GraphRAG

RAG为llm提供了从某些数据源检索到的信息，作为其生成答案的依据。也就是将根据上下文相关信息进行检索，基于检索到的知识指导llm进行生成。

📍 RAG能一定程度上缓解大模型面临的问题：

- **幻觉**：通过检索相关的文档，减少生成内容幻觉，提供更多的可解释性。
- **知识实时更新**：RAG模型的非参数化记忆可以轻松更新，以反映当下世界的知识变化，无需对模型重新训练。
- **数据隐私**：RAG通过本地化部署私有知识库，限定模型仅访问相关内部数据，从而有效防止敏感信息外泄。

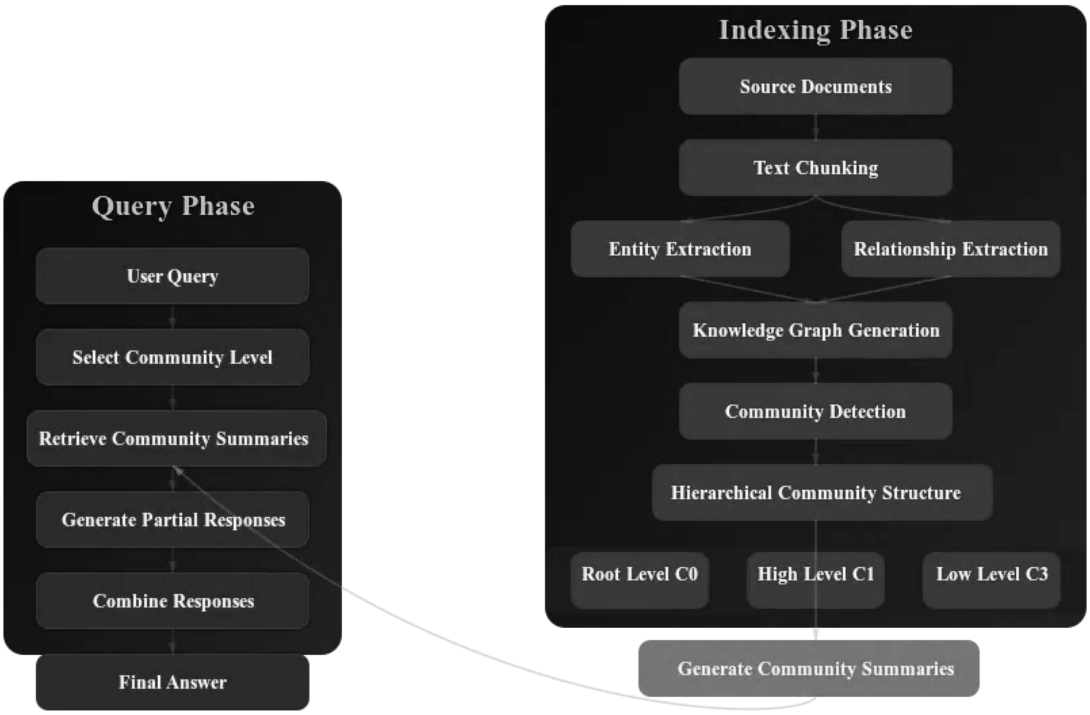
但是基于RAG的大模型应用面临的问题：

- **平面检索**: RAG 将每个文档作为一个独立的信息。想象一下，阅读单独的书页，却不知道它们之间是如何连接的。这种方法错过了不同信息片段之间更深层次的关系。
- **语境缺陷**: 如果不理解关系和语境，人工智能可能会提供不连贯的反应。这就像有一个图书管理员，他知道在哪里可以找到书，但是却不知道书中的故事之间的联系。
- **可伸缩性问题:** 随着信息量的增长，寻找正确的文档变得越来越慢，也越来越复杂，就像试图在不断扩展的库中找到一本特定的书一样。

