-shaping methods. Additional reward terms can be incorporated to penalize undesirable search behaviors, such as issuing duplicate queries. We leave a detailed exploration of this direction to future work.

D ADASEARCH-E2E

We unify the two stages in ADASEARCH to obtain an end-to-end variant, ADASEARCH-E2E. The overall training pipeline is shown in Algorithm 2. For each training instance (x, y) , we construct

²<https://developers.google.com/custom-search/v1/overview#pricing>

Table 10: Average number of searches, their estimated costs, and the average latency.

Method	Avg. Search	Est. Costs	Avg. Latency
Qwen2.5-3B-Instruct			
Search-R1	1.64	424.3	0.111
Naive Shaping	0.77	198.4	0.059
Awareness Shaping	1.00	259.0	0.074
IKEA	0.88	228.6	0.066
ADASEARCH	1.08	279.2	0.089

Algorithm 2 ADASEARCH-E2E Training

Require: base policy LLM π_{θ_b} ; search engine \mathcal{E} ; training set $\mathcal{D} = (x_i, y_i)$; prompts $s_{\text{param}}, s_{\text{search}}, s_{\text{decision}}$; string concatenation $[\cdot, \cdot]$; solve-rate threshold ρ .

- 1: Initialize policy $\pi_{\theta} \leftarrow \pi_{\theta_b}$
- 2: /* End-to-End Training */
- 3: **for** iteration = 1 to T **do**
- 4: Sample a batch $\mathcal{D}_b \sim \mathcal{D}$
- 5: $\mathcal{R} \leftarrow \{\}; \mathcal{A} \leftarrow \{\}$
- 6: **for each** $(x_i, y_i) \sim \mathcal{D}_b$ **do**
- 7: **for** $g \in \{\text{param, search, decision}\}$ **do** ▷ Rollout Phase
- 8: **if** g is param **then**
- 9: Generate $\mathcal{R}_g^i = \{\tau_{\text{param}}^n\}_{n=1}^N$, where $\tau_{\text{param}}^n \sim \pi_{\theta}(\cdot \mid [s_{\text{param}}, x_i])$.
- 10: **else if** g is decision **then**
- 11: Generate $\mathcal{R}_g^i = \{\tau_{\text{decision}}^k\}_{k=1}^K$, where $\tau_{\text{decision}}^k \sim \pi_{\theta}(\cdot \mid [s_{\text{decision}}, x_i])$.
- 12: **else**
- 13: Generate $\mathcal{R}_g^i = \{\tau_{\text{search}}^m\}_{m=1}^M$, where $\tau_{\text{search}}^m \sim \pi_{\theta}(\cdot \mid [s_{\text{search}}, x_i]; \mathcal{E})$.
- 14: Compute rewards for all $\tau \in \mathcal{R}_g^i$ with Eq 4.
- 15: Compute group-wise advantages \mathcal{A}_g^i for \mathcal{R}_g^i .
- 16: $\mathcal{R} \leftarrow \mathcal{R} \cup \mathcal{R}_g^i; \mathcal{A} \leftarrow \mathcal{A} \cup \mathcal{A}_g^i$ ▷ Aggregate rollouts & advantages
- 17: Update π_{θ} using GRPO with \mathcal{R} and \mathcal{A} .
- 18: **return** π_{θ}

three variants of the input by applying the parametric knowledge prompt s_{param} , the search prompt s_{search} , and the decision-making prompt s_{decision} .

The pseudo labels for decision making are generated on the fly: for each training instance (x, y) , we estimate the empirical solve rate p from $\mathcal{R}_{\text{param}}$ using **SubEM**, and assign pseudo labels according to whether p exceeds a predefined threshold ρ . The reward function is simply the binary outcome reward, and the remaining optimization follows GRPO (Shao et al., 2024).

The hyperparameters largely follow those of our two-stage method (Table 5), except for the batch and mini-batch sizes. We search over $\{(256, 128), (512, 256)\}$ and use (256, 128) for Qwen and (512, 256) for Llama.

E PROMPT TEMPLATES

Table 11: Knowledge-probing system prompt for dataset construction.

