

我们是否仍需要 GraphRAG ? 针对智能体搜索系统的 RAG 与 GraphRAG 基准测试

Dongzhe Fan
纽约大学上海分校
上海, 中国
df2362@nyu.edu

Zheyi Xue, Siyuan
Liu
纽约大学上海分校
上海, 中国
{zx1793,sl11766}@nyu.edu

Qiaoyu Tan
纽约大学上海分校
上海, 中国
qiaoyu.tan@nyu.edu

摘要

基于检索增强生成(RAG)及其基于图形的扩展(GraphRAG)是通过将生成过程锚定在外部知识上来提升大语言模型 (LLM) 推理能力的有效范式。然而, 大多数现有的 RAG 和 GraphRAG 系统在静态或 one-shot 检索下运行, 即在单次传递中向 LLM 提供一组固定的文档。相比之下, 近期的 Agentic 搜索系统在推理过程中实现了动态的多轮检索与序列决策, 当与原始 RAG 结合时, 通过交互引入隐式结构, 已展现出显著性能提升。

这一进展引发了一个根本性问题: Agentic 搜索能否弥补显式图结构的缺失, 从而降低对昂贵 GraphRAG 流水线的需求?

为回答该问题, 我们提出了 RAGSearch, 一个统一的基准, 用于评估密集型 RAG 和代表性 GraphRAG 方法作为 Agentic 搜索中的检索基础设施。RAGSearch 涵盖了多个问答基准上的无训练和基于训练的 Agentic 推理。为确保公平且可复现的比较, 我们标准化了 LLM 主干、检索预算和推理协议, 并在完整的测试集上报告结果。除了答案准确率外, 我们还报告了离线预处理成本、在线推理效率及稳定性。结果显示, Agentic 搜索显著提升了密集型 RAG, 并缩小了其 GraphRAG 之间的性能差距, 尤其在基于强化学习(RL)的设置中表现突出。然而, 在复杂多跳推理任务中, GraphRAG 仍具有优势, 当其离线成本被分摊后, 表现出更稳定的 Agentic 搜索行为。综合来看, 这些发现阐明了显

式图结构与 Agentic 搜索之间的互补作用, 并为现代 Agentic RAG 系统的检索设计提供了实用指导。

该基准代码和评估脚本已在 <https://github.com/FanDongzhe12> 公开。

CCS Concepts

• **Do Not Use This Code** → **Generate the Correct Terms for Your Paper**; *Genesrate the Correct Terms for Your Paper*; • **请勿使用此代码** → 为您的论文生成正确的术语; 为您的论文生成正确的术语。

Keywords

检索增强生成, 图谱检索增强生成, 智能体搜索系统, 强化学习

ACM Reference Format:

Dongzhe Fan, Zheyi Xue, Siyuan Liu, and Qiaoyu Tan. 2018. 我们是否仍需要 GraphRAG? 针对智能体搜索系统的 RAG 与 GraphRAG 基准测试. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 21 pages. <https://doi.org/XXXXXXX.XXXXXXX>

1 引言

检索增强生成 (Retrieval-augmented generation, RAG) 是一种广泛采用的范式, 通过在推理时检索相关文档或文本块来将大语言模型 (Language Model, LLMs) 与外部知识相结合 [13, 20, 33]。由于其简洁性和高效性, 基于稠密检索的 RAG 已成为知识密集型应用中的标准组件。最近, 提出了基于图形的 RAG (Graph-based RAG, GraphRAG) 方法 [3, 8], 通过显式地将检索内容组织成结构化表示——如层次树 [3, 29]、实体图 [5–7, 10] 或超图 [4, 24], 从而进一步提升推理性能, 实现更有效的多跳 (Multi-Hop) 推理和证据 (Evidence) 聚合。

尽管效果显著, 现有的 RAG 和 GraphRAG 系统主要

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Conference acronym 'XX, Woodstock, NY

© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-1-4503-XXXX-X/2018/06

<https://doi.org/XXXXXXX.XXXXXXX>

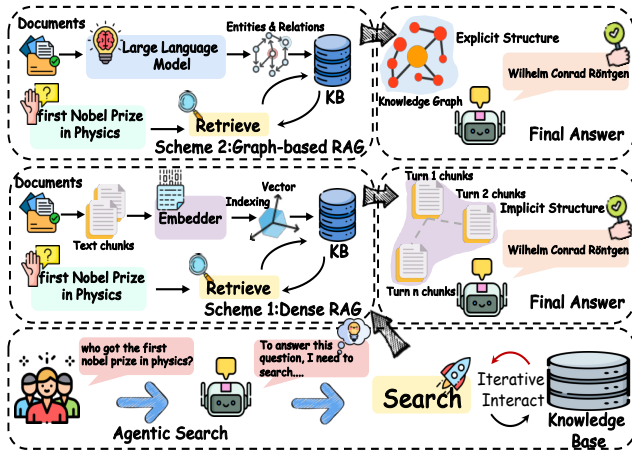


图 1: RAG 系统中显式结构与隐式结构的对比。GraphRAG 依赖于显式的图结构构建, 而稠密 RAG 上的 Agentic 搜索则可通过多轮检索与推理来生成隐式的证据结构。

单次传递中向大模型提供一组固定的检索文档。这一假设限制了检索过程在推理过程中适应中间推理状态的能力。与此同时, Agentic 搜索系统 [14, 21, 22, 25, 27, 36] 近期作为一项强大的替代方案崭露头角, 将检索从静态的预处理步骤转变为动态的多轮过程。通过实现序列化决策, Agentic 系统使大模型能够基于部分推理结果迭代地优化查询和检索策略, 并在与基础 RAG 结合时展现出显著的实证优势 [20]。

这些发展揭示了现代检索增强系统中的一个根本性矛盾。从结构角度看 (图 1), 稠密 RAG 基于语义相似度独立地检索和排序文本片段, 未显式建模检索单元之间的关系。GraphRAG 引入了显式的图结构, 通过注入关系归纳偏置来指导多步检索与推理。相比之下, Agentic 搜索则通过交互引入隐式结构, 利用序列化控制在推理过程中组织信息访问。这引发了一个核心问题:

Agentic 检索能否弥补稠密 RAG 中显式图结构的缺失, 还是说在 Agentic 推理下 GraphRAG 依然必要?

回答该问题并不简单。尽管先前的研究表明, GraphRAG 要最新信息时, 往往会产生幻觉内容。为缓解这些问题, 检索增强生成 (RAG) [20] 通过从外部知识库中检索相关文档片段来增强大模型的能力。在此流水线基础上, 近期基于 RAG 的大模型推理研究聚焦于改进模型在规划、检索和利用外部证据进行推理方面的表现。总体而言, 现有方法可大致分为无需训练 [9, 39] 和基于训练 [12, 32, 38] 两种范式。

然而, 尽管这个问题至关重要, 现有的评估并未提供明确的答案。先前的研究 [14, 19, 23, 37] 通常采用不一致的评估协议, 依赖部分测试集, 或在不同方法之间存在计算资源预算的差异——尤其是对于 Agentic 系统, 其检索调用和 token 使用量很少受到控制。

此外, GraphRAG 方法通常被评估为一个整体的端到端系统 [2], 而非可跨不同推理范式重复使用的检索基础设施。这些局限性使得难以判断显式的图结构在何时真正必要, 以及 Agentic 搜索在何时可作为有效的替代方案。

为了填补这一空白, 我们引入了 RAGSearch, 这是一个用于研究智能体搜索下检索增强生成的统一基准。RAGSearch 将稠密 RAG 和基于图形 RAG 视为 RAG 范式中的替代检索基础设施, 并在统一协议、匹配的检索预算和完整测试集评估下, 评估它们与智能体推理的交互作用。我们的关键贡献如下:

- **统一的基准。**我们提出了 RAGSearch, 这是首个及时推出的基准, 系统性地评估稠密 RAG 和多种代表性 GraphRAG 流水线作为智能体搜索系统中检索基础设施的表现。RAGSearch 统一了数据集、大语言模型骨干网络、检索预算和评估协议, 使得在静态检索与动态、智能体推理情景之间能够进行受控比较。
- **全面的评估。**我们实现了基于原始 RAG 和五种代表性 GraphRAG 方法的无训练智能体搜索方法以及强化学习优化的智能体系统, 并进行了基准测试。这使我们能够研究不同检索结构形式如何与推理范式中的智能体控制相互作用, 而非孤立地评估检索或智能体。
- **多维度分析。**除了答案准确率之外, RAGSearch 还报告离线预处理成本、在线推理效率以及在智能体控制下的稳定性, 揭示了在何种情况下智能体搜索能够弥补无结构感知检索的不足, 以及在构建成本更高的情况下基于图形的检索为何依然具有优势。所有代码、配置和排行榜均已发布, 以支持可复现的评估和未来的扩展。

2 相关工作

我们的工作与以下三个方向密切相关。

基于 RAG 的大模型推理。大模型 (LLMs) 取得了显著进展, 但仍存在明显局限性, 特别是在领域特定或知识密集型场景中, 当查询超出其训练数据范围或需检索相关文档片段来增强大模型的能力。在此流水线基础上, 近期基于 RAG 的大模型推理研究聚焦于改进模型在规划、检索和利用外部证据进行推理方面的表现。总体而言, 现有方法可大致分为无需训练 [9, 39] 和基于训练 [12, 32, 38] 两种范式。

GraphRAG 增强的大模型推理。尽管标准的 RAG 能够通过检索到的文本证据有效锚定大模型, 但在多跳或关系型查询中, 支持信息分布在多个文档时, 可能表现不足。GraphRAG [3] 通过利用知识的图结构表示解决了这些局限性。受此范式启发, 近期工作扩展了

GraphRAG 的设计空间，并提出了更计算高效的图构建流水线。RAPTOR [29] 通过递归聚类 and 摘要文本块构建层次化树状索引，实现多层次抽象下的检索，从而提升长上下文和多跳推理能力。受海马体记忆索引理论启发，HippoRAG2 [7] 从抽取的事实构建以实体为中心的语料库图，并应用类似个性化 PageRank 的传播机制，在文档间检索具备多跳和关系感知特性的证据。超越传统的图结构表示，HyperGraphRAG [24] 将知识库构建为超图，以捕捉高阶关系。而 LinearRAG [40] 则采用轻量级实体抽取和语义链接构建无关系的分层“三图”，在无需昂贵关系抽取的前提下，实现可扩展的基于图的检索。

