OASES: Outcome-Aligned Search-Evaluation Co-Training for Agentic Search

Erhan Zhang[1]*, **Yiqun Chen**[1]*, **Zechun Niu**[1], **Wei Yang**[3], **Xiaochi Wei**[2], **Yan Gao**[2], **Yi Wu**[2], **Yao Hu**[2], **Jiaxin Mao**[1]†
[1] Renmin University of China [2] Xiaohongshu Inc. [3] University of Southern California
erhanzhang@ruc.edu.cn, chenyiqun990321@ruc.edu.cn,
maojiaxin@gmail.com

Abstract

Agentic search enables language models to solve knowledge-intensive tasks by adaptively acquiring external evidence over multiple steps. Reinforcement learning with verifiable rewards (RLVR) has emerged as a widely adopted training paradigm for search agents, yet outcome-only rewards are sparse and provide limited credit assignment for intermediate search actions. Existing process-reward methods therefore seek to densify supervision through proxy signals, external evaluators, or likelihood-based information gain. However, proxy rewards can deviate from the final outcome objective, while fixed evaluators can become stale as the search policy evolves, leading to unreliable process supervision. To address these challenges, we propose **OASES**, an **O**utcome-**A**ligned **S**earch-**E**valuation **S**upervision framework for agentic search. OASES derives outcome-aligned process rewards by evaluating how well each intermediate search state supports answering the original question. It further co-trains the search policy and the state evaluator on policy, allowing the evaluator to adapt to evolving search behavior and provide more reliable process rewards. Experiments on five QA benchmarks show that OASES consistently outperforms strong RL baselines, with further analyses confirming the benefits of outcome-aligned process rewards and search-evaluation co-training. [3]

1 Introduction

Large language models (LLMs) are increasingly moving beyond one-shot generation toward *agentic* problem solving, where a model interacts with external tools and environments over multiple steps [41, 23]. Among these paradigms, *agentic search* has become an important framework for knowledge-intensive question answering, allowing models to actively acquire task-relevant evidence [14, 9, 27]. Unlike standard retrieval-augmented generation (RAG), which typically relies on a fixed retrieval stage [13], agentic search adaptively acquires information needed for problems that cannot be solved from parametric memory alone.

Recent work has explored reinforcement learning (RL) to improve the search and reasoning abilities of such agents. In particular, reinforcement learning with verifiable rewards (RLVR) [12, 6] provides a natural training paradigm, since many knowledge-intensive tasks admit automatically checkable outcome signals, such as exact match or F1. By optimizing the outcome objective, RLVR avoids reliance on supervised trajectories and allows the agent to discover its own search strategies. However, applying RLVR to agentic search remains challenging from both the *search* and *evaluation* sides.

*Equal contribution.
†Corresponding author.
[3]The code is available at https://github.com/Meow-E/Oases.

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

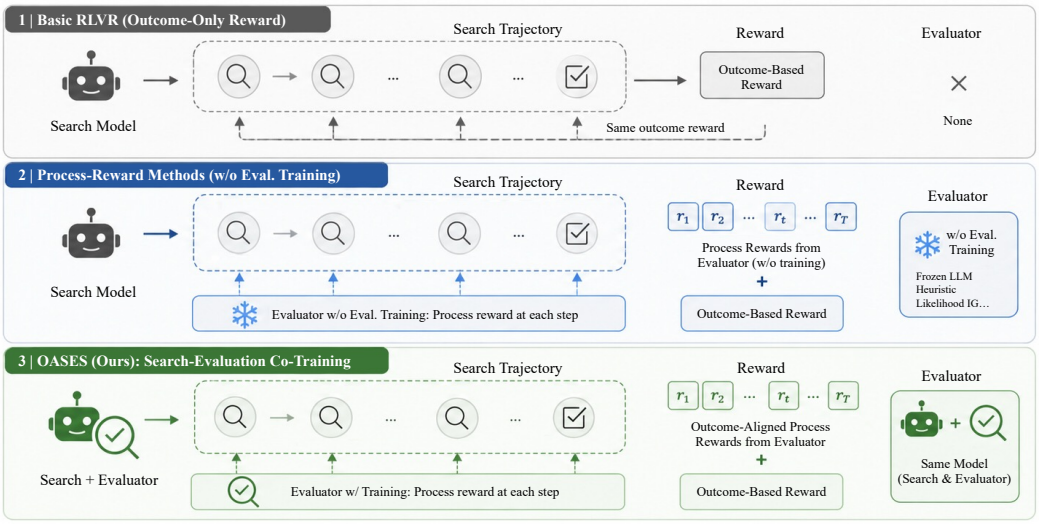


Figure 1: Comparison of reward designs for RLVR-based agentic search. Outcome-only RLVR provides trajectory-level feedback. Existing process-reward methods use proxy signals or separate evaluators for step-level feedback. OASES co-trains a single policy for search and state evaluation, producing outcome-aligned process rewards while reducing evaluator–policy mismatch.

On the search side, the key challenge is to teach the agent how to search by telling which intermediate actions are truly useful [19]. A search trajectory may contain multiple rounds of reasoning, query formulation, evidence acquisition, and answer refinement, but outcome rewards provide feedback only after the final answer is produced. Such sparse and delayed supervision makes it difficult to assign credit to individual search decisions [28, 35, 38]. As a result, RLVR for agentic search naturally calls for dense process supervision that can guide the policy at intermediate search states.

On the evaluation side, however, reliable process supervision is difficult to obtain. Existing methods therefore rely on proxy signals, such as retrieved-document similarity [44, 21], external LLM evaluators [36, 43, 32], or heuristic rules [26]. While these signals provide denser feedback, they often optimize intermediate criteria such as relevance, format compliance, or judge preference, rather than whether the search state improves the final answer under the outcome objective. *In short, proxy process supervision provides denser feedback, but it is not aligned with the outcome objective.*

Another group of methods defines process rewards through information gain, for example by measuring how the likelihood or perplexity of the gold answer changes after new information is acquired. Although these rewards are more directly tied to the final answer, their gain estimates can still be unreliable. Specifically, using a fixed model to derive the likelihood and information gain may suffer from a distributional mismatch with the evolving search policy [37], while directly using the policy model itself to score the intermediate search states can be inconsistent with the model's own generation behavior under partial search prefixes [31, 17, 8]. Optimizing such likelihood-based evaluation may reduce this mismatch, but risks answer leakage by fitting the gold answer before sufficient evidence is available. *In short, likelihood-based information-gain rewards are closer to the outcome objective, but they can still provide inaccurate process supervision due to evaluator–policy mismatch.*

These limitations suggest that improving agentic search requires jointly addressing the search and evaluation sides. The search policy needs dense supervision to overcome sparse outcome feedback, while the evaluator must provide reliable feedback that remains aligned with the outcome objective and compatible with the evolving policy. This motivates a unified framework where state evaluation is not a separate scoring procedure, but a learnable capability optimized together with search.

In this paper, we propose **OASES**, an **O**utcome-**A**ligned **S**earch–**E**valuation **S**upervision framework for agentic search. OASES derives outcome-aligned process rewards by evaluating how well each intermediate search state supports answering the original question. Specifically, given the evidence accumulated up to a search state, an evaluator generates a state-conditioned answer, which is then

scored by the same verifiable outcome metric used for the final answer. The improvement of this score across consecutive search states is used as a dense process reward, converting sparse outcome feedback into fine-grained credit assignment while keeping the reward signal aligned with the final objective. To reduce evaluator–policy mismatch, OASES further co-trains a shared-parameter model as both the search policy and the state evaluator. This allows the evaluator to adapt to the evolving search behavior and enables the policy to jointly improve its evidence acquisition and state-evaluation capabilities, without relying on a separate process reward model. Figure 1 illustrates this distinction: compared with outcome-only RLVR and existing process-reward methods that rely on separate or non-optimized evaluators, OASES jointly optimizes search and state evaluation under a shared policy, yielding dense rewards that are both outcome-aligned and adaptive to the evolving search behavior.

We evaluate OASES on multi-hop QA benchmarks, showing that it consistently improves agentic search performance over strong RL baselines. Further analyses demonstrate that outcome-aligned process rewards provide more effective credit assignment, and that search–evaluation co-training outperforms alternatives based on frozen evaluators or search-only optimization.

Our contributions are summarized as follows:

- We propose **OASES**, an outcome-aligned search–evaluation supervision framework that converts outcome improvements across search states into dense process rewards.
- We introduce search–evaluation co-training, where a single policy learns both to search and to evaluate whether its accumulated information supports the final answer.
- We conduct experiments on multi-hop QA benchmarks, showing that OASES improves agentic search over strong RL baselines and mitigates reward sparsity through more effective credit assignment.

2 Related Work

Agentic Search and RL-based Training. Recent advancements have focused on training LLMs as agentic search systems that interleave multi-step reasoning with external retrieval. Leveraging RL frameworks such as PPO [24] or GRPO [25], representative agentic search methods such as Search-R1 [9] and R1-Searcher [27] have demonstrated that such RL-based paradigms significantly bolster performance in complex search tasks. Parallel to these end-to-end agents, extensive research has investigated the optimization of specific search components, including planning [2], query reformulation [20, 16, 1], document selection [10, 15], and answer generation [39]. More recently, there is a growing emphasis on the joint optimization of these modular components to achieve superior global performance [4, 3]. Orthogonal to training-based agentic search, RSE [34] studies how to recycle rollout-level experience at test time by reusing intermediate conclusions and failure patterns across rollouts.