GraphRAG 不使用非结构化的文本，而是利用**知识图谱**，利用图结构捕捉数据中的实体、关系及复杂依赖，从而更高效地检索相关信息并生成准确答案。GraphRAG 的一大特色是利用图机器学习算法针对数据集进行**语义聚合和层次化分析**，因而可以回答一些相对高层级的抽象或总结性问题, 这一点恰好是常规 RAG 系统的短板(例如：用户提问一个问题，需要全局搜索整个数据集，而不是搜索相似性片段，在这种场景下rag性能比较差)。

GraphRAG 基本步骤为：

- 将输入语料库切分为一系列文本单元，利用LLM创建对源数据中所有**实体和关系**的引用，然后使用这些引用来创建 LLM 生成的知识图谱。
- 通过图算法检测**社区结构**（进行创建自下而上的聚类），构建分层结构。使用 LLM 为每个社区生成自然语言**摘要**，帮助全面理解数据集。
- 在用户查询时，可以进行**局部搜索**或者**全局搜索**。



建立知识图谱索引

知识图谱是真实实体及其之间关系的结构化表示。回忆数据结构的知识，图是由节点和边构成的：

- **实体（节点）**：表示关键的概念。
- **关系（边）**：表示实体之间的关系。

知识图谱的构建流程：

1. **输入文档**：GraphRAG 将一组文本文档的chunking作为输入，这些输入一般存储在**图形数据库**中。

> ⚽ **常见的图数据库：**
>
> - **Neo4j**：最流行的属性图数据库，具有“无索引邻接”特性，每个顶点维护着指向其邻接顶点的直接引用，图导航操作代价与图大小无关，仅与图的遍历范围成正比。支持ACID事务，适用于多种应用场景，包括社交网络、推荐系统、生物信息学等。
> - **ArangoDB**：多模型数据库，支持图、文档和键值存储。允许在单个数据库中同时使用多种数据模型，适用于各种不同的应用场景。
> - **TigerGraph**：高性能的分布式图数据库，支持复杂的图查询和分析，以及内置的图算法库。适用于处理大规模的图数据。

2. **实体和语义关系提取**: LLM 用于从输入文档中自动提取实体(人、地点、概念)以及它们之间的关系。这是使用**命名实体识别**和**关系提取**等自然语言技术完成的。
3. **知识图谱生成**: 利用提取的实体和关系构造知识图谱数据结构，通过**知识融合**对数据进行逻辑归属和冗杂/错误过滤。
4. **分层社区检测**: 使用图算法（例如Leiden），找出紧密相关的实体群体形成的**社区**。这些社区代表了跨越多个文档的主题或主题。社区按等级组织，高层次社区包含低层次的子社区。

