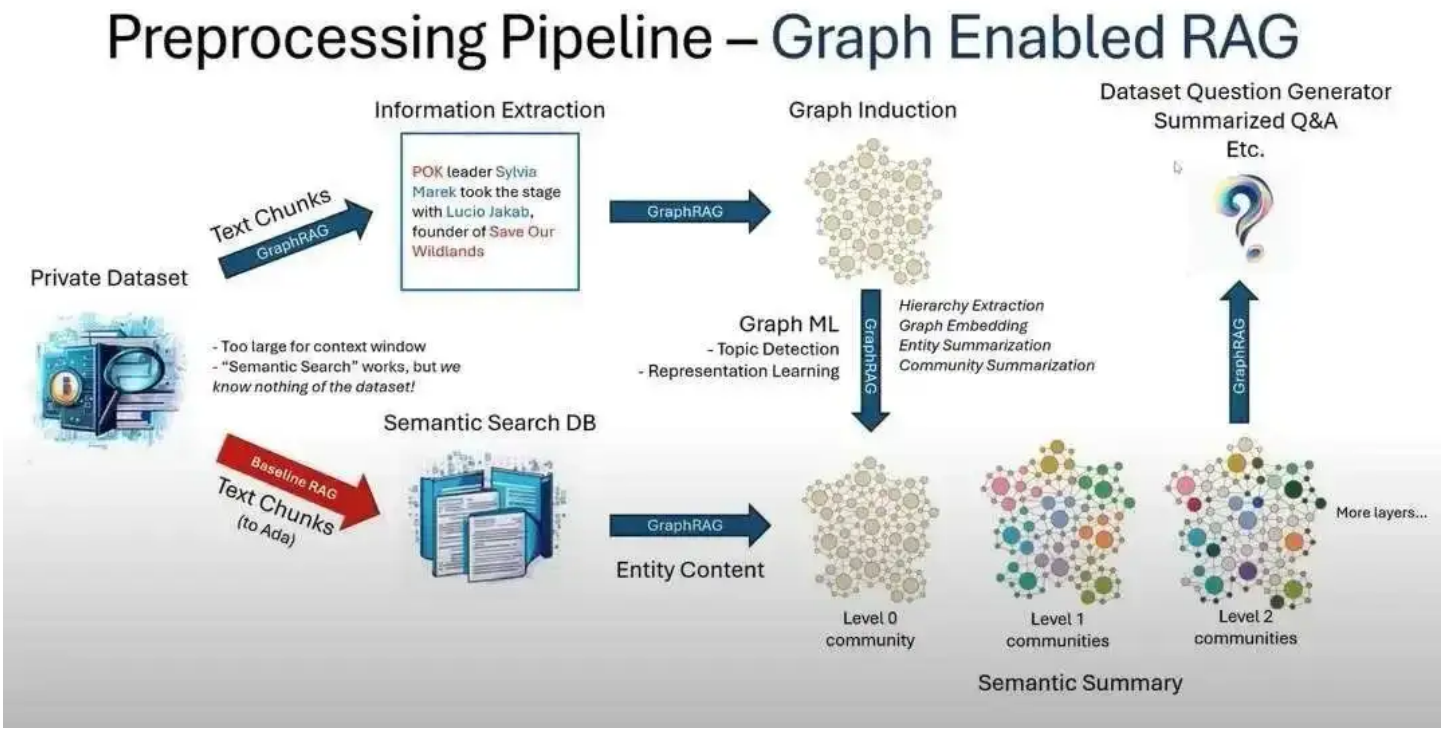
GraphRAG解读

一.概述

微软在7 月 2 日开源了 GraphRAG，一种基于图的检索增强生成 (RAG) 方法，可以对私有或以前未见过的数据集进行问答。通过 LLM 构建知识图谱结合图机器学习，GraphRAG 极大增强 LLM 在处理私有数据时的性能，同时具备连点成线的跨大型数据集的复杂语义问题推理能力，其基于前置的知识图谱、社区分层和语义总结以及图机器学习技术可以大幅度提供此类场景的性能。

GraphRAG 方法可以归结为：利用大型语言模型 (LLMs) 从数据来源中提取知识图谱；将此图谱聚类成不同粒度级别的相关实体社区；对于 RAG 操作，遍历所有社区以创建“社区答案”，并进行缩减以创建最终答案。