Process Rewards for Agentic Search. Process supervision aims to alleviate the sparsity of outcome-only rewards by assigning feedback to intermediate search actions [18, 19]. Existing agentic-search methods obtain such feedback from several sources. Monte Carlo or tree-search methods propagate final outcome scores to intermediate decisions [42], but are often rollout-intensive and costly to integrate into online RL. Retrieval-oriented methods reward search steps using evidence relevance or coverage [21, 26, 44], while evaluator-based methods use learned reward models or external LLM judges to assess intermediate behavior [5, 38, 43, 32, 36]. These approaches provide denser feedback, but they introduce separate scoring criteria or separate evaluators, which can be costly and may suffer from evaluator–policy mismatch. Closely related to our work, information-gain-based methods assign rewards according to whether a search state increases the likelihood of the gold answer [37, 31, 17, 8]. However, likelihood-based gains can be inaccurate, as they may be affected by evaluator–policy mismatch and fail to reflect whether the current state improves the model's final answer[4].

[4]We further discuss why these methods could not fully support a joint optimization of the search and evaluation models in Appendix I.

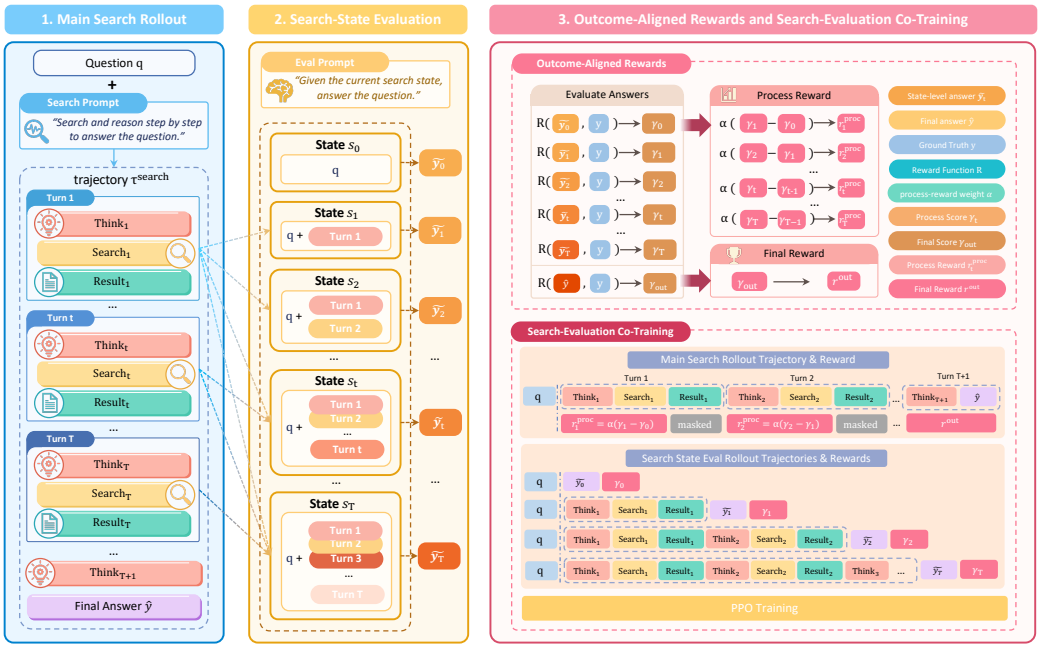


Figure 2: Overview of the OASES framework. The policy generates a main search trajectory τ^{search} and extracts an ordered sequence of search states $\{s_t\}_{t=0}^{T}$. For each state, the same policy generates a state-conditioned evaluation answer $\tilde{y}_t$, whose score γ_t reflects how well the accumulated information supports the final answer. OASES derives process rewards from state-score differences $\gamma_t - \gamma_{t-1}$, combines them with the final outcome score γ_{out}, and co-trains search and state evaluation with PPO.

3 OASES: Outcome-Aligned Search-Evaluation Co-Training

This section presents OASES, an outcome-aligned search–evaluation framework for agentic search. As shown in Figure 2, OASES uses the same verifiable outcome signal for state-level and final-answer evaluation, converting outcome improvements across search states into dense process rewards. It then co-trains a single policy for search and state evaluation with PPO [24].

3.1 Preliminaries

Task setup. Let $\mathcal{D} = \{(q_i, y_i)\}_{i=1}^{N}$ denote a training set of question–answer pairs, where q_i is a question and y_i is its ground-truth answer. For simplicity, we omit the sample index i and write $(q, y) \sim \mathcal{D}$. The goal of agentic search is to learn a policy that, given a question q, interacts with an external search environment over multiple turns and produces a final answer that matches y.

Agentic search trajectory. We consider a multi-turn search setting with T search turns. At each search turn t, the model generates a reasoning segment c_t and a search action a_t, and then receives an observation o_t from the environment. After the final search turn, the model generates a final reasoning segment c_{T+1} and outputs an answer $\hat{y}$. The resulting main search trajectory is

$$\tau^{\text{search}} = \big(q, \{(c_t, a_t, o_t)\}_{t=1}^{T}, c_{T+1}, \hat{y}\big), \tag{1}$$

where c_t, a_t, o_t, and $\hat{y}$ correspond to `<think>`, `<search>`, `<result>`, and `<answer>`, respectively, and c_{T+1} is the final reasoning segment before answering. To expose the accumulated information available along the trajectory, we define a sequence of search states:

$$s_t = \begin{cases} \big(q, \{(c_{t'}, a_{t'}, o_{t'})\}_{t'=1}^{t}\big), & t = 1, \ldots, T, \\ q, & t = 0, \end{cases} \tag{2}$$

where s_t denotes the information state after the first t search turns, and s_0 contains the original question without external search observation. Thus, $\{s_t\}_{t=0}^{T}$ forms an ordered sequence of search states, where each state contains exactly the information available to the model at that point.

3.2 Search-State Evaluation

OASES uses the same model π_θ for both the main search task and the search-state evaluation task, with different behaviors specified by different prompt templates. Let $P^{\text{search}}(q)$ denote the prompt for multi-turn agentic search, and let $P^{\text{eval}}(s_t)$ denote the prompt for search-state evaluation. Detailed prompt templates are provided in Appendix E. In the evaluation prompt, the model answers the original question conditioned on state s_t, without further search. For each question q, we construct the main search trajectory as

$$\tau^{\text{search}} \sim \pi_\theta(\cdot \mid P^{\text{search}}(q)), \tag{3}$$

where τ^{search} follows the trajectory format defined in Eq. (1). From this trajectory, we extract the ordered search states $\{s_t\}_{t=0}^{T}$ according to Eq. (2). For each search state s_t, we construct a state-conditioned evaluation trajectory by prompting the same model to answer the original question, and collect all such trajectories as

$$\mathcal{T}^{\text{eval}} = \left\{ \tau_t^{\text{eval}} = \big(s_t, \tilde{y}_t\big) \;:\; \tilde{y}_t \sim \pi_\theta(\cdot \mid P^{\text{eval}}(s_t)),\; t = 0, 1, \ldots, T \right\}. \tag{4}$$

Here, $\tilde{y}_t$ is generated from the current state without further search. Together, τ^{search} and $\mathcal{T}^{\text{eval}}$ provide the trajectory-level information needed to estimate improvements across search states.

The evaluator observes the complete search state available to the policy, including retrieved evidence and accumulated reasoning. Its score measures the model's ability to answer from that state. The model may combine retrieved information with parametric knowledge; for example, identifying a bridge entity can make a question answerable before every supporting fact has been retrieved. The score reflects these contributions under the final-answer objective.

3.3 Outcome-Aligned Process Rewards

Answer quality scoring. Given the main search trajectory τ^{search} and the search-state evaluation trajectories $\mathcal{T}^{\text{eval}}$, OASES first scores all generated answers with the same verifiable answer scoring function $R(\cdot, \cdot)$. Specifically, we compute

$$\gamma_t = R(\tilde{y}_t, y), \quad t = 0, 1, \ldots, T, \qquad \gamma_{\text{out}} = R(\hat{y}, y), \tag{5}$$

where γ_t is the state-level score of the evaluation answer $\tilde{y}_t$, and γ_{out} is the outcome score of the final answer $\hat{y}$. Since $\tilde{y}_t$ is generated from s_t without further search, γ_t measures how well the model can answer the original question from the current search state.

Rewards for the main search trajectory. As illustrated in Figure 2, OASES converts the ordered state-level scores $\{\gamma_t\}_{t=0}^{T}$ into stepwise process rewards for the main search trajectory, and combines them with a final outcome reward. Intuitively, the t-th search turn is rewarded according to how much the newly accumulated information improves state answerability, while the final answer receives the standard outcome reward. Formally, we define

$$r_t^{\text{proc}} = \alpha\big(\gamma_t - \gamma_{t-1}\big), \quad t = 1, \ldots, T, \qquad r^{\text{out}} = \gamma_{\text{out}}, \tag{6}$$

where $\alpha \geq 0$ controls the relative contribution of process rewards in the overall reward. We assign each process reward r_t^{proc} to the end of the segment that triggers the t-th search interaction, and assign the outcome reward r^{out} to the end of the final answer segment. Let m_t and m_{ans} denote these token positions, and let L_{search} be the number of generated tokens in the main search rollout. The token-level reward sequence is

$$\rho_k^{\text{search}} = \sum_{t=1}^{T} \mathbb{I}[k = m_t]\, r_t^{\text{proc}} + \mathbb{I}[k = m_{\text{ans}}]\, r^{\text{out}}, \qquad k = 1, \ldots, L_{\text{search}}. \tag{7}$$

As shown by the gray blocks in the "Main Search Rollout Trajectory & Reward" row of Figure 2, the PPO loss is applied only to model-generated tokens. We therefore mask out tokens from external search observations, i.e., the `<result>` segments, during optimization.