Knowledge-Probing System Prompt s_{probing}

You are a helpful assistant. Your goal is to answer questions using step-by-step reasoning. Use only your internal knowledge to recall and connect relevant information. Do NOT assume access to external tools or documents. Provide the final answer only after completing your reasoning, following the output instructions below.

Output Instructions

1. Wrap your final answer within `<answer>` and `</answer>` tags. For example: `<answer>Wilhelm Conrad Röntgen</answer>`.
2. Only include the direct final answer within these tags; place all explanations and reasoning outside.

Table 12: System prompt for search, adapt from Jin et al. (2025a)

Search System Prompt s_{search}

Answer the given question. You must conduct reasoning 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 `<output>` and `</output>`. You can search as many times as you 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>xxx</answer>`.

Table 13: Parametric-knowledge system prompt. This prompt is used for CoT and RL w/o Search baselines as well.

Parametric-knowledge System Prompt s_{param}

Answer the given question. Always conduct and show your step-by-step reasoning first, then give the final answer. When providing the answer, wrap it inside `<answer>` and `</answer>` without detailed illustrations. For example, `<answer>xxx</answer>`.

Table 14: Decision-making system prompt.

Decision-Making System Prompt s_{decision}

Your goal is to assess whether your knowledge is sufficient to answer the given question. Think step-by-step and then output the final assessment by following the instructions below:

Output Instructions

1. Always output your step-by-step reasoning first.
2. After your reasoning, output a single assessment token wrapped within `<assessment>` ... `</assessment>` tags.
 - If you are confident you can answer the question directly, output: `<assessment>yes</assessment>`
 - Otherwise, output: `<assessment>no</assessment>`
3. Only include the final assessment token within these tags; place all reasoning outside the tags.

1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187

Table 15: System prompt for reward-shaping baselines (Naive Shaping, Awareness Shaping, and IKEA), adapt from Wang et al. (2025)

System Prompt for Reward-shaping Baselines

Answer the given question. You must conduct reasoning 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 `<output>` and `</output>`. You need to make every search call count and gain helpful results. If you find no further external knowledge needed, you can directly provide the answer inside `<answer>` and `</answer>` without detailed illustrations. For example, `<answer>xxx</answer>`.

Table 16: System prompt for direct answer baseline.

System Prompt for Direct Answer Baseline

Answer the given question. When providing the answer, wrap it inside `<answer>` and `</answer>` without detailed illustrations. For example, `<answer>xxx</answer>`.

Table 17: System prompt for RAG baseline.

System Prompt for RAG Baseline

Answer the given question based on the provided documents. When providing the answer, wrap it inside `<answer>` and `</answer>` without detailed illustrations. For example, `<answer>xxx</answer>`.

Table 18: Default user prompt template, where `question` will be replaced with the actual input question.

Default User Prompt Template

Question: {question}

Table 19: RAG User prompt template, where `document_i` and `question` will be replaced with the retrieved document and the actual input question, respectively.

RAG User Prompt Template

Doc 1: {document_1}
Doc 2: {document_2}
Doc 3: {document_3}
Question: {question}

1188 Table 20: Retry prompt used when the model produces an invalid action under the search prompt
 1189 s_{search} .

1190
 1191 **Retry Prompt (Search)**
 1192
 1193 My previous action is invalid. If I want to search, I should put the query between `<search>` and
 1194 `</search>`. If I want to give the final answer, I should put the answer between `<answer>`
 1195 and `</answer>`. Let me try again.
 1196

1197
 1198 Table 21: Retry prompt used when the model fails to wrap the answer in
 1199 `<answer>...</answer>` under the parametric-knowledge prompt s_{param} .

1200
 1201 **Retry Prompt (Parametric-knowledge)**
 1202
 1203 I should put the answer between `<answer>` and `</answer>`.
 1204
 1205

1206 Table 22: Retry prompt used when the model produces a search call under the parametric-knowledge
 1207 prompt s_{param} .

1208
 1209 **Retry Prompt (Parametric-knowledge; When search is invoked)**
 1210
 1211 My previous action is invalid. I should rely on my own step-by-step reasoning to solve the
 1212 problem and then put the answer between `<answer>` and `</answer>`. Let me try again.
 1213
 1214

1215 F CASE STUDIES

1216 F.1 CASE 1: PARAMETRIC KNOWLEDGE IS SUFFICIENT

1217
 1218 Table 23: Input question. The answer is “Christian Louboutin”.

