arXiv:2611.32917v1 [cs.IR] 24 Nov 2026

SOARS：面向智能体搜索的 System One 辅助重排序

Amin Bigdeli 滑铁卢大学
abigdeli@uwaterloo.ca

Liye Fu 汤森路透实验室
liye.fu@thomsonreuters.com

Fattane Zarrinkalam 圭尔夫大学
fzarrink@uoguelph.ca

Joel Stremmel 汤森路透实验室
joel.stremmel@thomsonreuters.com

Elizabeth Lingg 汤森路透实验室
elizabeth.lingg@thomsonreuters.com

摘要

深度搜索智能体通过迭代检索与推理来满足复杂的信息需求，而检索到的证据的相关性会影响下游性能。现有重排序器能够提升检索效果，但在整个搜索轨迹中持续调用会带来显著的延迟和成本。我们提出 SOARS，这是一个免训练框架，利用 System One 模型快速、校准地估计相关性。在每个搜索步骤中，SOARS 从长文档中筛选出查询相关段落，并在一次调用中对候选项进行重排序，同时考虑原始问题和智能体生成的子查询。在 BrowseComp-Plus 和 WebShaper 上使用六种大语言模型骨干进行评测，SOARS 始终能够提升答案准确率和证据召回率，同时减少搜索调用次数、端到端延迟和推理成本。在 BrowseComp-Plus 上，SOARS 将 GPT-5.4 Mini 的准确率从 73.9% 提升至 84.7%，同时将搜索调用次数从 23 次降至 14 次，展现了 System One 重排序在提升深度搜索效果与效率方面的潜力。

1 引言

深度搜索智能体利用大语言模型（LLM），通过反复进行搜索、阅读和推理来应对复杂的信息需求，并根据整个搜索过程中收集的证据决定下一步检索和阅读的内容 [5, 6, 20, 21]。这一范式支持推理密集型任务：其答案并不包含在任何单一文档中，而必须从分散于多个文档中的证据中加以汇总 [2]。鉴于深度搜索智能体的交互预算有限，在网络搜索中，一个核心挑战是如何让这些智能体有效且高效地发现最相关的证据。

为了向深度搜索智能体提供相关的语料库证据，已有研究探索了多个方向。一些方法通过针对智能体生成的子查询训练稠密检索器来改进第一阶段检索 [1]，或将词汇检索调整用于长文档 [5]。另一种方法通过分解来对智能体进行热启动——

提出问题，并对互补搜索的结果进行重排序，以构建初始证据上下文，指导迭代搜索与推理 [14]。检索感知型智能体控制也已得到探索，旨在通过监测信息覆盖率和搜索新颖性来缓解推理停滞并减少重复搜索 [16]。然而，这些方法关注的是检索、初始化或轨迹控制，而不是在整个搜索过程中对证据进行重排序。

对第一阶段检索得到的候选项进行重排序，是提升检索效果的常见方法，通常使用交叉编码器 [9, 10] 或基于大语言模型的重排序器，通过提示或推理来评估相关性 [12, 13, 17, 24]。近期研究

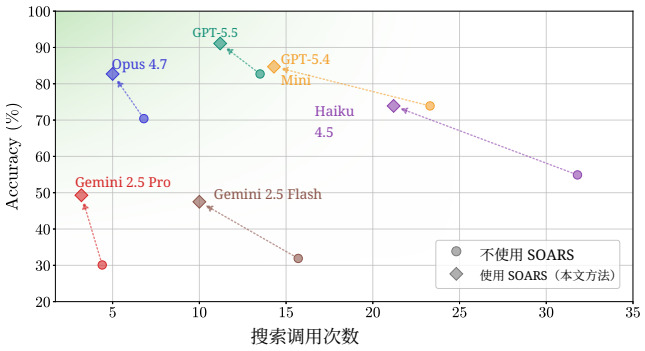


图1：在 BrowseComp-Plus 上，SOARS 对不同大语言模型所驱动的深度搜索智能体的准确率和搜索调用次数的影响 [2]。

已将重排序应用于深度搜索智能体 [8, 15]，并证明这能提升检索效果和答案准确率。然而，在深度搜索智能体中，重排序并非一次性操作，而是往往在整个搜索过程中每次搜索调用时都会执行，这会增加延迟，却无法显著减少搜索次数。此外，现有方法通常仅根据智能体生成的子查询对候选项进行排序，忽略了原始问题的全局信息需求，并依赖初始文档词元，而这些词元可能遗漏相关证据。因此，这些方法以牺牲效率为代价提升了检索效果，却未能显著减少智能体的搜索工作量。