Summing the process rewards over consecutive search states gives

$$\sum_{t=1}^{T} r_t^{\text{proc}} = \alpha \sum_{t=1}^{T} (\gamma_t - \gamma_{t-1}) = \alpha(\gamma_T - \gamma_0). \tag{8}$$

Algorithm 1: OASES Training within PPO

Input: training set $\mathcal{D}$, batch size B, policy π_θ, value function V_ϕ, process-reward weight α
Output: updated policy parameters θ and value-function parameters ϕ
Initialize policy parameters θ, critic parameters ϕ, search buffer $\mathcal{B}^{\text{search}}$ and evaluation buffer $\mathcal{B}^{\text{eval}}$;
for *each training epoch* **do**
  for *each batch* $\mathcal{Q} \subset \mathcal{D}$ *with* $|\mathcal{Q}| = B$ **do**
    Clear search buffer $\mathcal{B}^{\text{search}}$ and evaluation buffer $\mathcal{B}^{\text{eval}}$;
    for *each question–answer pair* $(q, y) \in \mathcal{Q}$ **do**
      Sample the main search trajectory τ^{search} by Eq. (3), structured as Eq. (1);
      Extract the search states $\{s_t\}_{t=0}^{T}$ according to Eq. (2);
      Sample the search-state evaluation trajectories $\mathcal{T}^{\text{eval}} = \{\tau_t^{\text{eval}}\}_{t=0}^{T}$ by Eq. (4);
      Compute state-evaluation scores $\{\gamma_t\}_{t=0}^{T}$ and final outcome score γ_{out} by Eq. (5);
      Compute the process rewards $\{r_t^{\text{proc}}\}_{t=1}^{T}$ and the outcome reward r^{out} according to Eq. (6);
      Construct the token-level rewards ρ^{search} by Eq. (7), and append $(\tau^{\text{search}}, \rho^{\text{search}})$ to $\mathcal{B}^{\text{search}}$;
      for $t = 0$ **to** T **do**
        Construct the token-level rewards $\rho^{\text{eval},t}$ by Eq. (9), and append $(\tau_t^{\text{eval}}, \rho^{\text{eval},t})$ to $\mathcal{B}^{\text{eval}}$;
    Combine $\mathcal{B}^{\text{search}}$ and $\mathcal{B}^{\text{eval}}$ into a joint training buffer $\mathcal{B}$;
    Compute returns and advantages from $\mathcal{B}$, then update π_θ and V_ϕ using Eq. (10) and Eq. (11);

Here, γ_0 measures answerability before retrieval, so the accumulated reward credits improvements over the model's initial answerability. Unchanged consecutive scores yield zero process reward. Score changes reflect contributions from retrieved evidence, accumulated reasoning, and parametric knowledge. Appendix B provides the trajectory-level analysis and initial-state evaluation details.

Rewards for search-state evaluation trajectories. In addition to supervising the main search rollout, OASES trains the model on the search-state evaluation trajectories. For each $\tau_t^{\text{eval}} = (s_t, \tilde{y}_t)$, we use its state-level answer score γ_t as the terminal reward. Let m_t^{eval} denote the token position at the end of the evaluation answer $\tilde{y}_t$, and let $L_{\text{eval},t}$ be the number of generated tokens in the t-th evaluation rollout. The token-level reward sequence for this evaluation rollout is

$$\rho_k^{\text{eval},t} = \mathbb{I}\big[k = m_t^{\text{eval}}\big]\,\gamma_t, \qquad k = 1, \ldots, L_{\text{eval},t}, \quad t = 0, 1, \ldots, T. \tag{9}$$

Together, the search and evaluation objectives provide complementary supervision. The search objective rewards actions that improve state answerability, while the evaluation objective trains the model to judge whether the accumulated information supports the final answer. Because the process rewards are defined by score differences $\gamma_t - \gamma_{t-1}$, they telescope over the search trajectory and form an outcome-consistent redistribution of answerability improvement. We provide further analysis in Appendix B.

3.4 Search-Evaluation Co-Training

Algorithm 1 summarizes the PPO training procedure of OASES. A central design is to co-train search and evaluation with a single policy model π_θ. For each training question, the model first acts as the search policy and generates a main search trajectory τ^{search} with reward sequence ρ^{search}. This reward sequence trains the model to improve the final-answer outcome, while state-level process rewards assign outcome-aligned credit to intermediate search decisions. We store these search rollouts in a search buffer $\mathcal{B}^{\text{search}}$.

OASES then reuses the intermediate states along τ^{search} to construct one state-conditioned evaluation rollout τ_t^{eval} for each search state s_t, yielding $\mathcal{T}^{\text{eval}} = \{\tau_t^{\text{eval}}\}_{t=0}^{T}$ with reward sequences $\{\rho^{\text{eval},t}\}_{t=0}^{T}$. In these rollouts, the same model answers the original question conditioned on each state, and its reward is computed against the ground-truth answer. Thus, the evaluation rollouts train the model to produce better outcome-aligned answers from the information available at each state. We store these evaluation rollouts in an evaluation buffer $\mathcal{B}^{\text{eval}}$.

We compute return targets $\hat{G}$ and advantage estimates $\hat{A}$ over the joint buffer $\mathcal{B} = \mathcal{B}^{\text{search}} \cup \mathcal{B}^{\text{eval}}$ using generalized advantage estimation (GAE). We also apply a format penalty to trajectories that

violate the required interaction format, encouraging the model to follow the search protocol. Let $r(\theta) = \pi_\theta(x \mid x_<)/\pi_{\theta_{\text{old}}}(x \mid x_<)$ be the PPO importance ratio for a generated token x conditioned on its prefix $x_<$. The actor objective is

$$\mathcal{L}_{\text{actor}}(\theta) = -\mathbb{E}_{(x,\hat{A})\sim\mathcal{B}^{\text{search}}\cup\mathcal{B}^{\text{eval}}}\left[\min\Big(r(\theta)\hat{A},\ \text{clip}(r(\theta), 1-\epsilon, 1+\epsilon)\,\hat{A}\Big)\right], \tag{10}$$

where ϵ is the clipping coefficient. The critic regresses return targets over the same joint buffer:

$$\mathcal{L}_{\text{critic}}(\phi) = \mathbb{E}_{(x_<,\hat{G})\sim\mathcal{B}^{\text{search}}\cup\mathcal{B}^{\text{eval}}}\left[\Big(V_\phi(x_<) - \hat{G}\Big)^2\right]. \tag{11}$$

This co-training scheme couples the two roles through both rewards and data distributions. The evaluation task provides denser and more outcome-aligned feedback for the search task by estimating how much each intermediate state supports the final answer. At the same time, the search task supplies the evaluation task with states sampled from the current policy's own search distribution. This is important because an evaluator trained on states that differ from the policy's evolving search behavior may provide inaccurate process rewards. By optimizing search and evaluation rollouts with the same policy parameters, OASES keeps the evaluator adapted to the search policy while using the evaluator to improve search credit assignment. At inference time, only the search policy is executed, so OASES introduces no additional inference-time cost. Further discussion on the necessity of joint optimization is provided in Appendix I.

4 Experiments

We conduct experiments to answer the following research questions:

RQ1: Does OASES consistently outperform existing baselines?

RQ2: Is joint search–evaluation optimization effective compared with non-co-trained policy or frozen evaluators?

RQ3: How much do outcome-aligned process rewards contribute?

RQ4: How sensitive is OASES to the reward weight α?

4.1 Experimental Settings

Datasets. We train OASES on HotpotQA [40] and 2WikiMultihopQA [7], and evaluate it on NQ [11], HotpotQA, 2WikiMultihopQA, Bamboogle [22], and MuSiQue [30].

Evaluation. We evaluate model performance using exact match (EM) and token-level F1. During training, validation and checkpointing are performed every fixed number of PPO updates, determined by the training batch size and dataset size.

Baselines. We compare OASES with three categories of baselines.[5]

Non-agentic methods. Direct LLM answers questions using only parametric knowledge. **Naive RAG** [13] retrieves relevant passages once and appends them to the input for answer generation.

Agentic search methods. Search-o1 [14] combines dynamic retrieval with a Reason-in-Documents module for evidence analysis. **Search-R1** [9] trains interleaved reasoning and search with outcome-based RL and retrieved-token masking. **R1-Searcher** [27] uses two-stage outcome-based RL to learn search invocation and evidence use.

Process-supervision methods. StepSearch [44] rewards improvements in retrieved-document coverage of gold evidence and penalizes redundant retrieval. **ReasonRAG** [42] uses MCTS to obtain preferences over intermediate RAG actions and optimizes them with DPO. **TIPS** [37] shapes turn-level rewards using increases in a teacher model's likelihood of the correct answer. **SubSearch** [21] combines subquery-answerability and query-decomposition rewards with final-answer and format rewards. Further details are provided in Appendix F.

[5] For a fair comparison, we evaluate all baselines under the same training and inference settings as OASES. When available, we use released checkpoints for inference; otherwise, we reproduce the methods using their publicly released code within our training and evaluation pipeline, rather than directly quoting reported numbers.

Table 1: Comprehensive QA benchmark results on Qwen2.5-7B-Instruct. **Bold** indicates the best performance. *Impv. vs Best* denotes OASES's absolute improvement over the strongest baseline.