GraphRAG的核心就是两个图，一个是文档图谱，一个是文档内部的实体关系图谱。

文档（Document）表示系统输入的文档。这些可以代表CSV中的单独行或单独的.txt文件；

文本单元（TextUnit）表示待分析的文本块。这些块的大小、重叠以及是否遵守任何数据边界可以进行配置。

实体（Entity）表示从文本单元中提取的实体。这些代表人、地点、事件或您提供的其他实体模型；

关系（Relationship）表示两个实体之间的关系；

协变量（Covariate）表示提取的声明信息，其中包含可能有时间限制的关于实体的陈述；

社区报告（Community Report）表示一旦生成实体，会对它们执行层次化的社区检测，并为这个层次结构中的每个社区生成报告；

节点（Node）：包含已嵌入和聚类的实体和文档的渲染图视图的布局信息。

1 索引构建过程

文本文档被转换为GraphRAG知识模型，这个过程包括以下几个主要阶段：

1. 文本单元切分（Compose TextUnits）：输入文档被转换为文本单元，这些文本单元用于图提取技术，并作为提取知识项的源引用。

2. 图谱抽取（Graph Extraction）：分析每个文本单元，提取实体、关系和声明，该阶段包括多个步骤，包括实体&关系抽取、实体&关系总结、实体消歧、声明抽取。其中，实体和关系总结(Entity & Relationship Summarization)将实体及关系这些列表总结成一个实体和关系的单一描述，通过要求LLM提供一个简短的总结来完成，从而实现所有的实体和关系都有一个简洁的单一描述。

2. 图谱增强（Graph Augmentation）：理解实体的社区结构，并通过社区检测和图嵌入来增强图谱。在有了一个可用的实体和关系图之后，希望理解它们的社区结构，并用额外的信息增强这张图，通过两个步骤来完成：社区检测和图嵌入。这些步骤提供了明确（社区）和隐含（嵌入）的方式来理解我们图的拓扑结构。

- 社区检测这一步，使用层次化的Leiden算法生成实体社区的层次结构，对上一步构建好的图应用递归的社区聚类，直到我们达到社区大小的阈值，可以提供一种在不同粒度级别上导航和总结图的方法。
- 图嵌入使用Node2Vec算法可以生成向量表示，可以在查询阶段提供一个额外的向量空间，用于搜索相关概念。

3. 社区摘要（Community Summarization）：生成每个社区的报告，提供对图的高层次理解。即在社区数据的基础上进一步构建，并为每个社区生成报告。这为在图的不同粒度级别上提供了对图的高层次理解。例如，如果社区A是最高层级的社区，我们将得到一个关于整个图的报告。如果社区是低层级的，将得到一个关于本地集群的报告。

- 首先，生成社区报告，在这一步中，使用LLM为每个社区生成摘要，报告包含执行概览，并引用社区子结构中的关键实体、关系和声明。
- 其次，总结社区报告，每个社区报告随后通过LLM进行总结，以供简写使用。
- 最后，社区嵌入，通过生成社区报告、社区报告摘要和社区报告标题的文本嵌入来生成社区的向量表示。

4. 文档处理（Document Processing）：对文档进行表示，并形成文档图

- 首先是链接到文本单元，将每个文档与第一阶段创建的文本单元关联起来，能够理解哪些文档与哪些文本单元相关。
- 其次是文档嵌入，使用文档切片的平均嵌入来生成文档的向量表示。具体地，重新分块文档，不重叠块，然后为每个块生成嵌入，创建这些块的加权平均值，按token计数加权，并将其用作文档嵌入，然后基于这种文档表示，能够理解文档之间的隐含关系，并帮助生成文档的网络表示。

2 两个应用场景

1. 全局搜索：通过利用社区摘要，对整个语料库进行整体问题的推理，利用大型语言模型（LLM）生成的知识图谱来组织和聚合信息，以回答需要跨数据集聚合信息的查询。

传统的RAG模型在处理需要跨数据集聚合信息的查询时表现不佳，例如“数据中的前5个主题是什么？”这类问题。这是因为传统RAG模型依赖于向量搜索来找到语义上相似的文本内容，而没有明确的查询来指导它找到正确的信息。GraphRAG可以回答这类问题，因为由LLM生成的知识图谱的结构告诉我们整个数据集的结构（以及主题）。这允许私有数据集被组织成有意义的语义集群，这些集群是预先总结过的。