Agentic Search. 然而，大多数现有的 GraphRAG 系统仍然遵循单次检索范式，近期方法 [14, 21–23] 通过在大语言模型 (LLM) 引导下迭代优化查询，探索多步检索。Search-o1 [21] 将逐步的 LLM 推理与按需外部检索交织进行，并使用“文档内推理”模块在整合到推理过程前对检索到的文档进行精炼。GraphSearch [22] 通过联合查询文本片段和 GraphRAG，实现迭代式的多步检索，以支持复杂的多跳推理。除了无需训练的方法外，近期工作也探索了基于训练的方法。Search-R1 [14] 采用基于强化学习 (RL) 的训练范式，教导大语言模型将逐步推理与多轮搜索交织进行，通常以稠密检索方式在外部语料库中获取信息，以支持每个推理步骤。而 Graph-R1 [23] 则将多轮搜索扩展至 GraphRAG 情景。

3 初步的

我们考虑开放域问答系统，输入为查询 q 和大型语言模型 (LLM) \mathcal{M} 。在标准的 LLM 推理中，模型仅根据查询生成答案 $y: y \sim \mathcal{M}(q)$ 。尽管强大，这种形式完全依赖于模型参数中编码的知识，难以处理需要外部信息访问的知识密集型或多跳推理任务。

检索增强推理 (RAG)。 RAG 通过将生成过程基于外部知识语料库来扩展大语言模型的推理能力 $\mathcal{K} = \{d_1, \dots, d_N\}$ 。一个 RAG 系统由一个检索器 \mathcal{R} 和一个大语言模型 \mathcal{M} 组成。给定一个查询 q ，检索器会选择一组相关的文档或文本片段 $C_q = \mathcal{R}(q | \mathcal{K})$ ，并将这些内容附加到模型输入中以生成答案: $y \sim \mathcal{M}(q, C_q)$ 。在大多数现有的 RAG 流水线中，检索在解码之前执行一次，且检索到的上下文在整个生成过程中保持不变。这种静态或 *one-shot* 检索的假设是基于稠密检索的 RAG 系统的主要基础。

基于图形的检索增强生成 (GraphRAG) GraphRAG 通过在推理前将知识语料库显式组织为结构化的图或超图，进一步扩展了检索增强推理。我们将 GraphRAG 知识库抽象为 $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ ，其中结点 \mathcal{V} 表示文本单元 (例如文档、实体或摘要)，边 \mathcal{E} 编码它们之间的关系。GraphRAG 中的检索通常从 \mathcal{G} 中选择结点、路径或子

图，然后将这些结构化证据提供给大语言模型: $y \sim \mathcal{M}(q, Z_q)$, $Z_q \subseteq \mathcal{G}$ 。与稠密 RAG 相比，GraphRAG 引入了显式的结构归纳偏置，能够提升多跳推理和证据聚合能力。然而，与标准 RAG 类似，GraphRAG 的检索通常仅在单次预处理步骤中完成，所选证据在解码前被迫追加至模型输入。

在本工作中，我们聚焦于在 *Agentic* 检索情景下对 RAG 和 GraphRAG 进行基准测试，其中检索被整合到推理过程中，并在解码期间迭代执行。该范式实现了多轮检索以及基于中间推理状态的自适应信息访问，从根本上改变了检索与大语言模型推理之间的交互方式。

4 RAG 搜索基准

在本节中，我们介绍 **RAGSearch**，这是一个旨在系统研究不同检索基础设施与智能体搜索系统之间交互关系的基准。RAGSearch 将稠密 RAG 和 GraphRAG 视为统一智能体搜索框架内的可互换检索后端，从而在标准化情景下实现对不同智能体推理范式之间可控且公平的比较。RAGSearch 框架的概览如图 2 所示。

我们首先在第 4.1 节中形式化了一种通用的智能体搜索抽象及其与检索后端的交互方式。随后，我们在第 4.2 节中描述了 RAGSearch 中实例化的两种代表性智能体流水线：无需训练的智能体搜索，该方法依赖结构化提示和启发式控制；在第 4.3 节中则介绍了基于强化学习的智能体搜索，其中智能体策略通过领域特定数据进行优化。最后，第 4.4 节明确了 RAGSearch 中包含的检索后端及其作为基准内环境的实例化方式。

4.1 通用 Agentic 搜索表述

我们使用受 ReAct [36] 框架启发的高层抽象形式化 RAGSearch 中的 *Agentic* 检索，将推理建模为推理与检索环境交互的交错环。给定输入查询 q ，一个配备有大语言模型 \mathcal{M} 的智能体 \mathcal{M} 将在多个步骤中与检索后端 \mathcal{B} (例如稠密 RAG 或 GraphRAG) 进行交互。

在每个步骤 t ，智能体执行推理，其内容由 `<think>` 和 `</think>` 限定，基于可用信息和交互历史，决定是否调用检索或终止并生成最终答案。当触发检索时，智能体会发出一个搜索查询 q_t ，其内容由 `<search>` 和 `</search>` 限定。系统随后在 \mathcal{B} 上执行检索操作：

$$\mathcal{I}_t^q = \text{RETRIEVE}(q_t, \mathcal{B}). \quad (1)$$

此处， \mathcal{I}_t^q 表示检索到的信息，可以是一组文本块 C_{q_t} 或一个子图 Z_{q_t} 。这些信息用 `<information>` 和 `</information>` 包裹，并附加到正在进行的推理序列中。该过程持续迭代，直到智能体决定在 `<answer>` 和 `</answer>` 内

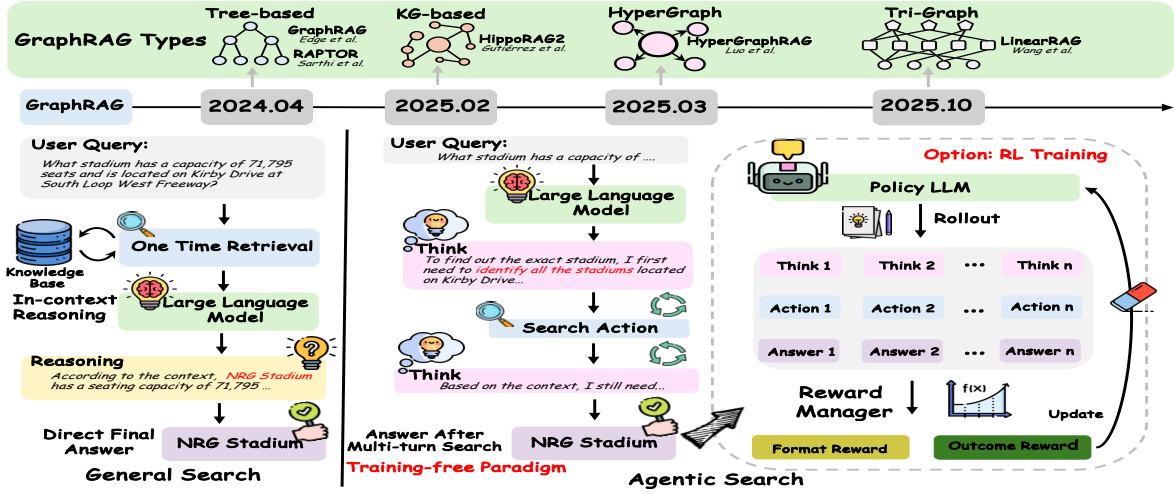


图 2: RAGSearch 基准概述。RAGSearch 将智能体搜索建模为大语言模型智能体通过统一接口与可互换的检索后端（稠密 RAG 或 GraphRAG）交互，基于标准化协议对无需训练的智能体搜索和基于强化学习的智能体搜索进行基准测试。

生成最终答案。生成的推理过程可表示为：

$$P(R, a \mid \mathcal{P}, \mathcal{B}) = \underbrace{\prod_{t=1}^{T_R} P(R_t \mid R_{<t}, \mathcal{P}, q_t, \mathcal{I}_{<t}^q)}_{\text{Reasoning Process}} \cdot \underbrace{\prod_{t=1}^{T_a} P(a \mid R, \mathcal{I}_t^q)}_{\text{Answer Generation}} \quad (2)$$

其中 \mathcal{P} 表示系统模板， R_t 为在步骤 t 生成的推理序列。

该公式有意忽略了动作空间和记忆表示的低层细节，突出了现代 Agentic 系统共享的两个关键特性：(i) 检索在推理过程中动态执行，而非作为 one-shot 预处理步骤；(ii) 相同的 Agentic 控制逻辑可作用于不同的检索基础设施。RAGSearch 采用此抽象，将 Agentic 推理与检索后端解耦，从而在统一的 Agentic 检索框架下实现对稠密 RAG 与 GraphRAG 的系统性基准测试。

4.2 无需训练的 Agentic 搜索流水线

基于第 4.1 节中的一般性智能体搜索公式，我们实例化了一类无需训练的智能体搜索流水线，其设计反映了当前最先进系统（如 Search-o1 [21] 和 GraphSearch [34]）的特点。这些方法不学习显式的控制策略；相反，它们依赖于结构化提示和启发式控制，在推理过程中引导多轮检索与推理。具体而言，当代无需训练的智能体搜索系统通常可分为两种代表性范式：(i) **以推理驱动的按需搜索**，其中模型根据不确定性或信息需求自适应地触发检索（例如 Search-o1），以及 (ii) **协调式多**

智能体工作流，通过提示和路由显式协调模块化角色，完成分解、验证和迭代检索任务（例如 GraphSearch）。

4.2.1 基于推理的按需搜索。 该范式通常依赖模型自身的推理过程来决定是否调用搜索。与基本的 ReAct 型环相比，该范式通过引入“文档中推理”组件，增强了推理与交互能力：

Query \rightarrow Think \rightarrow Search \rightarrow Knowledge Refinement \rightarrow Answer
Iterative