1219
 1220
 1221 **Input Question (Answer: Christian Louboutin)**
 1222
 1223 Which designer is famous for his/her red-soled footwear?
 1224
 1225
 1226

1227 Table 24: ADASEARCH stage-1 output. The model explicitly evaluates its own knowledge and
 1228 concludes that it is sufficient.

1229
 1230 **ADASEARCH Stage-1 Output (Answer: Christian Louboutin)**
 1231
 1232 Step 1: I need to think of a designer known for red-soled footwear.
 1233 Step 2: One famous designer known for red-soled footwear is Christian Louboutin.
 1234 Step 3: Christian Louboutin is a French fashion designer who is renowned for his distinctive
 1235 red-soled high-heeled shoes.
 1236 Step 4: Based on this information, I can answer the question.
 1237 `<assessment>yes</assessment>`
 1238
 1239
 1240
 1241

1242 Table 25: ADASearch stage-2 output. The model correctly answer the question using parametric
1243 knowledge.

1244
1245 ADASearch Stage-2 Output (Answer: Christian Louboutin)
1246
1247 To determine which designer is famous for his/her red-soled footwear, we need to consider some
1248 of the most well-known designers and their iconic designs.
1249
1250 1. Christian Louboutin is a renowned French fashion designer who is famous for creating
1251 red-soled high-heeled shoes.
1252 2. Other designers like Jimmy Choo and Alexander McQueen have also designed red-soled
1253 shoes, but Christian Louboutin is particularly known for this specific style.
1254
1255 Given this information, the designer famous for his/her red-soled footwear is Christian
1256 Louboutin.
1257
1258 <answer>Christian Louboutin</answer>

1258 Table 26: Search-R1 output. The agent overuses search on problems where its parametric knowledge
1259 is sufficient.

1260
1261 Search-R1 Output (Answer: Christian Louboutin)
1262
1263 To find out which designer is famous for their red-soled footwear, I will need to search for
1264 information about notable footwear designers and their signature styles.
1265
1266 <search>Which designer is famous for red-soled footwear</search>
1267 <output>
1268 Doc 1: "Christian Louboutin"
1269 Christian Louboutin Christian Louboutin (; born 7 January 1964) is a French fashion designer
1270 whose high-end stiletto footwear incorporates shiny, red-lacquered soles that have become his
1271 signature. Initially a freelance designer for fashion houses, he started his own shoe salon in Paris,
1272 with his shoes finding favor with celebrity clientele. He has partnered with other organizations
1273 for creative projects including limited edition pieces, gallery exhibits, and even a custom bar.
1274 His company has since branched out into men's footwear, handbags, fragrances and makeup.
1275 Louboutin was born and raised in Paris's 12th arrondissement. He was the only son of Roger, a
1276 Doc 2: "Christian Louboutin"
1277 tempered to 28,000 Mexican pesos (1471.89 dollars). Christian Louboutin Christian Louboutin (;
1278 born 7 January 1964) is a French fashion designer whose high-end stiletto footwear incorporates
1279 shiny, red-lacquered soles that have become his signature. Initially a freelance designer for
1280 fashion houses, he started his own shoe salon in Paris, with his shoes finding favor with celebrity
1281 clientele. He has partnered with other organizations for creative projects including limited
1282 edition pieces, gallery exhibits, and even a custom bar. His company has since branched out into
1283 men's footwear, handbags, fragrances and makeup. Louboutin was born and raised in Paris's
1284 12th arrondissement. He
1285 Doc 3: "Christian Louboutin"
1286 on his high heel shoes, commonly referred to as ""sammy red soled shoes"". His signature red
1287 soles were first created in 1993. ""My sketches were not reproduced exactly as I had designed
1288 them and I couldn't figure out why. The two dimensional sketch was so powerful on paper, but
1289 when turned into a three-dimensional object, it was somehow lacking energy. Frustrated after
1290 having tried different things to liven up the design, I spontaneously grabbed my assistant's red
1291 nail polish and painted the sole. I instantly knew that this would be a success!"". Christian
1292 Louboutin's red-bottom colour code is Pantone 18-1663
1293 </output>
1294
1295 From the search results, it is clear that Christian Louboutin is famous for his red-soled
1296 footwear.
1297
1298 <answer>Christian Louboutin</answer>

1296 Table 27: Naive Shaping output. The model produces an incorrect answer without explicit self-
1297 knowledge assessment.