Method	NQ		HotpotQA		2Wiki		Bamboogle		MuSiQue		Avg.	
	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM
Standard												
Direct LLM	17.16	5.98	25.25	17.60	25.80	22.03	14.38	8.00	10.14	2.57	18.55	11.24
Naive RAG [13]	37.26	18.59	36.69	27.45	27.06	21.98	26.87	16.80	9.32	3.52	27.44	17.67
Agentic Search												
Search-o1 [14]	32.43	15.71	44.86	34.07	39.69	32.71	37.56	27.20	20.11	11.09	34.93	24.16
Search-R1 [9]	40.98	20.78	51.24	39.77	47.10	39.71	48.43	36.54	22.50	12.70	42.05	29.90
R1-Searcher [27]	41.15	21.61	55.55	43.31	50.64	43.97	54.62	44.00	27.74	18.16	45.94	34.21
Process Reward												
StepSearch [44]	39.79	19.81	50.02	38.31	44.35	37.59	50.32	39.20	27.50	18.16	42.40	30.61
ReasonRAG [42]	39.65	19.78	41.53	31.24	36.58	31.39	40.53	28.80	16.37	8.94	34.93	24.03
TIPS [37]	40.38	20.30	45.98	35.34	42.95	36.51	46.20	36.00	18.59	10.59	38.82	27.75
SubSearch [21]	41.00	21.00	54.30	43.04	51.71	46.15	51.08	42.40	23.24	15.06	44.27	33.53
OASES (Ours)	**43.62**	**22.60**	**60.62**	**47.17**	**58.14**	**50.48**	**56.99**	**48.00**	**30.73**	**19.03**	**50.02**	**37.46**
Impv. vs Best	+2.47	+0.99	+5.07	+3.86	+6.43	+4.33	+2.37	+4.00	+2.99	+0.87	+4.08	+3.25

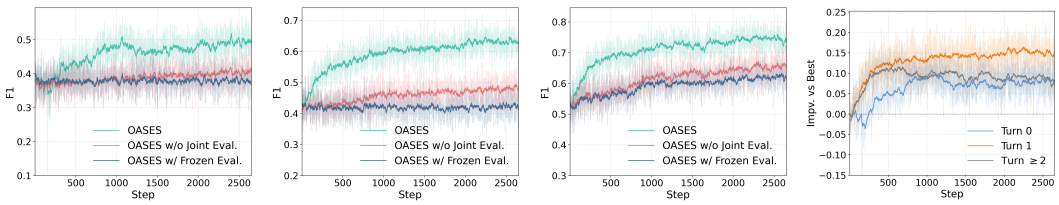
Implementation Details. We implement OASES using `verl`[6] and train it with PPO. Unless otherwise specified, we initialize the policy from Qwen2.5-7B-Instruct [29]; we also evaluate OASES on other model sizes and model families, as reported in Appendix A. For retrieval, we use E5 [33] as the retriever and Wikipedia as the retrieval corpus. We train for 2 epochs with training and validation batch sizes of 256. The actor and critic learning rates are 2×10^{-7} and 1×10^{-6}, respectively. The maximum prompt and response lengths are 3,000 and 2,000 tokens. We set the process-reward weight α to 0.5 by default. Environment details and additional training hyperparameters are provided in Appendix C. After convergence, each query uses an average of 2.76 state-evaluation rollouts, each generating 66.31 tokens. Together they contribute 183.02 tokens per query, or 14.94% of the total generated tokens (Appendix D).

4.2 Main Results Analysis (RQ1)

OASES consistently improves agentic search performance. Table 1 reports the main results on Qwen2.5-7B-Instruct. OASES achieves the best performance across all five benchmarks in both F1 and EM, outperforming standard RL methods, agentic search baselines, and process-supervision methods. On average, OASES improves over the strongest baseline by 4.08 F1 and 3.25 EM, with particularly clear gains on HotpotQA and 2WikiMultihopQA. The gains also extend to smaller models (Appendix A, Table 3). With Qwen2.5-1.5B-Instruct, OASES achieves 42.55 average F1 versus 22.84 for Search-R1, the strongest reported baseline at this size. With Qwen2.5-3B-Instruct, it reaches 45.97 versus 43.29 for SubSearch. These improvements of 19.71 and 2.68 points show benefits beyond the main 7B setting, with different magnitudes across model sizes.

Outcome-aligned state rewards provide more effective process supervision. Compared with outcome-only RL and agentic search baselines, OASES provides denser supervision by decomposing outcome improvement across intermediate search states, helping the policy identify useful reasoning and retrieval actions. It also outperforms prior process-supervision methods based on heuristics, external evaluators, or information gain, whose rewards may be only indirectly tied to the final outcome or rely on likelihood changes under partial evidence. In contrast, OASES evaluates each state via state-conditioned answer generation and scores it with the same verifiable outcome metric used for final answers, making its process rewards more outcome-aligned and adaptive to the evolving policy.

[6] `https://github.com/volcengine/verl`



(a) Turn 0 (b) Turn 1 (c) Turn ≥ 2 (d) Impv. vs Best

Figure 3: Step-wise analysis of the search-state evaluator under different optimization strategies. Panels (a)–(c) show the evaluator's answer F1 across training steps for search states at Turn 0, Turn 1, and Turn ≥ 2, respectively. Panel (d) reports *Impv. vs Best*, defined as OASES's F1 improvement over the stronger non-joint setting at each turn.

4.3 How Joint Optimization Shapes the Search-State Evaluator (RQ2)

To answer **RQ2**, we assess evaluator quality along two dimensions: the accuracy of answers generated from intermediate search states and the association between state-evaluation scores and final-answer quality.

State-evaluation accuracy. We compare three settings: **OASES**, which jointly optimizes search and state evaluation with shared model parameters; **OASES w/o Joint Eval.**, which uses the current policy as the state evaluator but does not train on state-evaluation rollouts; and **OASES w/ Frozen Eval.**, which uses the initial policy as a frozen evaluator throughout training. To make the effect of co-training explicit, we also report *Impv. vs Best*, defined as the F1 improvement of OASES over the stronger non-co-trained alternative at each training step, i.e., the higher F1 between **OASES w/o Joint Eval.** and **OASES w/ Frozen Eval.**.

As shown in Figure 3, OASES consistently achieves the highest state-evaluation F1 across search stages. This indicates that search–evaluation co-training not only improves the search policy, but also strengthens the model's ability to judge whether its accumulated evidence supports the correct answer. The improvement is most pronounced at Turn 1. At Turn 0, the evaluator only observes the original question and has access to little external evidence, leaving limited room for improvement. At Turn ≥ 2, the accumulated evidence is often sufficient for answering, making evaluation easier for all variants. Turn 1 is the most informative regime: the state has acquired some useful evidence but remains incomplete, so accurate evaluation is crucial for assigning credit to the preceding search action. By jointly optimizing the evaluator with the evolving search policy, OASES provides more reliable feedback in this intermediate regime, while the non-joint and frozen evaluators suffer from weaker adaptation to the policy's changing search behavior.

State–outcome correlation. We compare OASES with the current policy without evaluation optimization and frozen Qwen-7B and GPT-4o evaluators. For each setting, we compute the Pearson correlation between state-conditioned answer F1, γ_t, and final-answer F1, γ_{out}, along corresponding search trajectories. Correlations are computed separately for Turns 0, 1, and 2 across three consecutive 1,000-step intervals, corresponding to the early, middle, and late stages of training. Turns 3 and later are omitted because they occur in relatively few long-tail trajectories.

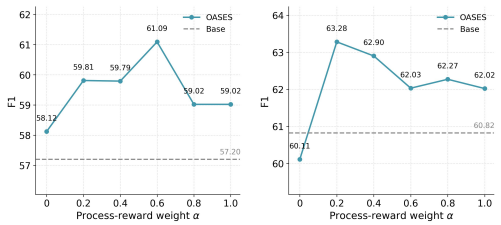
OASES has the highest correlation in all nine turn–stage combinations. In the late stage, its correlations at Turns 0, 1, and 2 are 0.4688, 0.6736, and 0.9371, respectively, compared with 0.4126, 0.5431, and 0.8231 for the current policy without evaluation optimization. OASES's correlations increase throughout training at each turn. Later states better predict final answers as evidence accumulates. Appendix K.2 reports the complete results.

4.4 Ablation Study on Search–Evaluation Co-Training and Process Rewards (RQ2 & RQ3)

Table 2 reports ablation results on three representative benchmarks. **OASES** is the full framework. Under **OASES w/o joint opt.**, we disable search–evaluation co-training and evaluate three types of state evaluator: the current policy without evaluation training, frozen pretrained models (Qwen and GPT-4o), and an offline SFT model. The offline SFT evaluator is fine-tuned on state–answer pairs and then frozen during actor training (Appendix K.1). **OASES w/o proc. reward** removes the

Table 2: Ablation results on three representative benchmarks. Δ denotes the performance difference compared with the full OASES model.

Method	HotpotQA		2Wiki		MuSiQue	
	F1	Δ	F1	Δ	F1	Δ
OASES	**60.62**	-	**59.69**	-	**30.73**	-
w/o joint opt.						
Policy model	57.16	-3.46	57.81	-1.88	27.46	-3.27
Frozen Qwen-7B	57.65	-2.97	57.93	-1.76	27.88	-2.85
Frozen Qwen-14B	57.57	-3.05	58.85	-0.84	26.93	-3.80
Frozen Qwen-32B	57.11	-3.51	58.51	-1.18	28.05	-2.68
Frozen GPT-4o	57.38	-3.24	58.82	-0.87	28.24	-2.49
Offline SFT evaluator	57.94	-2.68	58.10	-1.59	–	–
w/o proc. reward	58.80	-1.82	58.64	-1.05	27.62	-3.11
w/o joint opt. & proc. reward	58.42	-2.20	58.00	-1.69	28.79	-1.94



(a) 7B model on F1 (b) 14B model on F1

Figure 4: Effect of the process-reward weight α across model sizes. Panels (a) and (b) show F1 results for 7B and 14B. **Base** removes both process rewards and joint optimization.

state-level process rewards by setting $\alpha = 0$. **OASES w/o joint opt. & proc. reward** removes both co-training and process rewards, corresponding to the base variant.