知识提炼。 为了在不超出上下文限制的情况下支持长时程推理，无需训练的智能体通常会在将检索到的观测结果纳入智能体状态之前对其进行摘要。遵循 Search-o1 [21] 的设计，每个检索结果都会被压缩成一个简洁的摘要，以捕捉与当前推理目标最相关的关键证据。

4.2.2 编排的多智能体工作流。 该范式强调多模块之间的协同互动，将原始查询分解为更小、更易处理的子查询，通过顺序求解这些子查询来回答原始问题。该范式通常表现出以下过程：

Query \rightarrow Docomposition \rightarrow Search \rightarrow Verification \rightarrow Answer
Iterative

查询分解。 智能体不必发出单一的查询，而是可以将原始问题分解为一系列子查询，每个子查询代表原问题中一个更小且易处理的组成部分：

$$\{q_1, q_2, \dots, q_i\} = \text{Decomposition}(q) \quad (3)$$

每个子查询 q_i 专注于解决单一实体、关系或上下文依赖。每个子查询 q_i 及其关联的检索文档 I_{q_i} 由逻辑起

草模块处理, 以构建推理链 \mathcal{L} , 从而解决原始问题。该机制受 GraphSearch [34] 启发, 使检索器能够访问细粒度证据并降低推理复杂度。

证据验证. 该模块通过考虑事实依据、一致性和潜在矛盾, 评估 \mathcal{L} 中累积的证据是否足够且逻辑自治, 以支持最终答案。当证据缺失或不一致时, 该模块会进一步扩展子查询集, 并迭代执行额外的检索以获取相关上下文。

备注. 在该框架下, Search-o1 和 GraphSearch 可被视为同一无需训练的 Agentic 框架的不同实例, 其主要区别在于它们所交互的检索环境。稠密 RAG 环境返回非结构化文本块, 而 GraphRAG 环境返回基于图形的结构化证据。关键在于, Agentic 控制逻辑保持不变。RAGSearch 采用这一统一视角, 在相同的推理协议下, 对无需训练的 Agentic 检索流水线在稠密 RAG 和多个 GraphRAG 基础设施上进行基准测试。

4.3 基于强化学习的智能体搜索训练

尽管无需训练的 Agentic 流水线提供了强大且灵活的基准, 但其控制逻辑完全由提示驱动, 并在推理时固定不变。因此, 这类智能体严重依赖底层大语言模型的内在推理能力, 无法根据特定任务分布或推理模式调整其检索策略。这一局限性促使我们从领域特定数据中学习智能体的控制策略, 从而使得检索和推理行为能够适应目标任务。

基于强化学习的 Agentic 搜索公式化. 继 Search-R1 和 Graph-R1 之后, 我们将智能体搜索问题表述为强化学习问题, 其中智能体策略 π_θ 由具有可训练参数 θ 的大语言模型参数化。给定输入查询 q , 智能体与检索环境 \mathcal{B} (稠密 RAG 或 GraphRAG) 交互并生成一条轨迹

$$\tau^{(i)} = (s_1^{(i)}, a_1^{(i)}, \dots, s_T^{(i)}, a_T^{(i)}),$$

其中每个动作 a_t 对应一个与检索相关的决策或终止动作, 该动作生成最终答案 y 。在终止后, 为整个轨迹分配一个标量奖励 $r(q, y, \tau)$ 。学习目标是最大化训练分布上的期望奖励:

$$\max_{\pi_\theta} \mathbb{E}_{q \sim \mathcal{D}, \tau \sim \pi_\theta(\cdot|q, \mathcal{B})} [r(q, y, \tau)]. \quad (4)$$

该公式直接优化序列级别的智能体行为, 使策略能够学习特定任务的检索和推理模式。

为了优化智能体策略, RAGSearch 采用组相对策略最优化 (GRPO), 该方法在 Search-R1 和 Graph-R1 中已有应用。对于每个训练查询 q , 智能体在当前策略下采样一组 K 条轨迹 $\{\tau^{(1)}, \dots, \tau^{(K)}\}$ 。每条轨迹获

得一个奖励 $r^{(k)}$, 该奖励在组内进行规范化, 以计算相对优势:

$$\hat{A}(\tau_i) = \frac{r^{(i)} - \text{mean}\left(\{r^{(j)}\}_{j=1}^K\right)}{F_{\text{norm}}\left(\{r^{(j)}\}_{j=1}^K\right)}.$$

策略通过增加具有正相对优势轨迹的似然来更新, 同时通过 KL 正则化约束更新过程, 使其趋向于一个固定的参考策略 π_{ref} :

$$\mathcal{L}_{\text{GRPO}}(\theta) = \left[\frac{1}{K} \sum_{i=1}^K \frac{1}{|\tau_i|} \sum_{t=1}^{|\tau_i|} \min\left(\rho_\theta(a_t^{(i)}) \hat{A}(\tau_i), g(\epsilon, \hat{A}(\tau_i))\right) - \beta \mathbb{D}_{\text{KL}}(\pi_\theta \parallel \pi_{\text{ref}}) \right] \quad (5)$$

其中, $\rho_\theta(a_t^{(i)}) = \frac{\pi_\theta(a_t^{(i)} | s_{t-1}^{(i)}; \mathcal{B})}{\pi_{\theta_{\text{old}}}(a_t^{(i)} | s_{t-1}^{(i)}; \mathcal{B})}$, $g(\epsilon, \hat{A}(\tau_i)) = \text{clip}(\rho_\theta(a_t^{(i)}), 1 \pm \epsilon) \hat{A}(\tau_i)$ 和 β 控制正则化的强度。GRPO 避免了显式的价值函数学习, 在训练长时域基于大语言模型的智能体方面已被证明是有效的。

奖励设计. 在 RAGSearch 中, 奖励是在轨迹层面定义的, 重点关注任务正确性和输出有效性。具体而言, 我们结合了 (i) 一种基于结果的奖励, 用于衡量答案的正确性 (例如确切匹配或特定任务的准确率), 以及 (ii) 一种基于格式的奖励, 鼓励智能体遵循预期的交互和答案格式。重要的是, 相同的奖励公式被应用于稠密 RAG 和 GraphRAG 检索环境中, 确保学成的智能体行为差异反映了策略与检索基础设施之间的相互作用, 而非奖励设计上的差异。

备注. 从统一的视角来看, 基于强化学习的 Agentic 搜索可被视为在第 4.1 和 4.2 节所介绍的同一 Agentic 框架内学习一种自适应控制策略。检索后端和交互协议保持不变; 仅对智能体的策略使用领域特定的监督进行优化。这使得 RAGSearch 能够在相同的 RAG 与 GraphRAG 基础设施上系统地比较无需训练与学成的 Agentic 控制, 并评估策略学习在 Agentic 搜索中如何与显式的图结构相互作用。

4.4 RAGSearch 中的检索后端

RAGSearch 将一组固定的检索后端实例化为可互换的环境, 用于智能体搜索。所有后端向智能体暴露统一的检索接口, 仅在外部的知识和访问方式上有所不同, 从而实现对不同检索基础设施的可控比较。

我们包含一个结构无关的稠密 RAG 基准, 该基准将语料库索引为非结构化文本块, 并通过语义相似度搜索检索证据。此外, 我们还考虑了五种具有代表性的 GraphRAG 后端, 它们涵盖了多种图构建和检索策略:

- **基于树的:** **GraphRAG** [3], 基于层次化社区进行多跳证据聚合; **RAPTOR** [29], 从递归摘要树中检索证据;
- **基于实体图:** **HippoRAG2** [7], 采用以实体为中心的图表示;
- **基于超图的:** **HypergraphRAG** [24], 通过超边捕获高阶关系;
- **基于三图的:** **LinearRAG** [40], 在检索内容上施加了一个轻量级的线性结构。

所有 GraphRAG 后端在离线状态下进行预处理以构建其图表示, 但均通过第 4.1 节所述的相同 Agentic 交互协议访问, 无需针对特定后端修改智能体。这种标准化设置使得在无需训练和基于强化学习的 Agentic 推理下, 能够直接比较稠密检索与图结构检索的效果。

5 实验

在本节中, 我们在基准情景下进行了广泛的实验, 旨在回答以下研究问题: **RQ1:** Agentic 检索能否弥补稠密 RAG 中显式图结构的缺失? **RQ2:** 在无需训练的 Agentic 检索下, 显式图结构是否仍具有优势? **RQ3:** 通过强化学习进行策略学习如何与不同的检索基础设施相互作用? **RQ4:** RAG 与 GraphRAG 在 Agentic 推理下的鲁棒性与稳定性有何差异? **RQ5:** 不同模块在 Agentic 系统中的影响如何?