1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310

Naive Shaping Output (Answer: Christian Louboutin)

To determine which designer is famous for their red-soled footwear, let’s consider some prominent designers and their notable styles.

One of the most famous designers known for red-soled footwear is Alexander McQueen. He was known for his bold and innovative designs, including red-soled shoes that became iconic in his collections.

Given this information, the designer famous for his/her red-soled footwear is Alexander McQueen.

<answer>Alexander McQueen</answer>

1311
1312

1313 F.2 CASE 2: PARAMETRIC KNOWLEDGE IS INSUFFICIENT

1314

1315 Table 28: Input question. The answer is “Bardney”.

1316
1317

Input Question (Answer: Bardney)

1318 Southrey is a village in the civil parish of which village in the West Lindsey district of Lin-
1319 colnshire, England?
1320
1321

1322
1323

1324 Table 29: ADASEARCH stage-1 output. After self-assessment, the model is not confident in answer-
1325 ing using parametric knowledge.

1326
1327

ADASEARCH Stage-1 Output (Answer: Bardney)

1328 Step 1: Identify the key information in the question. The question asks about a civil parish that
1329 contains the village of Southrey.

1330 Step 2: Recall that a civil parish is a territorial unit of local government in England, and it is often
1331 part of a larger administrative division such as a district.

1332 Step 3: Think about the West Lindsey district in Lincolnshire, England. I need to find which
1333 village in this district contains Southrey.

1334 Step 4: Without specific knowledge of the exact location, I would need to search for the civil
1335 parish that includes Southrey. However, I don’t have immediate access to this information.

1336 Step 5: Given the lack of specific knowledge about the exact civil parish containing Southrey, I
1337 cannot answer the question directly.

<assessment>no</assessment>

1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349

1350 Table 30: ADASEARCH stage-2 output. The model invokes search and produces a correct answer.
1351

1352

1353

ADASEARCH Stage-2 Output (Answer: Bardney)

1354

1355

1356

To determine the village in the West Lindsey district of Lincolnshire, England, that is the civil parish of Southrey, I will need to search for this information.

1357

1358

`<search>`Village in the West Lindsey district of Lincolnshire that is the civil parish of Southrey`</search>`

1359

`<output>`

1360

Doc 1: Bardney

1361

1362

1363

1364

1365

1366

1367

consists of crafts, trade and community stands, funfair rides, bar and BBQ, sports, children’s races, car boot sales, classic vehicles, and an Exemption Dog Show. Originally part of the Wraggøe Wapentake, and of the Lincoln poor law union in the Parts of Lindsey, the parish is now part of West Lindsey District, and Lincolnshire County Council. The civil parish has become part of a group that consists of Bardney, Apley and Stainfield parishes. The villages of Southrey and Kingthorpe are also included. The 2011 electoral arrangements are: Bardney lies between 7 and 17 metres above sea level, on the edge

1368

Doc 2: Snelland

1369

1370

1371

1372

1373

1374

Snelland Snelland is a village and civil parish in the West Lindsey district of Lincolnshire, England It is situated approximately north-east from the city and county town of Lincoln and about from the town of Market Rasen. The civil parish includes the hamlet of Swinhorpe. The population at the 2011 census was included in the civil parish of Friesthorpe. Snelland is listed in “Domesday Book” of 1086 as “Sneleslunt”, with seven households. The nearby deserted medieval village of Swinhorpe is listed as “Sonetorp”, with three households. There is evidence that a Cistercian grange existed at Swinhorpe, attached to Kirkstead Abbey.

1375

Doc 3: “South Thoresby”

1376

1377

1378

1379

1380

1381

1382

South Thoresby South Thoresby is a village and civil parish in the East Lindsey district of Lincolnshire, England. It is situated north-east from Horncastle and south-east from Louth. South Thoresby is about east from the A16 in the Lincolnshire Wolds. The civil parish extends much further to the south, over the A16 and to include the hamlets of Calceby (a former medieval village) and Driby, extending southwards to border with Langton by Spilsby. To the south-west is Haugh. It lies close to the northern boundary with Aby with Greenfield. The parish council administers Swaby, South Thoresby and Haugh, although these

1383

`</output>`

1384

1385

1386

From the search results, it is clear that Southrey is a village in the civil parish of Bardney. Therefore, the village in the West Lindsey district of Lincolnshire, England, that is the civil parish of Southrey is Bardney.