Frozen or non-co-trained evaluators are insufficient. OASES performs best across all three benchmarks, showing the importance of both search–evaluation co-training and outcome-aligned process rewards. Removing joint optimization consistently degrades performance, even with stronger frozen evaluators such as GPT-4o, suggesting that evaluator scale alone cannot replace adaptation to the evolving search policy. The offline SFT evaluator obtains 57.94 F1 on HotpotQA and 58.10 on 2Wiki, below OASES by 2.68 and 1.59 points, respectively, showing that the gap persists with task-specific evaluator training. Using the policy model as an evaluator without co-training is also suboptimal, since RL may specialize the model for search while weakening standalone state evaluation. In contrast, OASES jointly optimizes search and evaluation with shared parameters, keeping process supervision aligned with the evolving policy. The drop after removing process rewards further shows that state-level feedback provides useful supervision beyond terminal rewards. Appendix H illustrates a failure after search-only training: the model retrieves evidence containing "reception room" and produces that correct final answer, yet generates incorrect answers when evaluating the intermediate search states. This discrepancy highlights the need to train state evaluation alongside search.

4.5 Effect of the Process-Reward Weight (RQ4)

Figure 4 shows the effect of the process-reward weight α on F1 across model sizes. The experiment is conducted on 2WikiMultihopQA with both 7B and 14B models. **Base** denotes the variant without both outcome-aligned process rewards and search–evaluation co-training. The setting $\alpha = 0$ keeps co-training but removes the contribution of process rewards, while positive values of α assign different weights to state-level outcome improvements.

Outcome-aligned feedback is consistently beneficial. The results show that **Base** and $\alpha = 0$ are generally worse than using a positive process-reward weight, confirming the importance of dense outcome-aligned feedback. For the 7B model, performance peaks at $\alpha = 0.6$, while for the 14B model, the best result is achieved at $\alpha = 0.2$. Although the optimal value differs across model sizes, positive values of α consistently outperform the no-process-reward setting, showing that OASES is robust to the choice of α within a reasonable range.

5 Conclusion

We presented OASES, an outcome-aligned search–evaluation co-training framework for agentic search. OASES uses the same verifiable outcome signal for state-level and final-answer evaluation, converting outcome improvements across search states into dense process rewards. By co-training a single policy for search and state evaluation, OASES provides outcome-aligned credit assignment while reducing evaluator–policy mismatch. Experiments on multi-hop QA benchmarks show that OASES consistently outperforms strong RL baselines, and ablations confirm the importance of outcome-aligned process rewards and search–evaluation co-training.

Acknowledgments and Disclosure of Funding

This research was sponsored by the Natural Science Foundation of China (62572475, 61902209, 62377044), Intelligent Social Governance Platform, Major Innovation Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China, the Research Funds of Renmin University of China (22XNKJ15), and Beijing Nova Program.

Competing interests. The authors declare no competing interests.

References

[1] Yiqun Chen, Lingyong Yan, Weiwei Sun, Xinyu Ma, Yi Zhang, Shuaiqiang Wang, Dawei Yin, Yiming Yang, and Jiaxin Mao. Improving retrieval-augmented generation through multi-agent reinforcement learning. *arXiv preprint arXiv:2501.15228*, 2025.

[2] Yiqun Chen, Erhan Zhang, Lingyong Yan, Shuaiqiang Wang, Jizhou Huang, Dawei Yin, and Jiaxin Mao. Mao-arag: Multi-agent orchestration for adaptive retrieval-augmented generation. *arXiv preprint arXiv:2508.01005*, 2025.

[3] Yiqun Chen, Lingyong Yan, Zixuan Yang, Erhan Zhang, Jiashu Zhao, Shuaiqiang Wang, Dawei Yin, and Jiaxin Mao. Beyond monolithic architectures: A multi-agent search and knowledge optimization framework for agentic search. *arXiv preprint arXiv:2601.04703*, 2026.

[4] Yiqun Chen, Erhan Zhang, Tianyi Hu, Shijie Wang, Zixuan Yang, Meizhi Zhong, Xiaochi Wei, Yan Gao, Yi Wu, Yao Hu, et al. Jade: Bridging the strategic-operational gap in dynamic agentic rag. *arXiv preprint arXiv:2601.21916*, 2026.

[5] Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025.

[6] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[7] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*, 2020.

[8] Senkang Hu, Yong Dai, Yuzhi Zhao, Yihang Tao, Yu Guo, Zhengru Fang, Sam Tak Wu Kwong, and Yuguang Fang. Optimizing agentic reasoning with retrieval via synthetic semantic information gain reward. *arXiv preprint arXiv:2602.00845*, 2026.

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

[10] Zixuan Ke, Weize Kong, Cheng Li, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. Bridging the preference gap between retrievers and llms. *arXiv preprint arXiv:2401.06954*, 2024.

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

[12] Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

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

[14] Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5420–5438, 2025.

[15] Xinze Li, Sen Mei, Zhenghao Liu, Yukun Yan, Shuo Wang, Shi Yu, Zheni Zeng, Hao Chen, Ge Yu, Zhiyuan Liu, et al. Rag-ddr: Optimizing retrieval-augmented generation using differentiable data rewards. *arXiv preprint arXiv:2410.13509*, 2024.

[16] Zhicong Li, Jiahao Wang, Zhishu Jiang, Hangyu Mao, Zhongxia Chen, Jiazhen Du, Yuanxing Zhang, Fuzheng Zhang, Di Zhang, and Yong Liu. Dmqr-rag: Diverse multi-query rewriting for rag. *arXiv preprint arXiv:2411.13154*, 2024.

[17] Zihan Liang, Yufei Ma, Ben Chen, Zhipeng Qian, Huangyu Dai, Lingtao Mao, Xuxin Zhang, Chenyi Lei, and Wenwu Ou. Ig-search: Step-level information gain rewards for search-augmented reasoning. *arXiv preprint arXiv:2604.15148*, 2026.

[18] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The twelfth international conference on learning representations*, 2023.

[19] Minhua Lin, Zongyu Wu, Zhichao Xu, Hui Liu, Xianfeng Tang, Qi He, Charu Aggarwal, Xiang Zhang, and Suhang Wang. A comprehensive survey on reinforcement learning-based agentic search: Foundations, roles, optimizations, evaluations, and applications. *arXiv preprint arXiv:2510.16724*, 2025.

[20] Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. Query rewriting for retrieval-augmented large language models. *arXiv preprint arXiv:2305.14283*, 2023.

[21] Roxana Petcu, Evangelos Kanoulas, and Maarten de Rijke. Subsearch: Intermediate rewards for unsupervised guided reasoning in complex retrieval. *arXiv preprint arXiv:2604.07415*, 2026.

[22] Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*, 2022.

[23] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36:68539–68551, 2023.

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

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

[26] Yaorui Shi, Sihang Li, Chang Wu, Zhiyuan Liu, Junfeng Fang, Hengxing Cai, An Zhang, and Xiang Wang. Search and refine during think: Facilitating knowledge refinement for improved retrieval-augmented reasoning. *arXiv preprint arXiv:2505.11277*, 2025.

[27] Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*, 2025.

[28] Weiting Tan, Xinghua Qu, Ming Tu, Meng Ge, Andy T Liu, Philipp Koehn, and Lu Lu. Process-supervised reinforcement learning for interactive multimodal tool-use agents. *arXiv preprint arXiv:2509.14480*, 2025.

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

[30] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

[31] Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, and Zhenzhe Ying. Information gain-based policy optimization: A simple and effective approach for multi-turn llm agents. *arXiv preprint arXiv:2510.14967*, 2025.

[32] Junzhe Wang, Zhiheng Xi, Hao Luo, Shihan Dou, Tao Gui, Qi Zhang, et al. Enhancing llm-based search agents via contribution weighted group relative policy optimization. *arXiv preprint arXiv:2604.14267*, 2026.

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

[34] Xinglin Wang, Jiayi Shi, Shaoxiong Feng, Peiwen Yuan, Yiwei Li, Yueqi Zhang, Chuyi Tan, Ji Zhang, Boyuan Pan, Yao Hu, et al. Do not waste your rollouts: Recycling search experience for efficient test-time scaling. *arXiv preprint arXiv:2601.21684*, 2026.

[35] Zhao Wang, Ziliang Zhao, and Zhicheng Dou. Prorag: Process-supervised reinforcement learning for retrieval-augmented generation. *arXiv preprint arXiv:2601.21912*, 2026.

[36] Tianle Xia, Ming Xu, Lingxiang Hu, Yiding Sun, Wenwei Li, Linfang Shang, Liqun Liu, Peng Shu, Huan Yu, and Jie Jiang. Search-p1: Path-centric reward shaping for stable and efficient agentic rag training. *arXiv preprint arXiv:2602.22576*, 2026.

[37] Yutao Xie, Nathaniel Thomas, Nicklas Hansen, Yang Fu, Li Erran Li, and Xiaolong Wang. Tips: Turn-level information-potential reward shaping for search-augmented llms. In *The Fourteenth International Conference on Learning Representations*, 2026.

[38] Peiran Xu, Zhuohao Li, Xiaoying Xing, Guannan Zhang, Debiao Li, and Kunyu Shi. Principle process reward for search agents.

[39] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[40] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*, 2018.

[41] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 2022.

[42] Wenlin Zhang, Xiangyang Li, Kuicai Dong, Yichao Wang, Pengyue Jia, Xiaopeng Li, Yingyi Zhang, Derong Xu, Zhaocheng Du, Huifeng Guo, et al. Process vs. outcome reward: Which is better for agentic rag reinforcement learning. *arXiv preprint arXiv:2505.14069*, 2025.

[43] Qingfei Zhao, Ruobing Wang, Dingling Xu, Daren Zha, and Limin Liu. R-search: Empowering llm reasoning with search via multi-reward reinforcement learning. *arXiv preprint arXiv:2506.04185*, 2025.