5.1 实验设置

5.1.1 数据集. 为了全面评估六个标准问答系统 (QA) 数据集 [15]: (1) **通用问答:** Natural Questions (NQ) [18], PopQA [26] 和 TriviaQA [16]; (2) **多跳问答:** HotpotQA [35], Musique [31] 和 2WikiMultiHopQA (2Wiki) [11]。更多细节见附录 A。

5.1.2 搜索智能体与 RAG 系统. 为了探究在智能体推理下 GraphRAG 是否仍然必要, 我们采用了四种典型的 Agentic 搜索系统: 两种无需训练的方法, Search-o1 [21] 和 GraphSearch [34], 以及两种基于强化学习的方法, Search-R1 [14] 和 Graph-R1 [23]。我们还采用静态的 one-shot 检索方法作为基准。对于稠密 RAG, 我们使用原始 RAG 作为检索后端, 并以 2018 年维基百科数据集 [17] 作为知识源。对于 GraphRAG, 我们选取了五种代表性方法: GraphRAG [3], RAPTOR [29], HippoRAG2 [7], HyperGraphRAG [24], LinearRAG [40]。对于无训练和基于强化学习的智能体系统, 我们分别实现了使用上述六种检索后端的变体。

5.1.3 实现细节. 我们使用 GPT-4o-mini 在不同的基于图形的检索基础设施中进行知识构建。所有无训练的

智能体系统均采用 Qwen2.5-7B-Instruct 和 Qwen2.5-32B-Instruct [28] 作为大语言模型骨干。对于基于强化学习的 Agentic 系统, 我们采用 GRPO [30] 作为训练过程。其大语言模型骨干使用 Qwen2.5-3B-Instruct 和 Qwen2.5-7B-Instruct。我们联合在 HotpotQA 和 NQ 上预训练所有基于强化学习的系统。从这些数据集的训练集中随机采样 5000 个结点。对于测试集, 我们使用测试集或开发集的完整数据。所有实验均在 2 块 NVIDIA A100 GPU (80GB) 上完成。更多细节见附录 B。

5.1.4 评价指标. 我们使用两个指标对所有智能体系统进行评估: F-1 和包含确切匹配 (Contain EM)。更多细节见附录 C

5.2 总体比较 (RQ1)

在本节中, 我们对基于图形和稠密检索的 RAG 在不同 Agentic 系统中的表现进行了全面比较。

5.2.1 单次推理. 观察 1 在单次推理情景下, 稠密 RAG 已经在通用问答任务中表现出色, 而 GraphRAG 主要是在多跳问答任务上带来决定性提升。如表 1 所示, 原始稠密 RAG 在通用问答基准测试中已取得具有竞争力的性能, 而在该情景下, GraphRAG 仅带来微小改进, 平均提升为 +0.47。相比之下, GraphRAG 在多跳问答任务上显著优于稠密检索, 在 HotpotQA、2Wiki 和 Musique 上平均提升达 +27.23。这一显著差异表明, 显式的图结构表示对于需要多跳证据聚合和组合推理的任务尤为有益, 而对于单跳或事实型问题则优势有限。

5.2.2 无需训练的搜索智能体. 观察 2 Agentic 检索可以增强稠密 RAG 并部分缩小与 GraphRAG 的差距, 但效果取决于智能体设计。对比表 1 和 2, 我们发现 Agentic 检索对稠密 RAG 的增益并不一致。在 Search-o1 下, 稠密 RAG 表现参差不齐, 甚至在多个基准上出现性能下降, 表明通用的多轮交互本身并不能可靠提升稠密检索。相比之下, 在 GraphSearch 下, 稠密 RAG 在除 Musique 外的一般问答和多跳问答任务中均显著优于单次推理, 这是由于有效的查询分解和迭代检索机制。定量分析显示, 多跳问答任务上的平均稠密 RAG–GraphRAG 差距从单次推理时的 +27.23 降低至 +26.59, 相较于表现第二好的 GraphRAG 变体减少了 32.3%, 表明结构化的 Agentic 检索能够在一定程度上弥补缺乏显式图结构的不足。

5.2.3 训练过的搜索智能体. 观察 3 基于强化学习的训练在稠密 RAG 和 GraphRAG 后端上通常能提升智能体性能, 但并未始终超越表现强劲的无训练智能体流水线。如表 1 所示, 基于强化学习的智能体 (Search-R1 和 Graph-R1) 在通用问答和多跳问答任务上均持续优于其对应的无训练基准。然而, 这些提升并未始终超过具有更结构化工作流的强劲无训练智能体流水线。

表 1: 总体包含单次推理、无训练智能体系统以及基于强化学习的智能体系统的确切匹配 (EM) 得分。♠ 表示五个 GraphRAG 变体中的最佳结果。↑ 和 ↓ 表示同一方法下基于图形与稠密检索的 RAG 之间的性能差异; † 表示领域内数据集, * 表示跨领域数据集。

System	Methods	General QA			Multi-Hop QA		
		NQ [†]	PopQA [*]	TriviaQA [*]	HotpotQA [†]	2Wiki [*]	Musique [*]
Single-shot	Qwen-2.5-7B-Dense	46.62	32.14	58.60	19.00	35.53	20.99
	Qwen-2.5-7B-GraphRAG♠	48.31	32.82	57.65	46.70	62.56	47.95
		(↑1.69)	(↑0.68)	(↓0.95)	(↑27.70)	(↑27.03)	(↑26.96)
Training-free	Search-o1-7B-Dense	38.20	25.57	58.74	33.76	29.64	12.62
	Search-o1-7B-GraphRAG♠	38.34	28.01	59.50	42.75	65.56	32.44
		(↑0.14)	(↑2.44)	(↑0.76)	(↑8.99)	(↑35.92)	(↑19.82)
	GraphSearch-7B-Dense	58.27	36.29	68.70	38.22	47.43	13.33
	GraphSearch-7B-GraphRAG♠	61.22	44.77	72.47	58.64	79.88	55.26
		(↑2.95)	(↑8.48)	(↑3.77)	(↑20.42)	(↑32.45)	(↑41.93)
RL-based	Search-R1-7B	48.72	33.10	63.96	35.76	33.56	14.42
	Graph-R1-7B♠	46.71	36.23	66.21	53.42	66.25	40.82
		(↓2.01)	(↑3.13)	(↑2.25)	(↑17.66)	(↑32.69)	(↑26.40)

表 2: 总体包含不同检索器后端的无训练智能体系统的 EM 结果。和 分别表示每种方法中的最佳和第二佳结果。表示稠密-RAG, 表示基于树的 GraphRAG, 表示基于实体图的 GraphRAG, 表示基于超图的 GraphRAG, 表示基于三元图的 GraphRAG。平均秩在每种方法内计算 (越低越好)。

Method	Knowledge Base	General QA			Avg Rank	Multi-Hop QA			Avg Rank
		NQ	PopQA	TriviaQA		HotpotQA	2Wiki	Musique	
Search-o1	Dense	38.20	25.78	58.74	2.33	33.76	29.64	12.62	5.33
	HypergraphRAG	33.02	25.57	56.72	5.33	33.90	50.58	28.05	3.67
	HippoRAG2	38.34	28.01	59.50	1.00	42.75	65.56	32.44	1.00
	LinearRAG	34.32	25.69	57.03	4.00	35.76	58.94	29.46	2.33
	RAPTOR	34.82	23.20	52.52	5.33	29.51	29.87	29.50	4.33
	GraphRAG	35.10	26.10	56.89	3.00	32.73	54.25	26.48	4.33
GraphSearch	Dense	58.27	36.29	68.70	4.00	38.22	47.43	13.33	6.00
	HypergraphRAG	51.04	44.72	69.97	3.33	46.83	73.62	54.80	2.33
	HippoRAG2	61.22	43.65	72.47	1.67	58.64	79.88	55.10	1.33
	LinearRAG	52.12	44.77	68.52	4.00	41.65	70.26	49.35	4.33
	RAPTOR	53.80	42.38	68.56	4.33	40.14	71.24	55.26	3.33
	GraphRAG	52.78	43.60	69.64	3.67	42.25	72.41	46.73	3.67

特别是, 基于 GraphSearch 的系统——尽管为无训练设计——通过采用查询分解和结构化检索等明确的设计选择, 通常优于 Search-R1 和 Graph-R1 变体。这些结果表明, 尽管基于强化学习的最优化能够优化智能体行为, 但设计良好的无训练智能体 workflow 仍然极具竞争力, 甚至可能超越基于强化学习的系统。

总而言之, 在多跳问答任务中, 智能体搜索通过迭代检索与推理引入了隐式的结构线索, 部分缓解了稠密 RAG 中缺乏显式图结构的问题。然而, 与观察 1–

3 一致, 图结构 RAG 方法在复杂多跳推理任务上仍展现出最强且最稳定的性能, 表明在此类场景下显式结构表示仍然具有优势。

相比之下, 在通用问答任务中, 智能体搜索与图结构 RAG 带来的性能提升相对有限 (例如, 多跳问答的平均提升分别为 +2.43 与 +26.25)。鉴于图结构 RAG 存在显著的离线构建与延迟开销 (表 8), 在搭配设计良好的智能体 workflow 时, 稠密 RAG 仍然是通用问答场景下一种实用且具有竞争力的替代方案。

5.3 无需训练的 Agentic 工作流 (研究问题 2)

为了回答 **RQ2**, 我们考察了不同检索后端在无训练智能体工作流下的性能影响。对于每种智能体方法, 我们仅改变检索后端, 而保持智能体设计和推理协议不变。结果如表 2 所示。我们观察到 **观察 4** 在无训练智能体工作流中, 显式的图结构仍能为多跳问答持续带来显著且一致的收益, 而稠密 RAG 在通用问答任务中依然具有竞争力。对于多跳问答, 基于图结构的检索后端在 Search-o1 和 GraphSearch 上均始终取得最佳或次佳表现, 表明即使在智能体搜索下, 显式结构表示对多跳推理依然有效。其中, 以实体为中心的 HippoRAG2 相较于稠密 RAG 获得了最大的提升。相比之下, 在通用问答任务中, 稠密 RAG 依然表现出很强的竞争力, 某些情况下甚至优于部分 GraphRAG 变体 (例如在 TriviaQA 上, LinearRAG 和 RAPTOR)。这表明, 尽管图结构对多跳推理尤为有益, 但在无训练智能体工作流中, 稠密 RAG 仍是通用问答任务中一个强大且实用的选择。

5.4 基于强化学习的搜索智能体 (RQ3)

我们评估了通过强化学习进行策略学习在通用问答与多跳问答情景下, 如何与不同的检索基础设施相互作用。从图 3 中我们观察到: **观察 5** 基于强化学习的 Agentic 性能高度依赖后端: 基于图形的检索器在多跳问答中带来更大的提升。在多跳问答中, GraphRAG 风格的后端始终优于稠密 RAG, 尽管基于图形的方法之间性能存在差异。具体而言, 以实体为中心的 HippoRAG2 在 HotpotQA、PopQA 和 2Wiki 等数据集上取得了最强的结果, 表明实体级别的图信号对强化学习尤其有效。相比之下, 在通用问答情景下, 稠密 RAG 的表现与若干基于图形的变体相当, 甚至更优, 特别是在 NQ 数据集上表现最佳。这表明, 即使在基于强化学习的 Agentic 系统中, 稠密 RAG 仍是通用问答的强有力基准, 而图结构在多跳推理中最具优势。