新兴的 System One 模型为同时提升检索效果与效率提供了一个颇具前景的方向。不同于生成式大语言模型，这些模型能在单次前向传播中生成结构化且经过校准的决策，从而实现快速且具成本效益的推理 [3, 4]。Li 等人近期的研究 [7] 表明，System One 模型 Jev 的相关性判断与前沿模型相当，而延迟和成本则显著更低。这些能力对于深度搜索智能体尤为重要，因为它们会反复进行检索和重排序。在每个搜索步骤中优先呈现相关证据，System One 重排序便能让智能体接触到更多有用信息，从而有望减少搜索调用次数、代理推理开销和成本。然而，要将这些模型应用于深度搜索，仍需解决长文档处理以及……之间相互作用的问题

原始问题的全局信息需求与智能体生成的子查询的局部信息需求之间。

为此，**本文的主要目标**是探究轻量级、反复进行的 System One 重排序如何提升深度搜索智能体的效果与效率。具体而言，我们提出了三个研究问题（RQs）。（**RQ1**）如何将 System One 模型适配为循环内重排序器，以提升深度搜索

智能体的检索效果？（**RQ2**）System One 重排序如何影响智能体后续的答案准确率和证据召回率？（**RQ3**）就延迟和成本而言，它对整体智能体效率有何影响？

为探究这些问题，我们提出了 SOARS（**S**ystem**O**ne **A**ssisted **R**eranking for Agentic **S**earch，面向智能体搜索的 System One 辅助重排序），这是一个无需训练的框架，将 System One 重排序集成到深度搜索智能体中。SOARS 使用查询相关段落表示长文档，并在单次调用中对候选项进行重排序，同时考虑原始问题和智能体生成的子查询。我们在两个深度搜索基准 BrowseComp-Plus [2] 和 Web-Shaper [18], 上，使用来自不同系列和规模的六个大语言模型对 SOARS 进行了评估。结果表明，SOARS 的检索效果可媲美强大的基于大语言模型的重排序器，同时运行速度快一到两个数量级。在智能体层面，SOARS 会优先处理相关证据，以支持后续推理，从而提升所有骨干模型的答案准确率，并减少搜索调用次数（图1）。值得注意的是，GPT-5.4 Mini 的准确率从 73.9% 提升至 84.7%，而搜索调用次数则从 23 次降至 14 次。这些改进还转化为更低的延迟和成本，表明高效且反复进行的重排序能够改善整体搜索轨迹。

表1：查询和文档表示对宏平均重排序器召回率和延迟（毫秒）的影响。

	Gold				证据				
	R@5		R@10		R@5		R@10		延迟
重排序器	首部重叠	首部重叠	首部重叠	首部重叠	首部重叠	首部重叠	首部重叠	首部重叠	(毫秒)↓
BM25	26.3		34.4		27.6		33.8		–
查询：仅子查询									
GTE-ModernBERT	29.4	31.2	36.5	39.0	27.1	27.6	32.8	34.6	416
monoT5	28.6	28.8	34.8	36.7	25.6	25.6	30.9	32.7	397
Qwen3-Reranker-4B	28.7	30.5	33.9	36.4	26.0	26.9	31.9	33.9	4,833
gpt-oss-120b	35.6	36.9	40.2	**42.8**	32.0	32.9	**38.1**	**39.5**	52,188
RankGPT [17]	**37.8**	**37.1**	**41.5**	42.3	**33.4**	**33.3**	38.0	39.1	8,167
SOARS（我们的方法）	35.8	36.6	40.1	41.2	30.8	31.7	37.6	38.2	**241**
查询：原始问题 + 子查询									
GTE-ModernBERT	32.5	33.4	38.0	38.7	27.9	28.4	33.5	34.3	712
monoT5	27.6	27.8	33.6	33.5	23.1	23.1	29.8	30.1	695
Qwen3-Reranker-4B	36.3	36.2	41.2	41.3	33.2	33.3	39.4	39.6	8,236
gpt-oss-120b	42.6	43.3	45.9	46.0	37.7	37.8	42.7	42.5	51,427
RankGPT	**43.6**	43.6	**46.5**	46.6	**39.2**	39.2	**43.8**	43.6	8,706
SOARS（我们的方法）	42.9	**44.0**	45.9	**47.2**	38.9	**39.4**	43.6	**44.4**	**254**
预言机	46.3		48.0		41.6		45.5		–

2 提出的方法

问题定义。给定一个复杂问题 Q，其支撑证据分布在语料库 D 中，深度搜索智能体通过对检索到的证据进行迭代式搜索、阅读和推理来生成答案。该智能体由大语言模型 S_θ 驱动，并通过策略 π_θ 在 ReAct 式循环 [20]中运行。在每一轮 t中，它生成推理轨迹 τ_t、选择一个操作 a_t并接收一个观测结果 o_t，从而形成如下所示的交互轨迹：

$$H_T = (\tau_1, a_1, o_1, \ldots, \tau_T, a_T, o_T), \qquad (\tau_t, a_t) \sim \pi_\theta(\cdot \mid H_{t-1}). \quad (1)$$