[44] Xuhui Zheng, Kang An, Ziliang Wang, Yuhang Wang, and Yichao Wu. Stepsearch: Igniting llms search ability via step-wise proximal policy optimization. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 21816–21841, 2025.

A Additional Results across Model Sizes and Model Families

We provide additional experiments to complement the main Qwen2.5-7B-Instruct results. Table 3 evaluates OASES on smaller Qwen2.5-Instruct models. OASES consistently improves over agentic search baselines on both Qwen2.5-1.5B-Instruct and Qwen2.5-3B-Instruct, showing that outcome-aligned search–evaluation co-training is also effective under smaller model scales. The gains are especially pronounced for Qwen2.5-1.5B-Instruct, where OASES improves the average F1 from 22.84 with Search-R1 to 42.55.

Table 4 further reports results on Qwen2.5 and Qwen3 base models before OASES training. These results provide an additional reference for understanding how model family and scale affect agentic search performance under the base-model setting. Qwen3-8B-Base achieves the strongest performance among the evaluated base models, suggesting that stronger base capabilities can provide a better starting point for agentic search training.

Table 3: Additional QA benchmark results on smaller Qwen2.5-Instruct models. **Bold** indicates best performance within each model-size group.

Method	NQ		HotpotQA		2Wiki		Bamboogle		MuSiQue		Avg.	
	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM
Qwen2.5-1.5B-Instruct												
Search-o1 [14]	1.23	0.69	4.51	3.43	4.82	4.13	4.83	3.20	1.36	0.54	3.35	2.40
Search-R1 [9]	31.72	15.96	31.24	22.67	25.73	21.60	15.11	6.73	10.40	3.89	22.84	14.17
SubSearch [21]	17.36	6.81	22.08	15.33	25.72	21.88	28.04	18.40	8.16	2.36	20.27	12.96
OASES (Ours)	**37.73**	**18.75**	**51.39**	**38.69**	**54.41**	**46.56**	**46.41**	**37.60**	**22.81**	**12.58**	**42.55**	**30.84**
Qwen2.5-3B-Instruct												
Search-o1 [14]	26.36	12.22	33.27	24.34	28.46	23.40	38.66	28.80	13.34	5.50	28.02	18.85
Search-R1 [9]	35.70	17.95	40.61	31.57	41.83	36.74	31.12	21.15	17.07	9.39	33.27	23.36
SubSearch [21]	39.05	19.78	50.73	39.41	51.00	45.09	**51.55**	39.20	24.12	**15.06**	43.29	31.71
OASES (Ours)	**40.78**	**20.19**	**55.31**	**42.81**	**57.07**	**50.46**	51.53	**40.80**	**25.18**	14.65	**45.97**	**33.78**

Table 4: QA benchmark results on Qwen2.5 and Qwen3 base models with different model sizes.

Method	NQ		HotpotQA		2Wiki		Bamboogle		MuSiQue		Avg.	
	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM
Qwen2.5-3B-Base	20.72	7.45	30.24	19.82	33.82	28.55	29.38	17.60	13.14	2.90	25.46	15.26
Qwen3-4B-Base	20.89	6.93	30.80	19.86	34.36	29.17	17.55	8.00	13.79	2.57	23.48	13.31
Qwen2.5-7B-Base	43.31	22.44	59.79	46.29	60.34	54.60	61.06	46.40	28.99	17.17	50.70	37.38
Qwen3-8B-Base	44.08	22.94	60.79	47.52	61.73	55.66	61.15	49.60	31.92	20.56	51.93	39.26

B Analysis of Outcome-Aligned Process Rewards

We provide an interpretation of OASES from the perspective of reward redistribution. Existing process-reward methods often introduce auxiliary reward sources whose optimization targets may deviate from the final task objective. In contrast, OASES derives dense process rewards by measuring how much each search step improves the model's ability to answer the original question under the same verifiable objective used for final-answer evaluation. Therefore, the process rewards can be viewed as an outcome-aligned decomposition of answerability improvements across intermediate search states. The accumulated process reward follows the telescoping identity in Eq. (8). Thus, OASES's process reward is not an independent reward source, but a telescoping redistribution of the improvement from the initial search state to the final search state. When the final search state provides sufficient evidence for answering the question, γ_T is expected to correlate with the terminal outcome score γ_{out}, and the trajectory-level reward can be interpreted as

$$G^{\text{OASES}}(\tau) = \gamma_{\text{out}} + \sum_{t=1}^{T} r_t^{\text{proc}} = \gamma_{\text{out}} + \alpha(\gamma_T - \gamma_0). \tag{12}$$

Under the idealized case where the state-evaluation score at the final search state matches the final-answer score, i.e., $\gamma_T \approx \gamma_{\text{out}}$, this becomes

$$G^{\text{OASES}}(\tau) \approx \gamma_{\text{out}} + \alpha(\gamma_{\text{out}} - \gamma_0). \tag{13}$$

In the special case where $\alpha = 1$, this further reduces to

$$G^{\text{OASES}}(\tau) \approx 2\gamma_{\text{out}} - \gamma_0. \tag{14}$$

This analysis suggests that OASES encourages trajectories whose search process improves the answerability of intermediate states under the same verifiable scoring function used for final answers. Rather than introducing a separate proxy target, the dense process reward redistributes state-level answerability improvements back to intermediate search decisions, providing step-level supervision that is closely tied to the final task objective.

At Turn 0, state evaluation is question-only: γ_0 records answerability before any retrieval. The Direct LLM row in Table 1 provides a related question-only reference, while Figure 3 tracks Turn 0 evaluation during training.

C Implementation and Environment Details

Hardware and software environment. All training experiments are conducted on a single node with 8 NVIDIA H800 GPUs. Each GPU has approximately 80GB memory. The NVIDIA driver version is 570.148.08, and the CUDA version reported by `nvidia-smi` is 12.8. We use vLLM as the rollout backend for agentic search generation.

Training setup. We train OASES with PPO on the combined HotpotQA and 2WikiMultihopQA training set. Unless otherwise specified, the validation set is selected from the target evaluation benchmark.

Model and optimization hyperparameters. We use Qwen-series base models as the policy and critic backbone. The PPO mini-batch size is 32, with a PPO micro-batch size of 2 per GPU. We enable gradient checkpointing and remove-padding optimization for both the actor and critic models. We do not use KL loss in the actor objective or KL penalty in the reward.

Rollout and retrieval configuration. We use the `vllm_agentic` rollout engine with tensor parallel size 1 and GPU memory utilization set to 0.6. The agent uses an external search tool for multi-turn retrieval. In our implementation, the search service is deployed as multiple parallel retrieval endpoints, and the agent queries these endpoints during rollout generation.

Reward configuration. The final outcome reward is the F1 score against the ground-truth answer. Each process reward is the difference between consecutive state-evaluation F1 scores, scaled by α, as defined in Eq. (6). For ablation experiments on the process-reward weight, we vary α as described in Figure 4.

D Efficiency Analysis of State-Evaluation Rollouts

OASES introduces additional state-evaluation rollouts during training. To quantify the resulting overhead, we report the generated-token cost of the main search rollout and the state-evaluation rollouts after convergence. We focus on generated tokens because they directly reflect the decoding cost during rollout collection. As shown in Table 5, although each query requires an average of 2.76 state-evaluation rollouts, each evaluation rollout is short, generating only 66.31 tokens on average. Consequently, state-evaluation rollouts account for only 14.94% of the generated-token cost per query, while the main search rollout accounts for 85.06%. This indicates that the additional evaluation rollouts provide dense training supervision with relatively lightweight decoding overhead.

Table 5: Generated-token cost of search and state-evaluation rollouts after convergence.

Component	Avg. # Rollouts per Query	Avg. Generated Tokens per Rollout	Avg. Generated Tokens per Query	Token Share
Search rollout	1.00	1041.76	1041.76	85.06%
State-evaluation rollouts	2.76	66.31	183.02	14.94%
Total	3.76	–	1224.78	100.00%

Table 6: Prompt templates used in OASES, including the main search prompt and two search-state evaluation prompts for question-only and context-aware evaluation.

Main Search Prompt

```
You are an intelligent Research Agent designed to answer questions
by retrieving information from the internet.

## Operational Logic
1. Plan: Analyze the user's question. If the answer requires
   multiple steps, break it down.
2. Search: Use the `<search>` tag to query the search engine. Keep
   queries specific and efficient.
3. Observe: Wait for the `<result>` output provided by the system.
4. Iterate: Analyze the search results in a `<think>` block.
   - If information is missing, formulate a new search query based
     on what you found.
   - If the information is sufficient, proceed to answer.
5. Answer: Output the final answer in an `<answer>` tag. Keep it
   concise and direct.

## Response Format
- Thinking Process: <think> ... </think>
- Search Action: <search> ... </search>
- Final Answer: <answer> ... </answer>

## Constraints
- Stop after Search: Do not generate `<result>` tags yourself.
  Wait for the system.
- Search Limit: There is a maximum limit on the number of
  search rounds. If the limit is reached, stop searching and
  output the final answer using available information.
```

Search-state Evaluation Prompts

Question-Only Evaluation

```
You are an expert AI assistant
capable of answering questions
precisely.
Answer the user's question based
on your internal knowledge.

You must provide a concise answer
enclosed within specific tags as
follows:
<eval> The Answer </eval>
```

Context-Aware Evaluation

```
You are an expert AI assistant
capable of answering questions
precisely.
Answer the user's question
using the provided Context and
your internal knowledge.

You must provide a concise answer
enclosed within specific tags as
follows:
<eval> The Answer </eval>
```

E Prompt Templates

Table 6 shows the prompt templates used in OASES. The main search prompt guides the agent to perform multi-step search, while the state-evaluation prompts assess whether each search state supports answering the original question. For state evaluation, we use a question-only prompt for the initial state and a context-aware prompt for later states with accumulated search observations.