5.5 敏感性分析

5.5.1 稳健分析. 为了回答 **RQ4**, 我们通过检索召回率以及 Contain-EM 的均值和方差 (见表 3), 研究了在 Agentic 检索中稠密 RAG 与 GraphRAG 的鲁棒性。**观察 6** GraphRAG 在 Agentic 检索中比稠密 RAG 更具鲁棒性和稳定性。尽管搜索深度相当, GraphRAG 实现了更高的文档命中率以及更低的 Contain-EM 方差, 表明其在多轮 Agentic 检索场景中具有更可靠的证据检索能力与更强的稳定性。

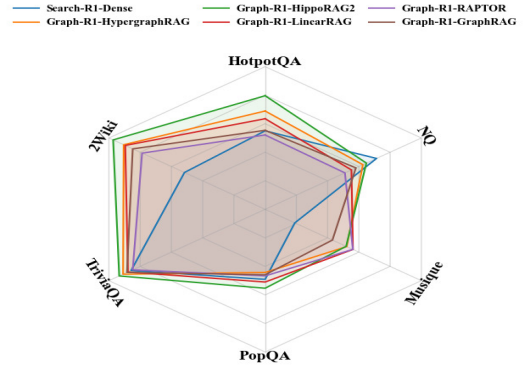


图 3: 基于强化学习的智能体系统中不同检索架构的性能表现

5.5.2 强化学习训练范式的影响. 为回答 **RQ5**, 我们研究了不同强化学习训练范式对智能体搜索性能的影响。我们在相同的智能体情景和检索后端下, 将 GRPO 与其他替代的强化学习算法进行比较。**观察 7** GRPO 是基于强化学习的智能体系统的一种有利训练范式。如图 4 所示, GRPO 在稠密 RAG 和 GraphRAG 两种后端上均持续优于其他强化学习算法。这表明 GRPO 为智能体搜索提供了更有效的策略最优化, 使智能体能够更好地利用结构化证据和多步推理。

5.5.3 不同规模大语言模型骨干网络的影响. 我们进一步分析大模型主干大小对性能的影响 (**RQ5**)。具体而言, 在无需训练的工作流中, 我们对比了 Qwen2.5-7B-Instruct 与 Qwen2.5-32B-Instruct; 在基于强化学习的范式中, 我们使用 Qwen2.5-3B-Instruct 实现了一个 3B 规模的变体。结果如表 4 和表 5 所示。**观察 8** 更大的主干模型不仅提升了推理性能, 还缩小了 GraphRAG 与稠密 RAG 之间的性能差距。在基于强化学习的系统中, 从 3B 缩放到 7B 使平均 GraphRAG-Dense 差距从 14.70 降低至 9.75 (例如, HotpotQA 从 19.08→15.99, PopQA 从 5.74→3.05)。在无需训练的系统中也出现了类似趋势, 当主干规模从 7B 扩展到 32B 时, 平均差距从 7.80 略微下降至 7.19。这些结果表明, 更强的大模型能够通过推理更好地利用隐含的结构线索, 部分弥补缺乏显式图结构所带来的不足。

6 结论

本文介绍了 RAGSearch, 这是一个用于系统评估稠密 RAG 及代表性 GraphRAG 流水线作为智能体搜索系统中检索基础设施的基准。我们的结果表明, 尽管智能体搜索可通过迭代检索与推理在一定程度上弥补稠密 RAG 中缺失的结构, 但显式的基于图形的检索对于稳健的多跳推理仍至关重要。在复杂情景下, GraphRAG 方法始终表现出更强的性能和更高的稳定性, 而稠密

表 3: 对智能体系统的敏感性分析

Method	HotpotQA			PopQA		
	Search Turn	Recall	Variance	Search Turn	Recall	Variance
Search-o1-Dense	2.20	79.38	33.65±1.03	1.53	76.33	25.62±0.61
Search-o1-HippoRAG2	2.03	80.27	42.36±0.22	1.52	78.12	27.81±0.36
Search-R1	1.82	81.67	34.82±0.95	1.36	77.15	33.15±0.54
Graph-R1-HippoRAG2	1.71	83.50	53.71±0.18	1.38	78.61	36.13±0.32

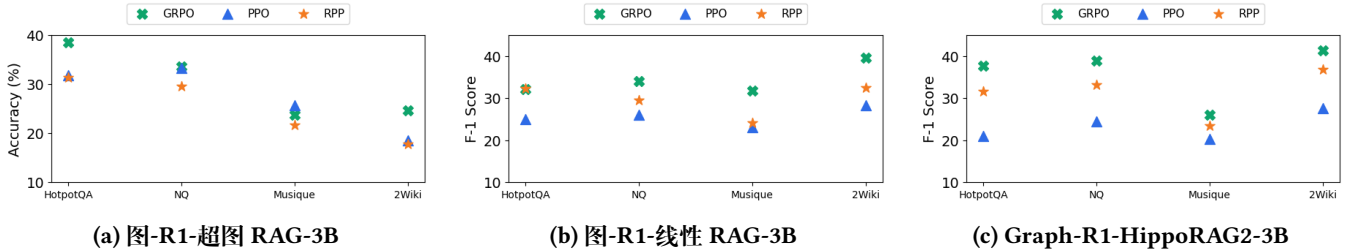


图 4: 不同强化学习算法的影响

表 4: 基于强化学习的 Agentic 系统中 3B 与 7B 的对比

Method	HotpotQA	Musique	NQ	PopQA
Search-R1-3B	23.67	4.96	30.50	25.34
Graph-R1-3B [♣]	42.75	31.73	37.72	31.08
Search-R1-7B	35.76	12.35	46.35	32.90
Graph-R1-7B [♣]	51.75	36.53	42.12	35.95

表 5: 7B 与 32B 在无训练系统中的对比

Method	HotpotQA	Musique	NQ	PopQA
Search-o1-Dense-7B	33.76	12.62	38.20	25.78
Search-o1-Graph-7B [♣]	42.75	32.44	38.34	28.01
Search-o1-Dense-32B	40.85	18.95	51.80	36.20
Search-o1-GraphRAG-32B [♣]	47.01	42.37	50.50	36.69

RAG 由于构建成本较低，仍是通用问答任务中实用且具有竞争力的选择。

更广泛地，我们的发现表明，智能体推理正在重塑大语言模型系统中检索结构的角色。智能体搜索并未取代显式结构，而是重新分配了结构产生的位置——将部分结构从离线图构建转移到在线交互中。理解显式结构与隐式结构之间的平衡，将成为设计下一代增强检索与智能体人工智能系统的关键，我们希望 RAGSearch 能推动该新兴设计空间的进一步研究。

References

[1] Boyu Chen, Zirui Guo, Zidan Yang, Yuluo Chen, Junze Chen, Zhenghao Liu, Chuan Shi, and Cheng Yang. 2025. PathRAG: Pruning Graph-based Retrieval Augmented Generation with Relational Paths. *arXiv:2502.14902 [cs.CL]* <https://arxiv.org/abs/2502.14902>

[2] Junnan Dong, Siyu An, Yifei Yu, Qian-Wen Zhang, Linhao Luo, Xiao Huang, Yunsheng Wu, Di Yin, and Xing Sun. 2025. Youtu-graphrag: Vertically unified agents for graph retrieval-augmented complex reasoning. *arXiv preprint arXiv:2508.19855* (2025).

[3] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitanaky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130* (2024).

[4] Yifan Feng, Hao Hu, Xingliang Hou, Shiquan Liu, Shihui Ying, Shaoyi Du, Han Hu, and Yue Gao. 2025. Hyper-RAG: Combating LLM Hallucinations using Hypergraph-Driven Retrieval-Augmented Generation. *arXiv:2504.08758 [cs.IR]* <https://arxiv.org/abs/2504.08758>

[5] Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2025. LightRAG: Simple and Fast Retrieval-Augmented Generation. *arXiv:2410.05779 [cs.IR]* <https://arxiv.org/abs/2410.05779>

[6] Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2025. HippoRAG: Neurobiologically Inspired Long-Term Memory for Large Language Models. *arXiv:2405.14831 [cs.CL]* <https://arxiv.org/abs/2405.14831>

[7] Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From RAG to Memory: Non-Parametric Continual Learning for Large Language Models. *arXiv:2502.14802 [cs.CL]* <https://arxiv.org/abs/2502.14802>

[8] Haoyu Han, Yu Wang, Harry Shomer, Kai Guo, Jiayuan Ding, Yongjia Lei, Mahantesh Halappanavar, Ryan A Rossi, Subhabrata Mukherjee, Xianfeng Tang, et al. 2024. Retrieval-augmented generation with graphs (graphrag). *arXiv preprint arXiv:2501.00309* (2024).

[9] Bolei He, Nuo Chen, Xinran He, Lingyong Yan, Zhenkai Wei, Jinchang Luo, and Zhen-Hua Ling. 2024. Retrieving, Rethinking and Revising: The Chain-of-Verification Can Improve Retrieval Augmented Generation. *arXiv:2410.05801 [cs.CL]* <https://arxiv.org/abs/2410.05801>

[10] Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-Retriever: Retrieval-Augmented Generation for Textual Graph Understanding and Question Answering. *arXiv:2402.07630 [cs.LG]* <https://arxiv.org/abs/2402.07630>