1387

1388

`<answer>`Bardney`</answer>`

1389

1390

1391

1392

1393

1394

Naive Shaping Output (Answer: Bardney)

1395

1396

1397

1398

1399

1400

1401

1402

1403

To determine the village in the West Lindsey district of Lincolnshire, England, that is the civil parish of Southrey, we need to find information about the civil parishes and their boundaries.

Southrey is a village in the civil parish of Little Sleaford. Little Sleaford is a village located in the West Lindsey district of Lincolnshire, England.

Therefore, the civil parish of Southrey is Little Sleaford.

`<answer>`Little Sleaford`</answer>`

1404 F.3 CASE 3: ADASEARCH UNDERUSE SEARCH

1405

1406

Table 32: Input question. The answer is “Mary Elizabeth Braddon”.

1407

1408

1409 Input Question (Answer: Mary Elizabeth Braddon)

1410

1411

Who is the author of Lady Audley’s Secret?

1412

1413

1414 Table 33: ADASEARCH stage-1 output. The model holds a false belief, which leads to an incorrect
1415 self-assessment.

1416

1417

1418 ADASEARCH Stage-1 Output (Answer: Mary Elizabeth Braddon)

1419

Step 1: I need to recall the author of the novel ”Lady Audley’s Secret.”

1420

Step 2: I know that ”Lady Audley’s Secret” is a novel.

1421

Step 3: I remember that the author of ”Lady Audley’s Secret” is Mrs. Wilkie Collins.

1422

Step 4: I can confirm that Mrs. Wilkie Collins is indeed the author of this novel.

1423

Step 5: Based on my knowledge, I can answer the question directly.

1424

<assessment>yes</assessment>

1425

1426

1427 Table 34: ADASEARCH stage-2 output. The model maintains its false belief, resulting in insufficient
1428 search.

1429

1430

1431 ADASEARCH Stage-2 Output (Answer: Mary Elizabeth Braddon)

1432

To determine the author of ”Lady Audley’s Secret,” we need to consider the historical context
1433 and the well-known works of that era.

1434

”Lady Audley’s Secret” is a novel published in 1862. The author of this novel is Mrs.
1435 Wilkie Collins, a prominent English novelist of the Victorian era.

1436

1437

Therefore, the author of ”Lady Audley’s Secret” is <answer>Mrs. Wilkie Collins</answer>

1438

1439

1440

1441

1442

1441 F.4 CASE 4: ADASEARCH OVERUSE SEARCH

1442

1443

Table 35: Input question. The answer is “Politician”.

1444

1445

1446 Input Question (Answer: Politician)

1447

1448

What is Javier Alva Orlandini’s occupation?

1449

1450

1451

1452

1453

1454

1455

1456

1457

1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511

Table 36: ADASEARCH stage-1 output. The model is not confident in its self-knowledge and therefore abstains.

ADASEARCH Stage-1 Output (Answer: Politician)

Step 1: I need to determine Javier Alva Orlandini’s occupation.
Step 2: I do not have any information about Javier Alva Orlandini in my knowledge base.
Step 3: Without specific information about this person, I cannot answer the question directly.
<assessment>no</assessment>

Table 37: ADASEARCH stage-2 output. The model produces a correct answer after search.

ADASEARCH Stage-2 Output (Search) (Answer: Politician)

To determine Javier Alva Orlandini’s occupation, I will need to search for information about him. Let me start with a search.