F Baseline Details

This section provides additional details about the baselines used in our experiments.

Direct LLM. Direct LLM is a non-retrieval baseline where the model directly answers the question using only its parametric knowledge. It does not access external documents or invoke search tools.

Naive RAG. Naive RAG retrieves relevant passages from the corpus and appends them to the model input before answer generation. Unlike agentic search methods, it follows a static retrieve-then-read pipeline and does not allow the model to iteratively refine queries, inspect intermediate evidence, or decide whether additional search is needed [13].

Search-o1. Search-o1 is an agentic search-enhanced reasoning framework that augments large reasoning models with dynamic retrieval during the reasoning process. It introduces an agentic RAG workflow that allows the model to retrieve external knowledge when encountering uncertain knowledge points, and uses a Reason-in-Documents module to analyze retrieved documents before injecting information back into the reasoning chain [14]. Search-o1 is included as a representative inference-time agentic search baseline.

Search-R1. Search-R1 trains language models to interleave reasoning and search through reinforcement learning. The model learns to autonomously generate search queries during step-by-step reasoning and interact with a real-time retrieval system. Its training relies on outcome-based rewards and retrieved-token masking for stable RL optimization [9].

R1-Searcher. R1-Searcher is a two-stage outcome-based RL framework designed to incentivize search capability in LLMs. It first teaches the model when and how to invoke external search, and then optimizes the model to use retrieved information for final answer generation [27].

StepSearch. StepSearch trains search-augmented LLMs with step-wise PPO and adds search-step rewards to the final answer reward [44]. GPT-4o is used to construct subquestion–answer pairs and search queries, which are then treated as gold supervision for reward computation. Based on this supervision, the process reward is derived from retrieval-side signals: each search step is rewarded by the marginal improvement in retrieved-document coverage over gold information documents, measured with TF–IDF cosine similarity, and penalized for retrieving redundant documents.

ReasonRAG. ReasonRAG studies process-supervised training for agentic RAG [42]. Instead of relying only on final-answer correctness, it constructs process-supervised rollouts with Monte Carlo Tree Search (MCTS), obtaining preference signals over intermediate RAG actions such as query generation, evidence extraction, and answer generation. These process-level preferences are then optimized with DPO to teach the model better search and evidence-use strategies.

TIPS. TIPS proposes turn-level information-potential reward shaping for search-augmented LLMs [37]. Its reward is computed from the increase in the teacher model's likelihood of the correct answer after each reasoning–tool-call turn. Specifically, TIPS treats this likelihood as an information potential and assigns a turn-level shaping reward according to how much the current turn increases that potential.

SubSearch. SubSearch trains search agents with intrinsic intermediate rewards for retrieval tasks [21]. Its process rewards come from two internally computed signals. First, a subquery answerability reward measures how well the top-k retrieved documents address each generated subquery, using embedding cosine similarity between the subquery and retrieved documents. Second, a decomposition reward evaluates whether the generated subqueries preserve the semantic coverage of the parent query while remaining distinct from each other. These intermediate rewards are combined with the final exact-match answer reward and a format reward.

G Limitations

Increased training cost. OASES introduces additional computational overhead during training. Besides generating the main search trajectory, it performs state-conditioned evaluation rollouts for intermediate search states. As a result, rollout collection becomes slower and the total training time increases compared with methods that rely only on sparse outcome rewards.

Limited applicability to search scenarios. OASES is currently designed for agentic search tasks, where intermediate states can be naturally defined by the accumulated retrieved evidence. Therefore, it may not directly apply to tasks without external search, clearly identifiable intermediate states, or a meaningful notion of evidence accumulation.

Dependence on outcome evaluation. The process supervision in OASES is derived from outcome improvements over search states. Thus, its effectiveness depends on the reliability of the final

Table 7: Negative case study showing degraded search-state evaluation after search-only training.

Query: What was the first use of the East Sitting Hall at the official residence and workplace of the President of the United States?
Ground Truth: reception room

Turn	Component	Output / Search-State Evaluation
Turn 0	Eval.	**Eval:** `<eval> Congress held its first full session </eval>` **Score:** $\gamma_0 = 0.000$ **(✗ unsupported)**
Turn 1	Search	`<search> official residence and workplace of the President of the United States </search>`
	Result	The official residence and workplace of the U.S. President is the **White House**.
	Eval.	**Eval:** `<eval> 1792 </eval>` **Score:** $\gamma_1 = 0.000$ **(✗ distracted by irrelevant evidence)** **Source:** *"Construction took place between 1792 and 1800 using Aquia..."*
Turn 2	Search	`<search> first use of East Sitting Hall in White House </search>`
	Result	The East Sitting Hall was first used as a **reception room for guests of the president**.
	Eval.	**Eval:** `<eval> 2,161,000 </eval>` **Score:** $\gamma_2 = 0.000$
Turn 3	Answer	**Answer:** `<answer> reception room</answer>` **Outcome:** $\gamma_{\text{out}} = 1.000$ **(✓ correct)**

Failure Pattern: The model obtains a correct final answer, but all state-conditioned evaluations are incorrect.

outcome evaluator. If the outcome reward is noisy, biased, or incomplete, the induced intermediate rewards may also be affected.

H Failure Case of Search-Only Training

Table 7 shows a failure case after search-only training. Although the model eventually produces the correct final answer, its state-conditioned evaluations are incorrect at every intermediate state. At Turn 0, the evaluator relies on unsupported internal knowledge; after the first search, it is distracted by irrelevant construction-year evidence; after the second search, it produces an unrelated answer despite having retrieved the correct evidence. This pattern suggests that optimizing only the search trajectory can improve final-answer behavior while leaving state evaluation unreliable. Such failures motivate search–evaluation co-training in OASES, which jointly trains the policy to search and to evaluate whether its accumulated information supports the final answer.

I Discussion: Why Joint Search–Evaluation Optimization?

Likelihood-based information-gain methods turn intermediate evaluation into a direct conditional-answering task, which differs from multi-turn agentic search [31, 17, 8]. In agentic search, the model decides whether to continue searching or stop and answer, and the final answer is generated after the policy's own reasoning and stopping decision. By contrast, likelihood-based scoring asks the model to score the ground-truth answer from a fixed intermediate state, often with a manually appended prefix such as "Now there is enough information to answer" [31]. Such prefixes inject an artificial stopping decision, while omitting them creates a context that differs from the model's usual ReAct-style answer-generation format.

More importantly, likelihood-based scoring is difficult to jointly optimize as an explicit evaluation behavior. Since the score is computed from the likelihood of the gold answer, directly training the evaluator to increase this likelihood under intermediate search states would expose the evaluator to the target answer before sufficient evidence is available. This creates a shortcut: the evaluator can improve the scoring objective by memorizing or fitting the gold answer, rather than learning to judge whether the accumulated evidence actually supports answering the question. As a result, the optimized evaluator may assign high scores to under-informed states, weakening the intended role of information gain as a measure of evidence usefulness. In addition, likelihood scoring provides only a scalar probability of a known answer, rather than a generated state-conditioned answer whose quality can be verified by the outcome metric. Therefore, it does not naturally produce trainable evaluation rollouts that match the policy's answer-generation behavior.

Table 8: Positive case study of outcome-aligned process reward in OASES.

Query: When is the director of film *The Opposite of Sex*'s birthday?
Ground Truth: April 14, 1955

Turn	Component	Output / Search-State Evaluation
Turn 0	Evaluation of s_0	**Eval Answer:** `<eval> April 20, 1962 </eval>` **Score:** $\gamma_0 = 0.333$
Turn 1	Think	`<thinking> I need to answer this in two steps. First, I must identify the director of the film "The Opposite of Sex". Second, I need to find the birth date of that specific director. </thinking>`
	Search	`<search> director of film The Opposite of Sex </search>`
	Result	`<result>` The Opposite of Sex is a 1998 American romantic comedy film written and directed by Don Roos and starring Christina Ricci, Martin Donovan and Lisa Kudrow. Don Roos is well known for his work as a screenwriter... `</result>`
	Evaluation of s_1	**Eval Answer:** `<eval> April 14, 1950 </eval>` **Score:** $\gamma_1 = 0.667$ **Search-State Gain:** $\gamma_1 - \gamma_0 = +0.334$
Turn 2	Think	`<thinking> The director of the film "The Opposite of Sex" is Don Roos. Now I need to find his birth date. </thinking>`
	Search	`<search> Don Roos birth date </search>`
	Result	`<result>` Don Roos Donald Paul "Don" Roos (born April 14, 1955) is an American screenwriter and film director. Roos was born in New York. He attended the University of Notre Dame in Indiana and... `</result>`
	Evaluation of s_2	**Eval Answer:** `<eval> April 14, 1955 </eval>` **Score:** $\gamma_2 = 1.000$ **Search-State Gain:** $\gamma_2 - \gamma_1 = +0.333$
Turn 3	Think	`<thinking> The search results confirm that Don Roos was born on April 14, 1955. I have the final answer. </thinking>`
	Answer	`<answer> April 14, 1955 </answer>` **Outcome Reward:** $r^{\text{out}} = \gamma_{\text{out}} = 1.000$

Final Prediction: `April 14, 1955` **(✓ Matches Ground Truth)**

As RL training progresses, this mismatch can grow because the policy may specialize in answering only after its own search and reasoning trajectory, whereas likelihood-based evaluation continues to score gold-answer probability under artificial or partial contexts. Consequently, the estimated likelihood gain may not accurately reflect whether the current state helps the policy produce a better final answer.