- [11] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing A Multi-hop QA Dataset for Comprehensive Evaluation of Reasoning Steps. *arXiv:2011.01060* [cs.CL] <https://arxiv.org/abs/2011.01060>
- [12] Shayekh Bin Islam, Md Asib Rahman, K S M Tozammel Hossain, Enamul Hoque, Shafiq Joty, and Md Rizwan Parvez. 2024. Open-RAG: Enhanced Retrieval-Augmented Reasoning with Open-Source Large Language Models. *arXiv:2410.01782* [cs.CL] <https://arxiv.org/abs/2410.01782>
- [13] Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C. Park. 2024. Adaptive-RAG: Learning to Adapt Retrieval-Augmented Large Language Models through Question Complexity. *arXiv:2403.14403* [cs.CL] <https://arxiv.org/abs/2403.14403>
- [14] Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516* (2025).
- [15] Jiajie Jin, Yutao Zhu, Zhicheng Dou, Guanting Dong, Xinyu Yang, Chenghao Zhang, Tong Zhao, Zhao Yang, and Ji-Rong Wen. 2025. FlashRAG: A Modular Toolkit for Efficient Retrieval-Augmented Generation Research. In *Companion Proceedings of the ACM on Web Conference 2025, WWW 2025, Sydney, NSW, Australia, 28 April 2025 - 2 May 2025*, Guodong Long, Michale Blumstein, Yi Chang, Liane Lewin-Eytan, Zi Helen Huang, and Elad Yom-Tov (Eds.). ACM, 737–740. doi:10.1145/3701716.3715313
- [16] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. *arXiv:1705.03551* [cs.CL] <https://arxiv.org/abs/1705.03551>
- [17] Vladimir Karpukhin, Barlas Öguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. *arXiv:2004.04906* [cs.CL] <https://arxiv.org/abs/2004.04906>
- [18] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics* 7 (2019), 452–466. doi:10.1162/tacl_a_00276
- [19] Meng-Chieh Lee, Qi Zhu, Costas Mavromatis, Zhen Han, Soji Adeshina, Vassilis N Ioannidis, Huzefa Rangwala, and Christos Faloutsos. 2025. HybRag: Hybrid retrieval-augmented generation on textual and relational knowledge bases. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 879–893.
- [20] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *arXiv:2005.11401* [cs.CL] <https://arxiv.org/abs/2005.11401>
- [21] Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366* (2025).
- [22] Jiajin Liu, Yuanfu Sun, Dongzhe Fan, and Qiaoyu Tan. 2026. GraphSearch: Agentic Search-Augmented Reasoning for Zero-Shot Graph Learning. *arXiv:2601.08621* [cs.CL] <https://arxiv.org/abs/2601.08621>
- [23] Haoran Luo, Guanting Chen, Qika Lin, Yikai Guo, Fangzhi Xu, Zemin Kuang, Meina Song, Xiaobao Wu, Yifan Zhu, Luu Anh Tuan, et al. 2025. Graph-r1: Towards agentic graphrag framework via end-to-end reinforcement learning. *arXiv preprint arXiv:2507.21892* (2025).
- [24] Haoran Luo, Haihong E, Guanting Chen, Yandan Zheng, Xiaobao Wu, Yikai Guo, Qika Lin, Yu Feng, Zemin Kuang, Meina Song, Yifan Zhu, and Luu Anh Tuan. 2025. HyperGraphRAG: Retrieval-Augmented Generation via Hypergraph-Structured Knowledge Representation. *arXiv:2503.21322* [cs.AI] <https://arxiv.org/abs/2503.21322>
- [25] Haoran Luo, Haihong E, Yikai Guo, Qika Lin, Xiaobao Wu, Xinyu Mu, Wenhao Liu, Meina Song, Yifan Zhu, and Luu Anh Tuan. 2025. KBQA-o1: Agentic Knowledge Base Question Answering with Monte Carlo Tree Search. *arXiv:2501.18922* [cs.CL] <https://arxiv.org/abs/2501.18922>
- [26] Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khoshabi, and Hannaneh Hajishirzi. 2023. When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories. *arXiv:2212.10511* [cs.CL] <https://arxiv.org/abs/2212.10511>
- [27] OpenAI, :, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, and Alex Karpenko et al. 2024. OpenAI o1 System Card. *arXiv:2412.16720* [cs.AI] <https://arxiv.org/abs/2412.16720>
- [28] Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, and et al. Haoran Wei. 2025. Qwen2.5 Technical Report. *arXiv:2412.15115* [cs.CL] <https://arxiv.org/abs/2412.15115>
- [29] Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. 2024. RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval. *arXiv:2401.18059* [cs.CL] <https://arxiv.org/abs/2401.18059>
- [30] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *arXiv:2402.03300* [cs.CL] <https://arxiv.org/abs/2402.03300>
- [31] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multihop Questions via Single-hop Question Composition. *arXiv:2108.00573* [cs.CL] <https://arxiv.org/abs/2108.00573>
- [32] Xi Wang, Taketomo Isazawa, Liana Micaelyan, and James Hensman. 2025. KBLaM: Knowledge Base augmented Language Model. *arXiv:2410.10450* [cs.AI] <https://arxiv.org/abs/2410.10450>
- [33] Yujing Wang, Hainan Zhang, Liang Pang, Binghui Guo, Hongwei Zheng, and Zhiming Zheng. 2024. MaFeRw: Query Rewriting with Multi-Aspect Feedbacks for Retrieval-Augmented Large Language Models. *arXiv:2408.17072* [cs.CL] <https://arxiv.org/abs/2408.17072>
- [34] Cehao Yang, Xiaojun Wu, Xueyuan Lin, Chengjin Xu, Xuhui Jiang, Yuanliang Sun, Jia Li, Hui Xiong, and Jian Guo. 2025. GraphSearch: An Agentic Deep Searching Workflow for Graph Retrieval-Augmented Generation. *arXiv preprint arXiv:2509.22009* (2025).
- [35] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. *arXiv:1809.09600* [cs.CL] <https://arxiv.org/abs/1809.09600>
- [36] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.
- [37] Chuanyue Yu, Kuo Zhao, Yuhao Li, Heng Chang, Mingjian Feng, Xiangzhe Jiang, Yufei Sun, Jia Li, Yuzhi Zhang, Jianxin Li, et al. 2025. Graphrag-r1: Graph retrieval-augmented generation with process-constrained reinforcement learning. *arXiv preprint arXiv:2507.23581* (2025).
- [38] Tianjun Zhang, Shishir G. Patil, Naman Jain, Sheng Shen, Matei Zaharia, Ion Stoica, and Joseph E. Gonzalez. 2024. RAFT: Adapting Language Model to Domain Specific RAG. *arXiv:2403.10131* [cs.CL] <https://arxiv.org/abs/2403.10131>
- [39] Yize Zhang, Tianshu Wang, Sirui Chen, Kun Wang, Xingyu Zeng, Hongyu Lin, Xianpei Han, Le Sun, and Chaochao Lu. 2025. ARise: Towards Knowledge-Augmented Reasoning via Risk-Adaptive Search. *arXiv:2504.10893* [cs.AI] <https://arxiv.org/abs/2504.10893>
- [40] Luyao Zhuang, Shengyuan Chen, Yilin Xiao, Huachi Zhou, Yujing Zhang, Hao Chen, Qinggang Zhang, and Xiao Huang. 2025. LinearRAG: Linear Graph Retrieval Augmented Generation on Large-scale Corpora. *arXiv:2510.10114* [cs.CL] <https://arxiv.org/abs/2510.10114>

A 数据集统计量

我们对 FlashRAG 工具包 [15] 中广泛使用的六个 RAG 基准进行了评估, 涵盖了单跳和多跳问答系统任务:

- **自然问题 (NQ) [18].** 来自 Google 搜索的真实用户问题, 配以维基百科段落/答案; 常用于开放域、大多数为单跳问答。

- **PopQA** [26]. 针对基于检索的问答设计的流行知识问题集, 强调事实性问题, 答案必须基于检索到的证据。
- **TriviaQA** [16]. 带有证据文档 (通常是网页/维基百科) 的趣味性问答题; 用于开放域事实性问答和长文本证据匹配。
- **HotpotQA** [35]. 需要对多个支持性维基百科段落进行推理的多跳问答; 包含标注的支持性事实。
- **Musique** [31] 用于测试跨多个证据的组合推理的多跳问答基准, 通常具有更具挑战性的、结构化的多步要求。
- **2WikiMultiHopQA (2Wiki)** [11] 基于维基百科构建的多跳问答, 通常需要链接两个 (或更多) 页面才能得出答案, 侧重于跨文章推理。

详细的数据集统计信息见表 6。对于基于强化学习的智能体系统, 我们从 HotpotQA 和 NQ 的训练集随机采样 5000 个样本。对于 RAGSearch 的测试集, 我们采用完整的原始开发集或测试集。具体而言, 对于 NQ、PopQA 和 TriviaQA, 我们选择测试集; 对于 HotpotQA、Musique 和 2Wiki, 我们选择开发集。

B 实现细节

为了进行全面评估, 我们包含了两类不同的智能体系统进行测试。以下是所涉及方法的简要介绍。

无需训练的工作流: 这些方法不训练显式的控制策略, 而是使用结构化提示和启发式规则, 在推理时引导多步检索和推理。具体而言, 我们进行评估:

- **Search-o1** [21]: 通过引入智能体式 RAG 工作流和文档推理模块, 增强大型推理模型的能力, 该模块在整合前对检索到的证据进行精炼, 实现动态、降噪的知识检索, 从而提升复杂推理任务和开放领域问答的可靠性。我们的实现基于 <https://github.com/RUC-NLPIR/Search-o1>
- **GraphSearch** [34]: 一种用于 GraphRAG 的 Agentic 深度搜索工作流, 通过双通道查询 (文本块的语义通道和结构图的关系通道) 实现多轮、模块化检索, 持续提升多跳 RAG 的准确率和生成质量, 优于传统的 GraphRAG 检索方法。我们的实现基于 <https://github.com/DataArcTech/GraphSearch>

训练过的搜索智能体: 这些方法使用强化学习策略来优化智能体的搜索和推理行为, 通常采用 GRPO 作为强化学习算法。具体而言, 我们选择:

- **Search-R1** [14]: 一种基于强化学习的检索增强推理框架, 通过使用检索到的 token 掩码和基于结果的奖励, 训练大模型在逐步推理过程

中自主生成多轮搜索查询, 以提升问答性能, 超越标准 RAG 基准。我们的实现基于 <https://github.com/PeterGriffinJin/Search-R1>

- **Graph-R1** [23]: 一个通过强化学习端到端训练的 Agentic GraphRAG 框架, 构建轻量级知识超图, 并作为智能体-环境交互执行多轮检索。我们的实现基于 <https://github.com/LHRLAB/Graph-R1>