搜索操作向检索器发出一个子查询 q_t，检索器返回针对 q_t检索到的文档列表 L_{q_t}。智能体在有限的交互预算下运行，因此，检索到的证据呈现顺序，以及每个文档被展示的形式，决定了智能体最终能够阅读和推理的内容。重排序L_{q_t} 可以重新组织这些证据，但由于它会在轨迹中的每次搜索调用时重复执行，因此要想具备实用性，不仅必须有效，还必须高效。本文提出 SOARS，这是一种无需训练的重排序器，可对每次循环内搜索返回的候选项进行重排序，然后再将其呈现给智能体。

深度搜索智能体。智能体通过发起搜索、预览返回的结果并完整阅读文档 [5]，与语料库进行交互。在每一轮 t中，当智能体选择带有子查询 q_t 的搜索操作时，第一阶段检索器 R 会从 D 中检索出文档，并返回如下排序列表：

$$L_{q_t} = \mathcal{R}(q_t, \mathcal{D}). \quad (2)$$

智能体随后预览并读取 L_{q_t}，以收集支持 Q 的证据。由于智能体只检查此列表的顶部，因此放置在列表前部附近的文档决定了哪些证据会进入轨迹。我们的目标是在每次搜索时高效地对 L_{q_t} 进行重排序，使智能体能够接触到更多相关文档。

System One 重排序。SOARS 通过为每个文档准备查询和紧凑表示，并使用 System One 模型对其进行评分，来对 L_{q_t}的前 k 个文档进行重排序。这样可以让模型的上下文容纳更多候选项，从而在一次调用中对大型候选池进行重排序。

由于智能体生成的子查询 q_t 捕捉了局部信息需求，仅对其进行匹配可能会忽略原始问题中的约束 Q。因此，我们根据组合查询 $\tilde{q}_t = Q \oplus q_t$对文档进行评分，同时兼顾局部相关性和全局约束。为适应 System One 模型的有限输入并实现更深入的重排序，我们使用包含 m 个词元的查询相关段落来表示每个文档 d。我们不选择前导词元，因为这可能会遗漏文档更深处的证据；相反，我们选择从智能体生成的子查询 q_t中某个非停用词词项最早出现的位置开始的、包含 m 个词元的片段 w。我们仅使用 q_t 进行段落选择，以聚焦于智能体的即时信息需求，同时在重排序过程中保留 Q，以考虑其与全局信息需求的相关性。随后，我们将文档标题置于段落之前，从而得到 $\hat{d} = \text{title}(d) \oplus w$。这种紧凑表示优先保留相关证据，同时使更多候选项能够纳入模型的输入预算之内。

令 J 表示 System One 重排序器。我们在一次推理调用中，将来自 L_{q_t} 的前 k篇文档提供给 J，由它对每个查询-文档对进行逐点评分。它按四个等级的序数相关性等级对每个候选项进行评定：相关性 $\in \{0、1、2、3\}$，分别对应离题、主题相关、部分满足和完全满足回答查询所需的信息。J 不会将结果限定为单一等级，而是针对每篇文档 d，在给定组合查询 $\tilde{q}_t$及其段落 $\hat{d}$的条件下，返回覆盖四个等级的校准分布$\mathbf{p}^d$。每个相关性等级对应的概率定义如下：

$$p^d(rel) = \Pr_{\mathcal{J}}\left(rel \mid \tilde{q}_t, \hat{d}\right), \quad (3)$$

其中，$\Pr_{\mathcal{J}}$ 表示 J 的校准概率，$\sum_{\text{相关性}=0}^{3} p^d(\text{相关性})$。我们根据该分布下的期望相关性等级为每个文档评分：

表2：大语言模型在 BrowseComp-Plus 和 WebShaper 上的深度搜索智能体性能。SOARS 显著提升了准确率、预览召回率和行为召回率，使智能体能够接触到更多相关证据。阅读文档数表示在预览后，每个智能体生成的子查询平均完整阅读的文档数量；该数值越高，表示为深入检查而选取的证据越多。† 表示相较于基线，在配对双尾 t-检验（$p < 0.05$）下取得显著改进。