<search>Javier Alva Orlandini occupation</search>
<output>

Doc 1: "Javier Alva Orlandini"

Javier Alva Orlandini Javier Alva Orlandini (born December 11, 1927) is a lawyer and a Peruvian politician. A prominent member of the political party Popular Action, he serves as the President of the party. He served as 2nd Vice President of Peru during the government of Fernando Belaúnde from 1980 to 1985.. Throughout his life he became a Deputy (1963-1968), Minister of Government and Police (1965-1966), Senator (1980-1985 and 1990-1992), President of the Senate (1981-1982) and Congressman (1995-2000). In 2002, he became a Magistrate of the Constitutional Court, at the same time being elected by Congress as the President of

Doc 2: "Luis Orlandini"

Luis Orlandini Luis Orlandini (born 1964) is a Chilean guitarist and professor at the University of Chile and the Pontifical Catholic University of Chile. Orlandini studied guitar with Ernesto Quezada at the University of Chile and then with Eliot Fisk at the School of Music of Cologne. In 1989, he obtained First Place at the ARD International Music Competition in Munich, thus initiating an international career that has led him to perform in various countries and record with different European labels. He performs mainly as a soloist, but has also played with groups of different natures of chamber music, one

Doc 3: "Javier Alva Orlandini"

the Court for a 3-year term. He was a candidate for the Presidency of the Republic in 1985, but got 4% of the popular vote, being defeated by Alan García. He later served as President of Peru’s constitutional tribunal. He has written several books, including ""Responding to the Dictatorship"", ""Yesterday, Today and Tomorrow"", the ""Vicious Circle"", ""I Minister"", ""Word of Honor"" and ""Yes I swear"", and numerous legal articles. The audits, representative legislative, legal, academic and professional are of extensive quality and are recognized by the Peruvian people and their various institutions that have given many honorary degrees. Javier Alva

</output>

Based on the search results, Javier Alva Orlandini is a lawyer and a Peruvian politician. He served as the President of the party Popular Action and was the 2nd Vice President of Peru during the government of Fernando Belaúnde from 1980 to 1985.

<answer>Politician</answer>

1512 Table 38: Model output under the parametric-knowledge system prompt s_{param} . The model is uncer-
 1513 tain about its answer, even though the answer is correct.

1514

1515

1516 ADASEARCH Stage-2 Output (Parametric Knowledge) (Answer: Politician)

1517 To determine Javier Alva Orlandini’s occupation, we need to look up information about him.
 1518 Given that the question does not provide specific details, we can infer based on common
 1519 knowledge or a search.

1520

1521 Javier Alva Orlandini is a Mexican politician who has held various positions in the Mexi-
 1522 can government. Given his background, it is most likely that he is a politician.

1523

1524 However, without a specific source, we can’t be certain. For the purpose of this question,
 1525 we will assume his occupation is based on his known roles in Mexican politics.

1526 `<answer>Politician</answer>`

1527

1528
 1529

1530 F.5 CASE 5: MULTI-HOP FAILURE OF ADASEARCH

1531
 1532

Table 39: Input question. The answer is “1978”.

1533

1534

1535 Input Question (Answer: 1978)

1536 The director of Panic 5 Bravo was born in what year?

1537

1538
 1539

1540 Table 40: ADASEARCH stage-1 output. The model abstains because it lacks sufficient information.

1541

1542 ADASEARCH Stage-1 Output (Answer: 1978)

1543

1544 Step 1: To answer this question, I need to know the director of the movie ”Panic 5 Bravo”.
 1545 However, there is no widely known movie or film titled ”Panic 5 Bravo” that I am aware of. This
 1546 suggests it might be a fictional or obscure title.

1547 Step 2: Without knowing the director of this specific movie, I cannot determine the year of birth
 1548 of the director.

1549 Step 3: Since I do not have the necessary information to answer the question, I cannot provide a
 1550 year of birth for the director.

1551 `<assessment>no</assessment>`

1552
 1553
 1554
 1555
 1556
 1557
 1558
 1559
 1560
 1561
 1562
 1563
 1564
 1565

1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619

Table 41: ADASEARCH stage-2 output (part 1). The model invokes search but makes a faulty inference (highlighted in red).

ADASEARCH Stage-2 Output (Answer: 1978)

To find out the year the director of "Panic 5 Bravo" was born, I will need to gather information about this film and its director. Let's start by searching for more details about the film "Panic 5 Bravo".