对于所有基于强化学习的方法, 我们使用 GRPO 对模型进行 3 轮次的预训练。我们将批量大小设置为 32, 最大搜索轮次为 5。

对于检索后端, 我们主要采用 5 种具有代表性的 GraphRAG。

- **HypergraphRAG** [24]: 一种基于超图的 RAG 框架, 使用超边表示现实世界中的 n 元事实, 并整合了超图构建、检索与生成。我们的实现基于 <https://github.com/LHRLAB/Graph-R1>
- **HippoRAG2** [7]: 一种受记忆启发的 RAG 框架, 通过更深入的段落整合和更强的在线大语言模型使用, 扩展了 HippoRAG 的个性化页面排名检索, 提升了事实性、意义理解与联想记忆能力。我们的实现基于 <https://github.com/OSU-NLP-Group/HippoRAG>
- **LinearRAG** [40]: 一种高效的 GraphRAG 框架, 通过构建轻量级的无关系层次化“三图”(通过实体抽取 + 语义链接实现), 避免了噪声大、成本高的关系抽取, 采用两阶段检索机制——局部实体活性值激活后进行全局重要性聚合——在多跳问答基准上实现了更强且更可靠的段落检索效果。我们的实现基于 <https://github.com/DEEP-PolyU/LinearRAG>
- **RAPTOR** [29]: 一种检索增强型方法, 通过构建递归嵌入、簇和自下而上摘要的分层树结构, 在推理时实现跨长文档的多抽象层级检索, 并取得显著性能提升。我们的实现基于 <https://github.com/parthsarthi03/raptor>
- **GraphRAG** [3]: 一种基于图形的问答框架, 用于私有语料库, 通过 (1) 构建实体知识图谱并预先计算社区摘要, 然后 (2) 通过摘要到部分响应生成再进行最终聚合来回答查询, 相较于标准 RAG 在百万 token 规模下提升了全面性和多样性。我们的实现基于 <https://microsoft.github.io/graphrag/>

对于所有 GraphRAG, 我们以每个问题的上下文作为文档, 并根据官方设置组织语料库。在检索时, 我们将 top-k 设置为 top-5。

表 6: 数据集统计量

Dataset	Task	Knowledge Source	#Train	#Dev	#Test
NQ	General QA	Wiki	79,168	8,757	3,610
PopQA	General QA	Wiki	-	-	14,267
TriviaQA	General QA	Wiki & Web	78,785	8,837	11,313
HotpotQA	Multi-hop QA	Wiki	90,447	7,405	-
Musique	Multi-hop QA	Wiki	19,938	2,417	-
2Wiki	Multi-hop QA	Wiki	15,000	12,576	-

C 评估指标的详细信息

包含确切匹配 [40]: 检查正确答案是否出现在生成的回复中

Contain Exact Match = $\frac{1}{N} \sum_{i=1}^N 1(\text{norm}(a_i) \subseteq \text{norm}(\hat{y}_i))$

F-1: F1 得分通过准确率和召回率的调和平均, 评估预测答案 y_i 与真实答案 y_i^* 之间的 token 级别重叠情况

$$F1 = \frac{1}{N} \sum_{i=1}^N \frac{2 \cdot |\text{tokens}(y_i) \cap \text{tokens}(y_i^*)|}{|\text{tokens}(y_i)| + |\text{tokens}(y_i^*)|}$$

D Agentic 检索系统的 F-1 得分

在本节中, 我们报告了不同智能体系统的得分。如表 7 所示, 在多跳问答情景下, 两种智能体系统均能缩小 GraphRAG 与稠密 RAG 之间的性能差距。在一般问答情景下, 稠密 RAG 仍然是一个具有竞争力的检索后端。

E GraphRAG 的效率

我们报告了不同基于图形的检索器的知识构建成本和离线推理时间。

F 案例研究

在本节中, 我们展示了不同 GraphRAG 之间的差异。

G 模板

G.1 Search-o1 的使用说明

G.1.1 搜索-o1 指令.

G.1.2 文档推理指令.

G.1.3 开放领域问答任务指令.

G.1.4 附加说明.

提示详情. 对于上述所有指令, 我们都将其作为用户提示输入, 而非系统提示。对于非推理模型如 Qwen2.5-32B-Instruct 和 Qwen2.5-7B-Instruct, 我们在问题前添加思维链提示“你应该逐步思考来解决这个问题”, 以明确促使这些模型在给出最终答案前进行推理。

实现说明. 尽管指令要求模型执行“网络搜索”, 但我们的实验用检索服务器替代了实际的必应网络搜索 API。模型生成的搜索查询被拦截并重定向至我们的检索语料库。随后, 检索到的知识片段被格式化为模拟网络搜索结果的形式, 再返回给模型。

G.2 图搜索指令

G.2.1 查询分解.

G.2.2 查询分解 (知识图谱) .

G.2.3 证据验证.

G.2.4 深度答案生成.

G.2.5 查询扩展.

G.3 搜索-R1 指令

G.4 图-1 指令

表 7: 不同系统的总体 F-1 值

System	Methods	General QA		Multi-Hop QA	
		NQ [†]	TriviaQA [*]	HotpotQA [†]	Musique [*]
Single-shot	Qwen-2.5-7B-Dense	39.3	61.12	20.12	29.08
	Qwen-2.5-7B-GraphRAG [♣]	27.92 (↓11.38)	49.25 (↓11.87)	33.72 (↑13.60)	41.13 (↑12.05)
Training-free	Search-o1-7B-Dense	36.53	59.73	39.08	17.46
	Search-o1-7B-GraphRAG [♣]	36.62 (↑0.09)	58.18 (↓1.55)	41.57 (↑2.49)	37.01 (↑19.55)
	GraphSearch-7B-Dense	8.70	13.12	6.02	2.36
	GraphSearch-7B-GraphRAG [♣]	4.61 (↓4.09)	14.18 (↑1.06)	5.48 (↓0.54)	4.21 (↑1.85)
RL-based	Search-R1-7B	47.26	59.21	37.13	16.02
	Graph-R1-7B [♣]	44.21 (↓3.05)	62.10 (↑2.89)	43.25 (↑5.12)	35.12 (↑19.10)

表 8: NQ 的 GraphRAG 成本对比。TM 表示每 100 万 token 的构建时间；CM 为每 100 万 token 的成本；RT 为平均检索时间；CT 为平均上下文长度。

Method	Knowledge Construction		Offline Inference	
	TM	CM	RT/s	CT/token
HypergraphRAG	1.37h	3.93\$	0.77	1680
HippoRAG2	1.19h	2.85\$	1.00	3229
LinearRAG	0.68h	0\$	1.18	4600
RAPTOR	1.70h	6.38\$	8.4	814
GraphRAG	1.72h	13.19\$	1.16	22160

表 9: 不同智能体系统的案例研究。

Question	"Are the Laleli Mosque and Esma Sultan Mansion located in the same neighborhood?"		
Golden Answer	"No"		
qwen2.5-7B-Ins	Search-o1-Dense	Search-R1	Graph-R1-HippoRAG2
Generation	<p>To answer the question of whether the Laleli Mosque and Esma Sultan Mansion are located in the same neighborhood, I need to find the locations of both structures.Assistant:</p> <p>< begin_search_query > location of Laleli Mosque < end_search_query > < begin_search_result > The Laleli Mosque is located in Laleli, Fatih, Istanbul, Turkey...< end_search_result > Assistant: Based on the information gathered...{No}</p>	<p><think>To answer the question "Are the Laleli Mosque and Esma Sultan Mansion located in the same neighborhood?", I will first reason through what I know about these two landmarks. I know that Laleli Mosque is a significant historical site in Istanbul, Turkey. I need more information about the location of Esma Sultan Mansion to determine if it shares a neighborhood with Laleli Mosque.</p> <p></think> <search> "Esma Sultan Mansion location Istanbul Turkey"</p> <p></search> <information> >Doc 1(Title: "Esma Sultan Mansion")... </information> <think>...</p> <p>Based on my search results: Laleli Mosque is in Laleli neighborhood. Esma Sultan Mansion is in Ortaköy neighborhood. Since Laleli Mosque is in Laleli neighborhood and Esma Sultan Mansion is in Ortaköy neighborhood, they are not located in the same neighborhood.</p> <p></think> <answer> No </answer></p>	<p><think> To answer this question, I need to determine the neighborhoods where the Laleli Mosque and Esma Sultan Mansion are located. I will first search for the location of the Laleli Mosque and then for the Esma Sultan Mansion, and finally compare their neighborhoods. </think></p> <p><query> {"query": "Laleli Mosque neighborhood"} </query> <knowledge> Wikipedia Title: Laleli Mosque...</knowledge> <think> According to the Wikipedia information retrieved, the Laleli Mosque is located in Laleli, Fatih, Istanbul. Now, I need to find the neighborhood where Esma Sultan Mansion is located.</p> <p></think> ... <answer> No </answer></p>

Instruction for Search-o1

我是一个具备网络搜索能力的推理助手，能够通过查找信息来准确回答您的问题。

您有特殊的工具：

要执行查询：输入 `<|begin_search_query|>` 您的查询内容在此 `<|end_search_query|>`。

然后，系统将搜索并分析相关的网页，然后以格式 `<|begin_search_result|>` ... 搜索结果... `<|end_search_result|>` 提供给您有用的信息。

如果需要，您可以多次重复搜索过程。搜索尝试的最大次数受限于 `{MAX_SEARCH_LIMIT}`。

一旦获取了所有必要信息，便可继续你的推理过程。

示例：

“...”