LLM	BrowseComp-Plus [2]									WebShaper [18]					
	答案质量		呈现召回率		已预览		行为		阅读文档数↑	答案质量		召回率			阅读文档数↑
	Acc.	校准↓	Gold	Evi.	Gold	Evi.	Gold	Evi.		Acc.	校准↓	呈现召回率	已预览	行为	
GPT-5.5	82.7	15.4	94.4	94.7	72.9	73.6	56.1	58.9	5.0	69.5	17.3	88.4	38.2	26.0	7.0
+ **SOARS**	**91.1**†	**4.3**†	**97.8**	**95.3**	**88.7**†	**83.3**†	**79.0**†	**68.7**†	**7.1**†	**77.0**†	**10.2**†	86.9	**47.9**†	**34.2**†	**10.5**†
GPT-5.4 Mini	73.9	11.1	94.9	92.2	69.9	63.7	55.6	45.7	4.9	57.5	17.7	84.7	30.6	19.3	6.1
+ **SOARS**	**84.7**†	**8.7**†	**95.5**	92.1	**81.7**†	**74.8**†	**67.8**†	**54.9**†	**7.0**†	**64.5**†	**16.4**	83.7	**38.7**†	**24.6**†	**8.8**†
Opus 4.7	70.4	10.3	86.8	81.2	52.5	43.3	43.0	30.4	2.1	67.5	4.4	68.8	21.4	14.4	3.2
+ **SOARS**	**82.7**†	**9.7**	85.5	78.5	**65.0**†	**53.9**†	**50.0**†	**35.3**†	**2.4**†	**68.0**	5.9	64.3	**28.9**†	**20.6**†	**4.3**†
Haiku 4.5	54.9	17.1	94.1	60.1	58.2	46.3	40.9	39.4	8.8	48.5	16.2	87.4	35.5	23.2	8.8
+ **SOARS**	**73.9**†	**10.6**†	**96.1**	**94.7**	**79.3**†	**75.7**†	**62.3**†	**54.7**†	**10.2**†	**57.5**†	16.3	87.1	**46.2**†	**31.2**†	**11.6**†
Gemini 2.5 Pro	30.1	36.5	73.1	71.5	26.4	25.5	16.8	14.3	1.2	38.0	30.7	71.9	16.7	6.8	1.1
+ **SOARS**	**49.3**†	**30.1**†	72.6	68.4	**45.4**†	**41.4**†	**28.6**†	**22.0**†	**1.7**†	**44.0**	34.3	71.1	**25.2**†	**10.7**†	**1.4**†
Gemini 2.5 Flash	31.9	20.1	80.3	79.1	34.8	34.6	21.9	19.9	2.4	27.5	30.1	77.5	19.5	8.1	1.7
+ **SOARS**	**47.5**†	**17.6**	**81.4**	78.4	**55.7**†	**51.5**†	**33.5**†	**28.4**†	**3.3**†	**35.5**†	**29.0**	75.1	**28.3**†	**12.7**†	**2.3**†

$$s(d) = \mathbb{E}_{rel \sim \mathbf{p}^d}[rel] = \sum_{rel=0}^{3} rel \cdot p^d(rel). \tag{4}$$

然后，我们按照 $s(d)$ 的降序重新排列排名前 k 个文档。相比仅使用最可能的等级，使用完整分布能够提供分级排序信号，而单次调用评分则可将每次搜索开销保持在较低水平。

3 实验

3.1 设置

3.1.1 数据集。我们在两个基准上评估 SOARS，这些基准旨在评估深度搜索智能体处理复杂且推理密集型问题的能力。**BrowseComp-Plus** [2] 包含 830 个复杂问题和约 10 万篇文档，并提供人工验证的判断，以区分回答问题所需的证据文档与同样包含答案的标准答案文档。**WebShaper** [18] 仅提供标准答案文档。我们随机抽取 200 个问题，并在 [1] 发布的约 100 万篇文档的语料库中进行搜索。

3.1.2 重排序器基线。我们将 SOARS 与 monoT5 [10] 和采用中等推理能力的 gpt-oss-120b [11], 进行比较，这两者均沿用此前深度搜索智能体研究中的做法 [8, 15]。我们还纳入了以 GPT-4.1 为骨干模型的 RankGPT [17] 、Qwen3-Reranker- 4B [23], 和 GTE-ModernBERT [22]。我们在单张 NVIDIA A10 GPU 上运行后两种重排序器以及 monoT5。

3.1.3 评估指标。我们围绕三个研究问题组织评估指标。对于**检索效果**（RQ1），我们测量每次搜索向智能体呈现的重排序列表的召回率。对于**智能体效果**（RQ2），遵循先前工作 [1, 2, 5, 15],, 我们以 GPT-4.1 作为评判者，测量端到端的准确率和校准误差，并通过三种召回率沿轨迹追踪标准答案/证据 [5]。具体而言，呈现召回率 基于智能体搜索返回的文档计算，预览召回率基于智能体检查其摘要片段的文档计算，而行为召回率 则基于智能体打开或在答案中引用的文档计算。对于**智能体效率**（RQ3），我们报告平均搜索调用次数、单次查询延迟以及总成本。我们根据提交时每百万个输入词元、输出词元和缓存读取词元对应的 API 费率计算成本。