5. **生成信息摘要**：利用LLM为每个社区生成摘要，包括社区中的实体、关系。此外再将社区的分层结构保留在分层摘要中。

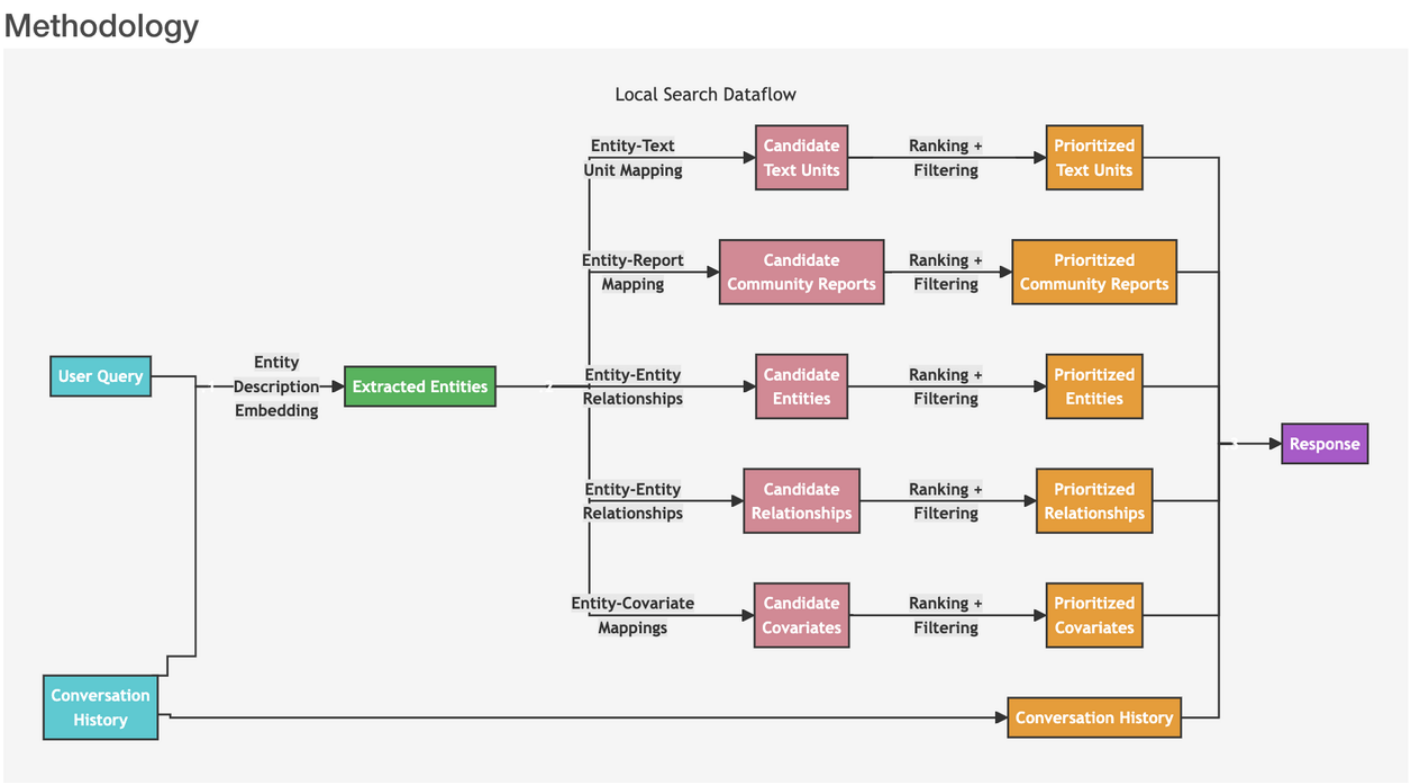
检索增强生成

主要有两种检索方式：

- **局部检索**: 局部搜索旨在理解和回答关于特定实体及其相关概念的详细问题。将用户查询与社区摘要进行匹配，以查找最相关的高社区，在社区中检索相关信息。
- **全局检索**: 全局搜索是为了理解和回答关于整个文档集的综合性问题，如“数据中的前N个主题是什么？”这类需要跨文档聚合信息的查询。利用知识图的分层结构对整个数据集进行搜索，以查找回答查询所需的特定实体、关系和信息。这包括了遍历知识图谱和组合来自多个社区的信息，可以提供全面的响应。

局部检索

当用户提出关于特定 实体（如人名、地点、组织等）的问题时，应该采用局部搜索的方法，如下图所示：



1. **用户查询**：首先，系统接收用户查询，这可能是一个简单的问题或更复杂的查询。
2. **搜索相似实体**：系统从知识图中识别出与用户输入语义相关的一组 实体。这些 实体 作为进入知识图谱的入口点。这一步骤中使用像 Milvus 这样的向量数据库进行文本相似性搜索。
3. **实体-文本单元映射**：提取的文本单元被映射到相应的 实体，移除原始的文本信息。
4. **实体-关系提取**：这一步提取关于 实体 及其相应关系的特定信息。

5. **实体-协变量（Covariate）映射**：这一步将 实体 映射到它们的协变量，这可能包括统计数据或其他相关属性。
6. **实体-社区 报告映射**：社区 报告被整合到搜索结果中，纳入一些全局信息。
7. **利用对话历史**：如果有对话历史，系统使用对话历史来更好地理解用户的意图和上下文。
8. **生成响应**：最后，系统根据前几步生成的经过过滤和排序的数据生成并响应用户查询。