OASES instead evaluates each intermediate state by generating a state-conditioned answer with the same policy and scoring it using the verifiable outcome signal (e.g. the F1 or EM scores). This turns intermediate evaluation into an explicit generation behavior rather than a likelihood-scoring heuristic. Given the evidence accumulated so far, the policy directly tests whether the current state can support a correct answer, and the resulting score is computed in the same output space as the final task objective. Therefore, the induced process reward measures outcome improvement across search states, rather than a proxy change in answer likelihood.

More importantly, OASES jointly optimizes search and state evaluation with shared parameters. This allows the model to learn two complementary abilities at the same time: acquiring useful evidence through multi-turn search and judging whether the evidence collected so far is sufficient for answering. As the search policy evolves during RL, the evaluator evolves with it, keeping intermediate state scores compatible with the policy's current generation behavior. As a result, OASES provides process rewards that are both outcome-aligned and policy-adaptive, leading to more reliable credit assignment for intermediate search decisions. Without such joint optimization, search-only RL may make the policy increasingly specialized for final-answer generation after complete search trajectories, while weakening its ability to answer from partial states; this evaluator forgetting can make state scores unreliable and cause process-reward supervision to fail.

Table 9: Negative case study of outcome-aligned process reward in OASES.

Query: Where did the director of film *Legally Blonde* study?
Ground Truth: California Institute of the Arts

Turn	Component	Output / Search-State Evaluation
Turn 0	Evaluation of s_0	**Eval Answer:** `<eval> University of Southern California (USC) </eval>` **Score:** $\gamma_0 = 0.444$
Turn 1	Think	`<thinking> I need to answer this in two steps. First, I must identify the director of the film "Legally Blonde". Second, I need to find the educational background of that specific director. </thinking>`
	Search	`<search> director of film Legally Blondes </search>`
	Result	`<result>` Legally Blonde is a 2001 American comedy film based on Amanda Brown's novel of the same name. It was directed by Robert Luketic, scripted by Karen McCullah Lutz and Kirsten Smith, and stars Reese Witherspoon, Luke Wilson, Selma Blair, Matthew Davis, Victor Garber... `</result>`
	Evaluation of s_1	**Eval Answer:** `<eval> University of Melbourne </eval>` **Score:** $\gamma_1 = 0.286$ **Search-State Gain:** $\gamma_1 - \gamma_0 = -0.158$
Turn 2	Think	`<thinking> The director of the film "Legally Blonde" is Robert Luketic. Now I need to find out where he studied. </thinking>`
	Search	`<search> Robert Luketic education </search>`
	Result	`<result>` The retrieved results do not match Robert Luketic. They instead mention unrelated entities such as Robert Lukens and Robert S. Lasnik. One snippet states that Robert Lukens received his bachelor's degree from Temple University, while another describes Robert S. Lasnik's education. `</result>`
	Evaluation of s_2	**Eval Answer:** `<eval> Temple University </eval>` **Score:** $\gamma_2 = 0.000$ **Search-State Gain:** $\gamma_2 - \gamma_1 = -0.286$
Turn 3	Think	`<thinking> The search results indicate that Robert Luketic studied at the University of California, Los Angeles (UCLA). </thinking>`
	Answer	`<answer> UCLA </answer>` **Outcome Reward:** $r^{\text{out}} = \gamma_{\text{out}} = 0.000$
Final Prediction: `UCLA` **(✗ Does Not Match Ground Truth)**		

J Case Studies of Outcome-Aligned Process Rewards

We provide two case studies to illustrate how OASES assigns stepwise process rewards from outcome-aligned improvements across search states. For clarity, Tables 8 and 9 report the unscaled score gain $\gamma_t - \gamma_{t-1}$ at each search turn. During training, this gain is multiplied by the process-reward weight α to obtain the actual process reward $r_t^{\text{proc}} = \alpha(\gamma_t - \gamma_{t-1})$.

Positive case. Table 8 shows a successful two-hop search trajectory. The query asks for the birthday of the director of *The Opposite of Sex*. At the initial state s_0, the evaluator only sees the question and predicts an incorrect date, yielding $\gamma_0 = 0.333$. After the first search, the trajectory identifies the bridge entity Don Roos as the director of the film. Although the target birth date has not yet been retrieved, this search state becomes more informative, and the evaluator answer improves from `April 20, 1962` to `April 14, 1950`, increasing the score to $\gamma_1 = 0.667$. This produces a positive search-state gain of $\gamma_1 - \gamma_0 = +0.334$, rewarding the first search turn for finding the key entity.

The second search directly retrieves Don Roos's birth date. With this evidence available in s_2, the evaluator outputs the exact answer `April 14, 1955`, and the score reaches $\gamma_2 = 1.000$. This yields another positive search-state gain of $\gamma_2 - \gamma_1 = +0.333$. The final answer also matches the ground truth, so the main trajectory receives the outcome reward $r^{\text{out}} = \gamma_{\text{out}} = 1.000$. This case illustrates the intended behavior of OASES: search turns are rewarded according to how much they improve

the answerability of the current search state. The first turn discovers the bridge entity, while the second turn retrieves the target attribute.

Negative case. Table 9 shows a contrasting case where search introduces misleading evidence. The query asks where the director of *Legally Blonde* studied, whose ground-truth answer is *California Institute of the Arts*. At the initial state s_0, the evaluator predicts `University of Southern California (USC)`, obtaining $\gamma_0 = 0.444$ due to partial token overlap with the ground truth. The first search correctly identifies Robert Luketic as the director, but the evaluator still predicts an incorrect institution, `University of Melbourne`. The score drops to $\gamma_1 = 0.286$, giving a negative search-state gain of $\gamma_1 - \gamma_0 = -0.158$.

The second search is more harmful: instead of retrieving Robert Luketic's education, it returns snippets about unrelated entities such as Robert Lukens and Robert S. Lasnik. The evaluator is then misled to answer `Temple University`, and the score decreases further to $\gamma_2 = 0.000$. This produces another negative search-state gain of $\gamma_2 - \gamma_1 = -0.286$. The final model answer is `UCLA`, which does not match the ground truth, so the trajectory receives no outcome reward. This case shows that OASES does not blindly reward additional search. When a search action makes the intermediate state less answerable by introducing irrelevant or misleading evidence, the corresponding score gain becomes negative, resulting in negative process feedback for that search turn.

Summary. Together, the two cases demonstrate that OASES provides outcome-aligned credit assignment for agentic search. Search steps that uncover bridge entities or target attributes receive positive process rewards when they improve state answerability, while misleading steps receive negative feedback when they degrade it. This dense feedback complements the sparse final outcome reward and helps the policy learn which search actions are useful for solving the original question.

K Evaluator Training and State–Outcome Correlation

K.1 Offline Evaluator Training

We sample 14,000 state–answer pairs from 1,400 co-training steps, covering multiple training stages and search turns, and use these pairs for supervised fine-tuning of an independent evaluator. Once trained, its parameters are held fixed throughout actor training. This produces a task-trained, static evaluator for the ablation in Table 2. The resulting actor achieves 57.94 F1 on HotpotQA and 58.10 on 2Wiki; no MuSiQue result is reported for this setting. Both scores are below OASES.

K.2 State–Outcome Correlation

For each setting, we pair the F1 score of each state-conditioned evaluation with the final-answer F1 of the corresponding trajectory. Pearson correlation is computed separately for each search turn and training stage. The four settings are: the current policy without optimization of evaluation rollouts; frozen Qwen-7B; frozen GPT-4o; and OASES with shared-parameter search–evaluation optimization. We divide the 3,000-step training run into three consecutive 1,000-step intervals, denoted Early, Middle, and Late. We report Turns 0–2; later turns are excluded because they occur in relatively few long-tail trajectories.

Table 10: Pearson correlation between state-evaluation F1 and final-answer F1, reported as $r \times 100$. Columns group results by search turn and training stage. Early, Middle, and Late denote the three consecutive 1,000-step intervals. Bold marks the highest correlation in each column.

Evaluator	Turn 0			Turn 1			Turn 2		
	Early	Middle	Late	Early	Middle	Late	Early	Middle	Late
Policy model	35.57	44.44	41.26	58.96	55.39	54.31	73.13	78.71	82.31
Frozen Qwen-7B	28.76	39.93	37.38	39.88	45.40	46.53	75.01	72.42	82.55
Frozen GPT-4o	39.47	42.48	24.07	39.21	52.14	35.13	46.00	64.60	45.68
OASES	**40.92**	**45.86**	**46.88**	**61.69**	**66.79**	**67.36**	**85.91**	**91.94**	**93.71**

OASES has the highest correlation in all nine comparisons and improves across successive training stages at each turn. These correlations describe the association between absolute state scores and final-answer quality. Each evaluator is assessed on trajectories from its corresponding training configuration, so the comparisons involve different state distributions.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: : We claimed the main contribution and the scope in our abstract and introduction.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss some limitations of our proposed method in Appendix G.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

Justification: The paper does not include theoretical results

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide all the details and the code url in the paper.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: The paper provides a code release link, and the released repository includes the implementation and instructions needed to reproduce the main results.

Guidelines:

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

6. **Experimental setting/details**

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

Answer: [Yes]

Justification: We provide many details of the experiments and the key experimental settings.

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [No]

Justification: We dont do the statistical significance tests.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We discuss experiments compute resources in Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: No ethics problem.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [N/A]

    Justification: This discussion is not necessary for our work.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: Our work poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Our paper meets this requirement.

    Guidelines:

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

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: We release the OASES code, with documentation for installation, training, and evaluation provided in the repository linked in the paper.

    Guidelines:

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

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

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

    Answer: [N/A]

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

    Guidelines:

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

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

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

    Answer: [N/A]

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

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: We only checked for syntax errors using LLM.

Guidelines:

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