3.1.4 实现细节。我们将 SOARS 集成到 Pi-Serini 智能体框架 [5] 中，并研究来自不同系列和规模的六种大语言模型骨干，即 GPT-5.5、GPT-5.4 Mini、Claude Opus 4.7、Claude Haiku 4.5、Gemini 2.5 Pro 和 Gemini 2.5 Flash。每个骨干模型均以中等推理强度运行，单次查询预算为 300 秒。遵循 [5],，我们使用 BM25（$k_1 = 25$、$b = 1$）作为第一阶段检索器。对于 System One 重排序，我们采用 Jev [19],，在每次搜索中使用文档标题和包含 $m = 128$ 个词元的查询相关段落，对排名靠前的 $k = 100$ BM25 候选项进行重排序。Jev 的成本为每百万输入词元 0.042 美元，输出免费 [19]。对于不使用 SOARS 的基础基线，我们复用 Pi-Serini 发布的 GPT-5.5、Opus 4.7 和 Haiku 4.5 轨迹，并以 GPT-4.1 作为评判者评估答案准确率；其余骨干模型则在相同设置下运行。

3.2 结果与发现

3.2.1 RQ1：检索有效性。我们研究如何将 System One 模型应用于循环内重排序，并将其效果与现有重排序器进行比较。我们随机选取了 50 个 BrowseComp-Plus 问题，其中包含由 GPT-5.4 Mini 生成的 1,044 个子查询及其 BM25 候选项。对于每个子查询，SOARS 和所有基线都会对 BM25 检索出的前 100 个文档进行重排序。我们考察了两种查询表示，以评估智能体搜索中的全局和局部信息需求。第一种使用智能体生成的子查询来捕捉即时检索目标；第二种则将其与原始问题结合，以保留更广泛的上下文和约束。我们还比较了两种文档表示：一种使用前导词元（*Lead*），另一种使用从子查询中的非停用词术语首次出现位置开始的段落（*Overlap*），以捕捉文档中更深层的证据。这些比较评估了查询上下文和段落选择如何影响重排序效果。表1报告了问题级别的宏平均 Recall@5、Recall@10 和重排序延迟。预言机会优先选择每个 BM25 候选池中的标准答案或证据文档，从而建立召回率上界。

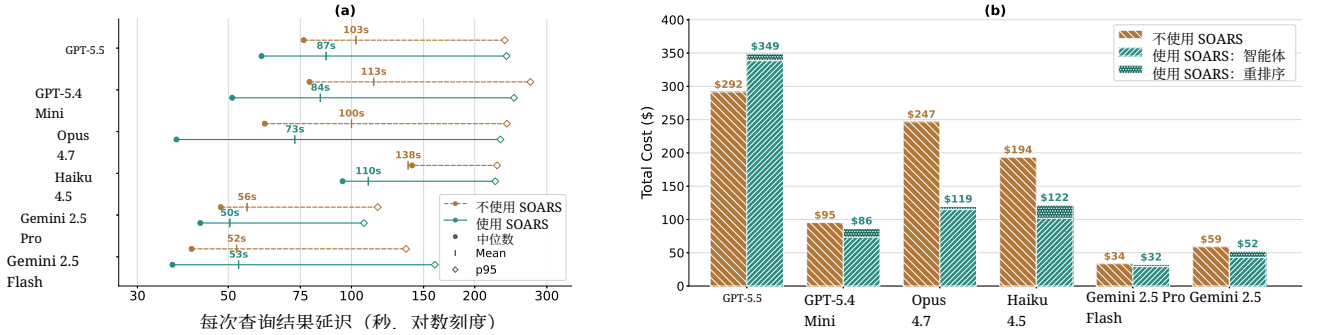


图 2：SOARS 对 六种不同大语言模型骨干网络下，深度搜索智能体的单次查询延迟（a）和总成本（b） es.

查询表示和文档表示都会影响检索效果。将原始问题与智能体生成的子查询结合，可提升大多数重排序器的召回率，而 *Overlap* 通常优于 *Lead*。在结合设置下，SOARS 在全部四项指标上均取得最高召回率，标准答案和证据 Recall@10 分别达到 47.2% 和 44.4%。这些结果接近预言机上界，表明当通过聚焦文档段落共同考虑全局和局部信息需求时，System One 重排序能够实现很高的效果。

除了检索效果外，SOARS 还显著降低了延迟。尽管 GTE-ModernBERT 和 monoT5 的单次查询延迟较低，SOARS 却能以更低的延迟实现显著更高的召回率。更值得注意的是，使用通过 Overlap 选出的段落，并将原始问题与子查询结合后，SOARS 的表现始终优于最强的大语言模型重排序器 RankGPT，而每次查询仅需 254 毫秒，相比之下，RankGPT 需要 8.7 秒。检索效果出色且延迟低，这一组合使 System One 重排序尤其适合在智能体的搜索轨迹中反复调用。