`<search>`The director of Panic 5 Bravo and the year of birth`</search>`

`<output>`

Doc 1: "Panic (2000 film)"

Panic (2000 film) Panic is a 2000 American crime film directed by Henry Bromell and starring William H. Macy, Neve Campbell, Tracey Ullman, John Ritter and Donald Sutherland. Despite not being a box office success, ""Panic"" received universal critical acclaim from critics, some of which called it one of the best movies of 2000. The performances of William H. Macy, John Ritter, and Donald Sutherland were hailed by critics. Alex (Macy), a sad-eyed mournful man, goes into psychotherapy disclosing that he is a hit man and also tells the doctor (Ritter) after a few sessions that he is attracted to

Doc 2: "Panic (2000 film)"

originality from the tone: quirky yet convincing, irreverent yet moral."" Panic (2000 film) Panic is a 2000 American crime film directed by Henry Bromell and starring William H. Macy, Neve Campbell, Tracey Ullman, John Ritter and Donald Sutherland. Despite not being a box office success, ""Panic"" received universal critical acclaim from critics, some of which called it one of the best movies of 2000. The performances of William H. Macy, John Ritter, and Donald Sutherland were hailed by critics. Alex (Macy), a sad-eyed mournful man, goes into psychotherapy disclosing that he is a hit man and also tells the doctor

Doc 3: "A Town Called Panic (film)"

A Town Called Panic (film) A Town Called Panic () is a 2009 internationally co-produced stop-motion animated adventure fantasy comedy family film directed by Stéphane Aubier and Vincent Patar and co-written by Aubier, Guillaume Malandrin, Patar, and Vincent Tavier. The film is based on the TV series of the same name and stars Aubier, Jeanne Balibar, Nicolas Buisse, Véronique Dumont, Bruce Ellison, Frédéric Jannin, Bouli Lanners, and Patar, among others. It premiered at the 2009 Cannes Film Festival and was the first stop-motion film to be screened at the festival. The film was released theatrically in Belgium on 17 June

`</output>`

From the search results, it seems "Panic 5 Bravo" is not a widely known film. **However, the first result mentions a film called "Panic (2000 film)" directed by Henry Bromell. Given this information, I will now search for Henry Bromell's birth year.**

1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673

Table 42: ADASEARCH stage-2 output. Continuing from part 1, the model uses the wrong director as the search query, which ultimately leads to an incorrect answer.

ADASEARCH Stage-2 Output (Continued) (Answer: 1978)

`<search>`Henry Bromell birth year`</search>`

`<output>`

Doc 1: "Henry Bromell"

Henry Bromell Alfred Henry Bromell (September 19, 1947 – March 18, 2013) was an American novelist, screenwriter, and director. Bromell joined the crew of NBC police drama "Panic" in 1994. He served as a writer and co-executive producer for the show's third season. He contributed to writing seven episodes for the season. He was promoted to executive producer for the fourth season and wrote a further 17 episodes. He scaled back his involvement with the fifth season and became a consulting producer. He wrote a further two episodes before leaving the crew at the end of the season in 1997.

Doc 2: "Henry Bromell"

screenwriter and director Caroline Thompson. He then married writer Trish Soodik, who died of cancer in January 2009; they had a son, William. His third wife was Sarah. Bromell died March 18, 2013 of a heart attack, at UCLA Santa Monica hospital at age 65. Henry Bromell Alfred Henry Bromell (September 19, 1947 – March 18, 2013) was an American novelist, screenwriter, and director. Bromell joined the crew of NBC police drama "Panic" in 1994. He served as a writer and co-executive producer for the show's third season. He contributed to writing seven episodes for the season. He was promoted

Doc 3: "Henry Bromell"

He contributed to a total of 26 episodes as a writer over three seasons with the series. He returned as a co-writer and co-executive producer for the feature-length follow-up "Homicide: The Movie" in 2000. He wrote and produced for many television series, including "Chicago Hope", "Northern Exposure", "Homicide: Life on the Street", "Brotherhood", "Carnivàle", and "Rubicon". He was a consulting producer, and later Executive Producer on the Showtime series "Homeland" at the time of his death and wrote four episodes: "The Good Soldier", "Representative Brody", "Q&A", and "Broken Hearts". He was awarded a for "The Good Soldier", and he was

`</output>`

From the search results, Henry Bromell was born in 1947. Therefore, the director of "Panic (2000 film)" was born in 1947.

`<answer>`1947`</answer>`