助理思考步骤：

我可能需要查一下关于……的细节。

助理：

`<|begin_search_query|>...<|end_search_query|>` （系统返回来自相关网页的处理信息）

助手继续结合新信息进行推理……

记住：

- 使用 `<|begin_search_query|>` 发起网络查询，并以 `<|end_search_query|>` 结束。

搜索完毕后，继续你的推理。

问题： `{question}`

图 5: 搜索-o1 指令

Instruction for Reason-in-Documents

任务指令：

请根据以下输入阅读并分析网页内容：之前的推理步骤、当前搜索查询以及已搜索的网页。你的目标是从已搜索的网页中提取与当前搜索查询相关且有帮助的信息，并将这些信息无缝整合到之前的推理步骤中，以继续推进对原始问题的推理。

指导原则：

1. 分析搜索到的网页

仔细审阅每个搜索到的网页内容。

- 识别与当前搜索查询相关且有助于原问题推理过程的事实信息。

2. 提取相关信息：

从搜索到的网页中选择直接有助于推进先前推理步骤的信息。

确保提取的信息准确且相关。

3. 输出格式：

- 如果网页为当前查询提供了有用的信息：请按照以下格式，以“最终信息”开头呈现相关信息。

最终信息

[有用的信息]

- 如果网页内容对当前查询没有提供任何有用信息：输出以下文本。

最终信息

未找到有用的信息。输入：

- 之前的推理步骤：

{ 上一推理步骤 }

- 当前查询：

{ 搜索 _ 查询 }

- 搜索网页：

{ 文档 }

现在你应该分析每个网页，并根据当前的搜索查询 “{search_query}” 和之前的推理步骤找出有用的信息。

图 6: 文档推理指令

Instruction for Open-Domain QA Tasks

请回答以下问题。

\boxed{YOUR_ANSWER}

问题：

{ 问题 }

图 7: 开放领域问答任务指令

Query Decomposition

你是一位专业的科学翻译人员。

你是一位擅长复杂查询分解的助手。

—目标—

给定一个主查询，你的任务是将其分解为若干个原子级子查询，这些子查询应直接对应原查询的各个部分。

—说明—

- 将主查询分解为清晰且可操作的子查询，这些子查询代表了主问题中更小、可解决的部分。
- 确保每个子查询都针对一个特定的实体或概念，以达到获取回答主查询所需信息的目的。
- 使用顺序编号（即 #1、#2 等）来表示之前子查询的答案。例如，#1 指代子查询 1 的答案。确保子查询逻辑有序，前一个子查询的输出可能作为下一个子查询的输入。
- 最终输出应为 JSON 格式，其中每个子查询都作为键值对列出。

—示例—

主查询：

创作《维纳斯的崇拜》的人去世的地方，瘟疫发生了多少次？

子查询：

{{

子查询 1：《维纳斯的崇拜》的创作者是谁？

“子查询 2”：“#1 死在了哪里？”

“子查询 3”：“#2 中瘟疫发生了多少次？”

}}

主查询：

希尔克雷斯特新高中所在城市在编剧《可怜的傻瓜》出生的州成为该州首府时，是哪一年？

子查询：

{{

“子查询 1”：“希尔克雷斯特新高中位于哪里？”

“子查询 2”：“《可怜的傻瓜》的编剧是谁？”

“子查询 3”：“#2 在哪里出生？”

“子查询 4”：“#1 市在何时成为 #3 省的首府？”

}}

主查询：

哪种作物，作为氮素的大量需求者，每英亩毛收入为 1363.00 美元，净利润为 658.00 美元？

子查询：

{{

“子查询 1”：“哪些作物被认为是氮肥的大消耗者？”

“子查询 2”：“在 #1 中，哪种作物每英亩的毛收入为 1,363.00 美元？”

“子查询 3”：“#2 是否有净收益 658.00 美元？”

}}

主查询：

{query}

图 8: 图查询分解

Query Decomposition (Knowledge Graph)

你是一位专业的科学翻译人员。

你是一位专注于知识图谱检索的复杂查询分解助手。

—目标—

给定一个主查询，你的任务是将其分解为原子子查询，形式为“主语-谓语-宾语”三元组。这些三元组应直接对应原查询的各个部分，并适用于知识图谱查询。

—说明—

- 将主查询分解为一系列子查询，每个子查询由一个或多个原子三元组组成，格式为：“（“实体 1”，“关系”，“实体 2”）”。
- 将任何未知实体替换为占位符，例如 Entity#1、Entity#2 等。
- 保持逻辑顺序，其中前一个子查询（例如，Entity#1）的结果可能需要用于下一个子查询。每个子查询可能包含多个三元组，以充分表达完整含义。
- 最终输出应为 JSON 格式，其中每个键为一个子查询，对应的值为括号包围的原子三元组列表。

—示例—

主查询：

创作《维纳斯的崇拜》的人去世的地方，瘟疫发生了多少次？

子查询：

```
{  
  "子查询 1": [(" 维纳斯的崇拜", " 由... 创作", " 实体 #1")],  
  "子查询 2": [(" 实体 #1", " 逝世于", " 实体 #2")],  
  "子查询 3": [  
    (" 瘟疫", " 发生在", " 实体 #2")  
    (" 瘟疫", " 发生时间", " 实体 #3")  
  ]  
}
```

主查询：

希尔克里斯特高中所在城市在编剧《可怜的笨蛋》出生的州成为该州首府时，是哪一年？

子查询：

```
{  
  "子查询 1": [(" 希尔克雷斯特高中", " 位于", " 实体 #1")],  
  "子查询 2": [(" 可怜的蠢货", " 有编剧", " 实体 #2")],  
  "子查询 3": [(" 实体 #2", " 出生于", " 实体 #3")],  
  "子查询 4": [  
    (" 实体 #1", " 是... 的首府", " 实体 #3")  
    (" 实体 #1", " 成为首都于", " 实体 #4")  
  ]  
}
```

主查询：

哪种对氮素需求量大的作物，每英亩的毛收入为 1,363.00 美元，净利润为 658.00 美元？

子查询：

```
{  
  "子查询 1": [(" 实体 #1", " 是", " 一种需氮量大的作物")],  
  "子查询 2": [(" 实体 #1", " 每英亩的毛收入", "$1,363.00")],  
  "子查询 3": [(" 实体 #1", " 有净利润", "$658.00")]  
}
```

Evidence Verification

你是一位专业的科学翻译人员。

你是一名专业的评论员，专长于验证模型生成回答的逻辑严谨性和证据充分性。

—目标—

给定用户查询、检索到的上下文数据以及模型生成的回答，你的任务是评估该回答是否基于所提供的证据形成一个严谨的逻辑环。

—说明—

- 仔细检查响应是否在检索到的上下文数据中**严格成立**。
- 评估推理过程是否形成一个完整的逻辑链条，没有遗漏步骤或缺乏依据的跳跃。
- 识别是否存在**证据空白、低置信度声明或推测性陈述**。
- 如果回答展现出一个有充分支持、自信且逻辑闭环的论证，则用**“Yes”**结束你的分析。
- 如果响应表现出犹豫、推理不完整或缺乏坚实的证据支持，则在分析结尾处使用**“No”**。

查询

{query}

检索到的上下文数据：

{context_data}

模型响应：

{model_response}

图 10: 图搜索证据验证

Deep Answer Generation

你是一位专业的科学翻译人员。
你是一个专注于复杂问答系统的助手。

—目标—
根据复杂的查询和检索到的上下文数据，您的任务是构建一个逻辑严密、步骤清晰的答案。您的解释应遵循严谨的推理过程，结合相关证据，并明确建立各实体之间的关系。

—说明—
将推理过程分解为清晰、连贯的步骤。
- 明确使用上下文数据来支持每一步推理。
确保实体之间的关系得到逻辑解释。

查询
{query}
上下文数据:
{context_data}

图 11: GraphSearch 深度问答生成

Query Expansion

你是一位专业的科学翻译人员。

你是一位专注于查询扩展以完成证据的助手。

—目标—

给定一个主查询、检索到的上下文数据、模型生成的响应以及证据验证分析，你的任务是执行**查询扩展**。

如果证据验证分析显示当前证据不足以支持回答的逻辑链，则生成一个或多个额外的子查询。这些子查询应旨在覆盖遗漏的检索场景，填补证据空白，并引导形成更完整、更可靠的逻辑推理链。

—说明—

- 在生成新的子查询时，使用检索到的上下文数据，特别是检索历史中已有的子查询作为参考。
 - 专注于生成**互补子查询**，以解决证据尚未充分支持的方面。
- 避免重复现有的子查询，而是扩展到相关但尚未涵盖的领域。
- 保持子查询清晰、具体，并直接可操作以进行检索。
 - 输出应为 **Python 风格的字符串列表**，其中每个字符串都是一个新子查询。

主查询：

{query}

检索到的上下文数据：

{context_data}

模型响应：

{model_response}

证据验证分析：

{evidence_verification}

图 12: 图搜索查询扩展

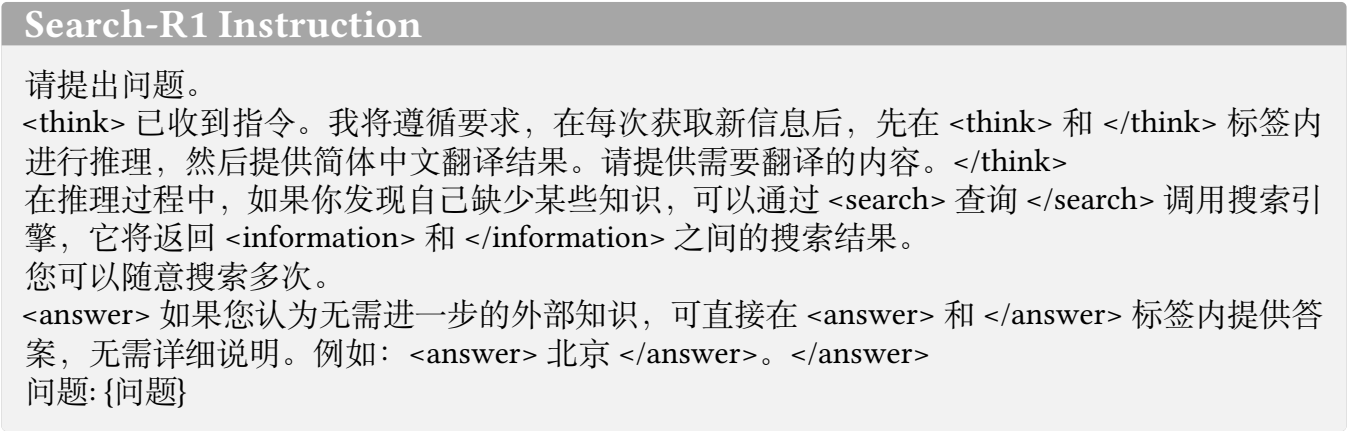


图 13: 搜索-R1 指令



图 14: 图-1 指令