3.2.2 RQ2：智能体效果。我们评估了将 System One 重排序集成到深度搜索智能体后，对两个基准测试中端到端答案准确率和证据召回率的影响。

BrowseComp-Plus。SOARS 在所有大语言模型上都显著提升了答案准确率，相对增幅从 GPT-5.5 的 10.2% 到 Gemini2.5Pro 的 63.8% 不等。这些提升在较弱的智能体上尤为明显，缩小了它们与更强骨干模型之间的性能差距。校准误差也持续降低，表明置信度估计更加可靠。更重要的是，证据召回率分析揭示了这些提升的来源。尽管使用相同的第一阶段检索器，且呈现召回率相近，SOARS 仍显著提升了所有大语言模型的预览召回率和行为召回率。对于 GPT-5.5，预览金标准召回率从 72.9 提升至 88.7，而行为金标准召回率从 56.1 提升至 79.0。

这些发现表明，SOARS 通过将已检索到的证据提升至智能体预览、阅读和引用的文档中，从而提高答案准确率。此外，所有大语言模型的平均文档阅读数量均显著增加。智能体会先预览简短的文档摘录，再选择性地打开有希望的文档阅读全文，这表明 SOARS 有助于识别更多值得深入检查的证据。值得注意的是，这些提升伴随着搜索调用次数的减少（图1），说明更有效的证据优先级排序降低了进一步检索的需求。

WebShaper。同样的规律也适用于 WebShaper：对于每种骨干模型，已预览召回率和行为召回率均显著提高，且大多数模型的准确率也显著提升。所有骨干模型的文档阅读数量也均显著增加，这表明系统选择了更多有望带来价值的证据进行深入检查。这些发现证实，SOARS 在整个搜索轨迹中持续提升证据利用效率，从而提高答案准确率。

3.2.3 RQ3：智能体效率。 我们从三个维度评估逐步 System One 重排序的效率：搜索调用次数、单次查询延迟和成本。我们报告了 BrowseComp-Plus 上的结果；在 WebShaper 上也观察到了大体相似的趋势。

搜索调用次数。SOARS 减少了所有大语言模型的搜索调用次数（图1），对于搜索密集型智能体而言尤为明显：GPT-5.4 Mini 的搜索调用次数从 23 次降至 14 次，Haiku 4.5 则从 32 次降至 21 次。通过将相关文档提升至排序列表顶部，SOARS 帮助智能体在搜索轨迹中更早地识别并利用有用证据，从而减少后续搜索的需求。

延迟。尽管每次搜索都会增加一次重排序调用，SOARS 仍降低了几乎所有骨干模型的单次查询延迟（图2a）：GPT-5.4 Mini 从 113 秒降至 84 秒，Opus 4.7 从 100 秒降至 73 秒。更少的搜索调用抵消了重排序器带来的少量额外开销，从而实现总体延迟下降。这些结果表明，轻量级重排序能够通过减少在重复搜索交互上耗费的时间，加快整体搜索过程。

成本。六个骨干模型中有五个的总成本有所下降（图2b），对于成本较高的智能体尤其明显：Opus 4.7 从 247 美元降至 119 美元，Haiku 4.5 从 194 美元降至 122 美元。搜索工作量的减少抵消了极少的重排序成本，从而降低了智能体的整体计算成本。GPT-5.5 是唯一的例外，其成本小幅增加，可能是由于阅读了更多证据，从而提升了准确率（表 2）。

4 总结

我们介绍了 SOARS，这是一种免训练框架，可将 System One 模型适配为深度搜索智能体迭代循环中的重排序器。在 BrowseComp-Plus 和 WebShaper 上的六个大语言模型骨干中，SOARS 始终能够提升答案准确率和证据召回率，同时减少智能体的搜索调用次数、延迟和成本。这些结果表明，快速且经过校准的 System One 模型能够使逐步重排序兼具有效性与效率，从而确立其作为构建深度搜索智能体的实用组件的地位。