全局检索

针对用户提出的需要全局搜索整个数据集的全局性问题，提出了一种基于GraphRAG的回复方法，大致分为6个步骤：

1. **源文档 → 文本块**

- **粒度**：将源文档的文本分割成块。
- **权衡**：更长的块需要更少的 LLM 调用，但可能因为较长的上下文窗口而降低召回率。
- **示例**：在 HotPotQA 数据集上，600 token的块大小提取的实体引用几乎是 2400 token块大小的两倍。

2. **文本块 → 元素实例**

- **目标**：从文本块中识别和提取图节点和边。使用 LLM 提示识别实体和关系，输出限定元组。
- **定制化**：可以通过为LLM提供少量领域特定的示例来定制提示, 以适应不同的知识领域(如科学、医学、法律等)。
- **协变量提取**: 支持二级提取提示,用于提取与提取的节点实例相关的其他协变量,如实体相关的声明、主题、对象、类型、描述、源文本范围以及开始和结束日期。
- **多轮提取**：在不牺牲块大小的情况下检测到更多实体。

3. **元素实例 → 元素摘要**

- **摘要**：LLM 抽象并总结文本中的实体、关系和声明。
- **处理重复**：LLM可能无法始终以统一的格式提取同一实体的引用,可能会产生重复的实体元素。但由于检测到密切相关的实体及其摘要，加上LLM可以理解多种名称变体对应的共同实体, 只要这些变体与一组密切相关的实体有足够的连接性, 整体方法就能够应对这种变体。

4. **元素摘要 → 图社区**

- **图建模**：创建一个无向加权图，其中节点是实体，边是关系。
- **社区检测**：使用 Leiden 算法将图分割成层次化社区，将具有较强内部连接的节点划分为社区，实现高效的全局摘要。

5. **图社区 → 社区摘要**

- **摘要创建**：为每个社区生成报告式摘要。

- **实用性**：摘要有助于理解数据集的全局结构和语义，辅助回答全局查询。用户可以浏览不同层级的社区摘要, 寻找感兴趣的一般主题, 然后深入到较低层级的摘要以获取更多细节。

6. **社区摘要 → 社区回答 → 全局回答**

- **查询社区摘要**：首先定位用户查询需要那一层级的社区摘要进行回复，然后在这一层级检索相关的社区摘要。
- **社区摘要处理**：社区摘要被随机打乱并划分为预设大小的块。这确保相关信息分布在各个块中, 而不是集中(并可能丢失)在单个上下文窗口中。
- **社区回答映射**：为每一块社区摘要并行生成社区回答，利用LLM 生成有用程度分数。
- **归纳全局回答**：按照有用程度得分降序对中间社区答案进行排序, 并迭代地添加到新的上下文窗口中, 直到达到token限制。