参考文献

[1] Zijian Chen、Xueguang Ma、Shengyao Zhuang、Jimmy Lin、Akari Asai 和 Victor Zhong。2026。AgentIR：面向深度研究智能体的推理感知检索。*CoRR* abs/2603 04384 2026 2603 04384 10 48550 2603 04384 。（）。arXiv：。[2] doi：。/ARXIV。。Zijian Chen、Xueguang Ma、Shengyao Zhuang、Ping Nie、Kai Zou、Sahel Sharify- moghaddam、Andrew Liu、Joshua Green、Kshama Patel、Ruoxi Meng、Mingyi Su、Yanxi Li、Haoran Hong、Xinyu Shi、Xuye Liu、Hosna Oyarhoseini、Nandan Thakur、Crystina Zhang、Luyu Gao、Wenhu Chen 和 Jimmy Lin。2026。BrowseComp- Plus：面向深度搜索智能体的公平且相互解耦的评测基准。载于 计算语言学协会第 *64* 届年会论文集（第 *1* 卷：长篇论文），Maria Liakata、Viviane P. Moreira、Jiajun Zhang 和 David Jurgens（编）。计算语言学协会，美国加利福尼亚州圣迭戈，22349 22370 10 18653 1 2026 1023 –。doi：。/v / .acl-long。[3] 2026 Tobias Deußer、Lorenz Sparrenberg 和 Rafet Sifa。。评估和基准测试 System One 模型 Jev。*CoRR* abs/2609.37647（2026）。arXiv：2609.37647 doi:10.48550/ARXIV.2609.37647[4] Joas Antonio dos Santos Barbosa。2026。面向自主渗透测试工具链的校准决策模型：JEV 和 Laya 作为大语言模型驱动渗透测试智能体的 System One 决策层。*CoRR* abs/2609.28940（2026）。arXiv：2609.28940 doi:10.48550/ARXIV.2609.28940[5] Tz-Huan Hsu、Jheng-Hong Yang 和 Jimmy Lin。2026。重新思考采用 Pi-Serini 的智能体搜索：词汇检索是否足够？*CoRR* abs/2605.10848（2026）。arXiv：2605.10848 doi:10.48550/ARXIV.2605.10848[6] Yuxuan Huang、Yihang Chen、Haozheng Zhang、Kang Li、Meng Fang、Linyi Yang、Xiaoguang Li、Lifeng Shang、Songcen Xu、Jianye Hao、Kun Shao 和 Jun Wang。2025。深度研究智能体：系统性考察与路线图。*CoRR* abs/2506 18096 2025 2506 18096 10 48550 2506 18096 。（）。arXiv：。[7] 2026 doi：。/ARXIV。。Yubo Li、Yidi Miao、Ramayya Krishnan 和 Rema Padman。。JEV 作为评判者：自信时接受，不确定时升级。*CoRR* abs/2609.26550（2026）。arXiv：2609.26550 doi:10.48550/ARXIV.2609.26550[8] Chuan Meng、Litu Ou、Sean MacAvaney 和 Jeff Dalton。2026。重新审视深度研究中的文本排序。载于 第 *49* 届国际 *ACM SIGIR* 信息检索研究与开发会议论文集，*SIGIR 2026*，澳大利亚墨尔本，*2026* 年 *7* 月 *20—24* 日，Alistair Moffat、Falk Scholer、Hannah Bast、Marc Najork 和 Min Zhang（编）。ACM，3006–3016。doi:10.1145/3805712。3808557[9] Rodrigo Nogueira 和 Kyunghyun Cho。2019。使用 BERT 进行段落重排序。*CoRR* abs/1901 04085 2019 1901 04085 10 48550 1901 04085 。（）。[10] 2020 arXiv：。doi：。/ARXIV。。Rodrigo Nogueira、Zhiying Jiang、Ronak Pradeep 和 Jimmy Lin。。使用预训练序列到序列模型进行文档排序。载于 计算语言学协会论文集：*EMNLP 2020*，线上会议，*2020* 年 *11* 月 *16—20* 日（*ACL* 论文集，*EMNLP 2020* 卷），Trevor Cohn、Yulan He 和 Yang Liu（编）。计算语言学协会，708–718。doi:10.18653/V1/2020.FINDINGS-EMNLP.63[11] OpenAI。2025。gpt-oss-120b 和 gpt-oss-20b 模型卡片。arXiv:2508.10925 [cs.CL] https://arxiv.org/abs/2508 10925 。Ronak Pradeep、Sahel Sharifymoghaddam 和 [12] Jimmy Lin。2023。RankVicuna：使用开源大语言模型进行零样本列表式文档重排序。*CoRR* abs/2309.15088（2023）。arXiv:2309.15088 doi:10.48550/ARXIV.2309。15088[13] Ronak Pradeep、Sahel Sharifymoghaddam 和 Jimmy Lin。2023。RankZephyr：高效且稳健的零样本列表式重排序易如反掌！*CoRR* abs/2312.02724（2023 2312 02724 10 48550 2312 02724 ）。arXiv：。doi：。/ARXIV。。[14] Radin Hamidi Rad、Amin Bigdeli、Negar Arabzadeh、Sajad Ebrahimi、Charles L. A. Clarke、Benjamin C. M. Fung 和 Ebrahim Bagheri。2026。问题的策略：第一步在智能体深度搜索中至关重要。*CoRR* abs/2609.14412（2026）。arXiv:2609.14412 doi:10.48550/ARXIV.2609.14412