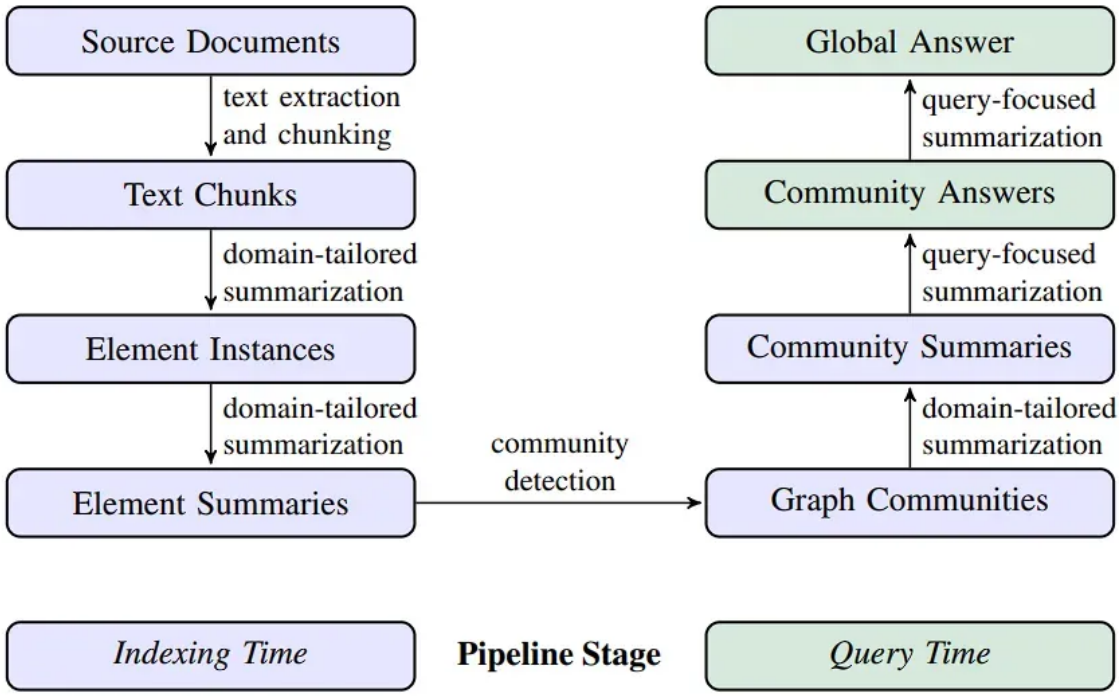


Figure 1: Graph RAG pipeline using an LLM-derived graph index of source document text. This index spans nodes (e.g., entities), edges (e.g., relationships), and covariates (e.g., claims) that have been detected, extracted, and summarized by LLM prompts tailored to the domain of the dataset. Community detection (e.g., Leiden, Traag et al., 2019) is used to partition the graph index into groups of elements (nodes, edges, covariates) that the LLM can summarize in parallel at both indexing time and query time. The "global answer" to a given query is produced using a final round of query-focused summarization over all community summaries reporting relevance to that query.

参考：

- 《一文带你解读GraphRAG：什么是GraphRAG？GraphRAG 的应用场景、工作原理，GraphRAG的实现示例》
- 《深度解读 GraphRAG：如何通过知识图谱提升 RAG 系统》
- 《From Local to Global: A GraphRAG Approach to Query-Focused Summarization》
- 《Welcome to GraphRAG》

GraphRAG 的主要优点有：

- **结构化知识表示**: GraphRAG 使用**知识图谱**来表示信息、捕获实体、关系和层次结构，更准确的理解上下文语义
- **高效处理**: 在知识图谱中将数据预处理可以降低计算成本，并且与传统的 RAG 方法相比，可以更快地检索。
- **多方面查询处理**: GraphRAG 可以通过综合来自知识图谱多个部分的相关信息来处理复杂的多方面查询。
- **可解释性**: 与大模型的黑盒输出相比，GraphRAG 中的结构化知识表示提供了更高的透明度和可解释性。

特点	局部搜索（Local Search）	全局搜索（Global Search）
目的	理解和回答关于特定实体及其相关概念的详细问题	理解和回答关于整个文档集的综合性
优化原理	在特定的子图或领域内进行搜索，聚焦于相关节点和边，查找与当前查询最相关的信息	在整个图谱或大范围的图谱中进行搜型语言模型（LLM）生成的社区报告先总结了数据集的语义结构
搜索范围	限于局部子图，可能只包含少数几个节点和关系	涉及整个知识图谱，搜索范围广泛，
效率	搜索范围较小，通常速度更快，计算量较低	搜索范围大，可能需要更多的时间和信息更为全面
信息相关性	聚焦于与查询最相关的局部信息，减少不相关信息的干扰	涉及更多的信息，可能包含一些冗余盖面广，但可能含有不相关部分
适用场景	需要理解文档中提到的特定实体的问题，例如“牛顿运动定律如何影响其他研究？”	需要跨文档聚合信息的查询，例如“要主题是什么？”