[15] Sahel Sharifymoghaddam 和 Jimmy Lin。2026。先重排序，再推理：通过深度搜索智能体中的有效词元成本分析重排序权衡。载于 计算语言学协会论文集，*ACL 2026*，美国加利福尼亚州圣迭戈，*2026*年*7*月*2*日至*7*日，Maria Liakata、Viviane P. Moreira、Jiajun Zhang 和 David Jurgens（编）。计算语言学协会，25874 25886 10 18653 1 2026 1289 –。doi：。/V / .[16] FINDINGS-ACL。Heydar Soudani、Elizabeth Lingg、Faegheh Hasibi 和 Navid Rekabsaz。2026。深度研究智能体何时陷入停滞：利用检索感知的智能体控制增强推理。载于 *2026*年自然语言处理实证方法会议论文集，Hung-Yi Lee、Sunipa Dev、Junyi Jessy Li 和 Desmond Elliott（编）。计算语言学协会，匈牙利布达佩斯，44072–44095。doi:10.18653/2026.emnlp-main.1937[17] Weiwei Sun、Lingyong Yan、Xinyu Ma、Shuaiqiang Wang、Pengjie Ren、Zhumin Chen、Dawei Yin 和 Zhaochun Ren。2023。ChatGPT 擅长搜索吗？探究大语言模型作为重排序智能体的表现。载于 *2023*年自然语言处理实证方法会议论文集，*EMNLP 2023*，新加坡，*2023*年*12*月*6*日至*10*日，Houda Bouamor、Juan Pino 和 Kalika Bali（编）。计算语言学协会，14918–14937。doi:10.18653/V1/2023.EMNLP-MAIN.923[18] Zhengwei Tao、Jialong Wu、Wenbiao Yin、Pu Wu、Junkai Zhang、Baixuan Li、Haiyang Shen、Kuan Li、Liwen Zhang、Xinyu Wang、Wentao Zhang、Yong Jiang、Pengjun Xie、Fei Huang 和 Jingren Zhou。2026。WebShaper：通过信息寻求形式化进行智能体式数据合成。载于第十四届国际学习表征会议，*ICLR 2026*，巴西里约热内卢，*2026*年*4*月*23*日至*27*日，Carl Vondrick、Bharath Har- iharan、Colin Raffel、Lerrel Pinto、Diyi Yang 和 Aleksandra Faust（编）。proceedings.iclr.cc。https://proceedings.iclr.cc/paper_files/paper/2026/hash/a6ffd1854c6191b1f33b7ad9509d46f9-Abstract-Conference.html[19] TypeSafe AI。2026。*TypeSafe API*参考。https://docs.typesafe.ai 访问日期：2026年9月20日。[20] Shunyu Yao、Jeffrey Zhao、Dian Yu、Nan Du、Izhak Shafran、Karthik R. Narasimhan 和 Yuan Cao。2023。ReAct：在语言模型中协同实现推理与行动。载于第十一届国际学习表征会议，*ICLR 2023*，卢旺达基加利，*2023*年*5*月*1*日至*5*日。OpenReview.net。https://openreview. net/forum?id=WE_vluYUL-X[21] Weinan Zhang、Junwei Liao、Ning Li 和 Kounianhua Du。2024。智能体信息检索。*CoRR* abs/2410.09713（2024）。arXiv:2410.09713 doi:10.48550/ ARXIV.2410.09713[22] Xin Zhang、Yanzhao Zhang、Dingkun Long、Wen Xie、Ziqi Dai、Jialong Tang、Huan Lin、Baosong Yang、Pengjun Xie、Fei Huang、Meishan Zhang、Wenjie Li 和 Min Zhang。2024。mGTE：面向多语言文本检索的通用长上下文文本表示与重排序模型。载于 *2024*年自然语言处理实证方法会议论文集：*EMNLP 2024—Industry Track*，美国佛罗里达州迈阿密，*2024*年*11*月*12*日至*16*日，Franck Dernoncourt、Daniel Preotiuc-Pietro 和 Anastasia Shimorina（编）。计算语言学协会，1393–1412。doi:10.18653/V1/2024.EMNLP-INDUSTRY.103[23] Yanzhao Zhang、Mingxin Li、Dingkun Long、Xin Zhang、Huan Lin、Baosong Yang、Pengjun Xie、An Yang、Dayiheng Liu、Junyang Lin、Fei Huang 和 Jingren Zhou。2025。Qwen3嵌入：通过基础模型推进文本嵌入与重排序。*CoRR* abs/2506.05176（2025）。arXiv:2506.05176 doi:10.48550/ARXIV.2506.05176[24] Shengyao Zhuang、Xueguang Ma、Zheng Yao、Shuai Wang、Bevan Koopman、Jimmy Lin 和 Guido Zuccon。2026。Rank-R1：通过强化学习增强大语言模型文档重排序器的推理能力。载于 第*49*届国际 *ACM SIGIR* 信息检索研究与开发会议，*SIGIR 2026*，澳大利亚墨尔本，*2026*年*7*月*20*日至*24*日，Alistair Moffat、Falk Scholer、Hannah Bast、Marc Najork 和 Min Zhang（编）。ACM，4419–4425。doi:10.1145/3805712.3809961