全局搜索方法使用LLM生成的社区报告集合作为上下文数据，以map-reduce方式生成响应。在map步骤中，社区报告被分割成预定义大小的文本块。每个文本块用于生成包含点列表的中间响应，每个点都有相应的数值评分，表示该点的重要性。在reduce步骤中，从中间响应中筛选出最重要的点，并将它们聚合起来，作为生成最终响应的上下文。

5. 局部搜索：通过扩展到特定实体的邻居和相关概念，对特定实体进行推理。

这种场景旨在通过结合知识图谱中的结构化数据和输入文档中的非结构化数据，增强大型语言模型（LLM）在查询时的上下文，从而更好地回答涉及输入文档中特定实体的问题。即用的图谱数据跟非结构化文本进行的增强，做的基于实体的推理，利用知识图谱中的结构化数据和输入文档中的非结构化数据，以在查询时为LLM提供与查询相关的实体信息。

GraphRAG首先识别与用户输入语义相关的知识图谱中的实体集合。这些实体作为访问知识图谱的入口，可以提取更多相关信息，如相关实体、关系、实体协变量和社区报告。此外，还从原始输入文档中提取与已识别实体相关联的相关文本块。然后，这些候选数据源将被优先排序和过滤，以适应预定义大小的单个上下文窗口，用于生成对用户查询的响应。

3 结语

GraphRAG的核心就是围绕全局搜索跟局部搜索两个应用场景做的优化，但串联了实体识别、实体关系抽取、社区聚类等算法，会存在误差传播，为了做这种图谱可能需要花费很大精力，也不一定会奏效。不过其通过文档内部，文档外部做图谱的思路，以及基于此做的嵌入作为补充也是值得借鉴的方法。

二.详细描述

GraphRAG 深入解析

GraphRAG[2] 是一种结构化的、分层的检索增强生成 (RAG) 方法，不同于使用纯文本片段的简单语义搜索方法。GraphRAG 流程包括从原始文本中提取知识图谱、构建社区层次结构、为这些社区生成摘要，然后在执行基于 RAG 的任务时利用这些结构。

检索增强生成 (RAG) 是一种使用真实世界的信息改进 LLM 输出的技术。这种技术是大多数基于 LLM 的工具的重要组成部分，大多数 RAG 方法使用向量相似性作为搜索技术，我们称之为 Baseline RAG。GraphRAG 使用知识图谱在推理复杂信息时大幅提高问答性能。RAG 技术在帮助 LLM 推理私有数据集

方面显示出良好的前景- 私有数据集是 LLM 未经过训练且从未见过的数据，例如企业的专业研究、商业文档或通信。Baseline RAG 的创建是为了帮助解决这个问题，但我们观察到 Baseline RAG 表现非常差的情况。例如：

- · 基线 RAG 难以将各个点连接起来。当回答问题需要通过共享属性遍历不同的信息片段以提供新的综合见解时，就会发生这种情况。
- · 当被要求全面理解大型数据集合甚至单个大型文档中的总结语义概念时，基线 RAG 的表现不佳。

为了解决这一问题，技术社区正在努力开发扩展和增强 RAG 的方法。微软研究院的新方法 GraphRAG 使用 LLM 根据输入语料库创建知识图谱。该图谱与社区摘要和图形机器学习输出一起用于增强查询时的提示。GraphRAG 在回答上述两类问题方面表现出了显著的进步，表现出的智能或掌握程度优于之前应用于私有数据集的其他方法。

概述

GraphRAG 建立在微软之前使用图机器学习的研究[3]和工具[4]的基础上。GraphRAG 流程的基本步骤包含索引和查询两部分。

索引

1. 将输入语料库切分为一系列 TextUnit，这些 TextUnit 作为其余过程的可分析单元，并在我们的输出中提供细粒度的参考。
2. 使用 LLM 从 TextUnits 中提取所有实体、关系和关键声明。
3. 使用莱顿算法[5]对图表进行层次聚类。
4. 自下而上地生成每个社区及其组成部分的摘要。这有助于整体理解数据集。

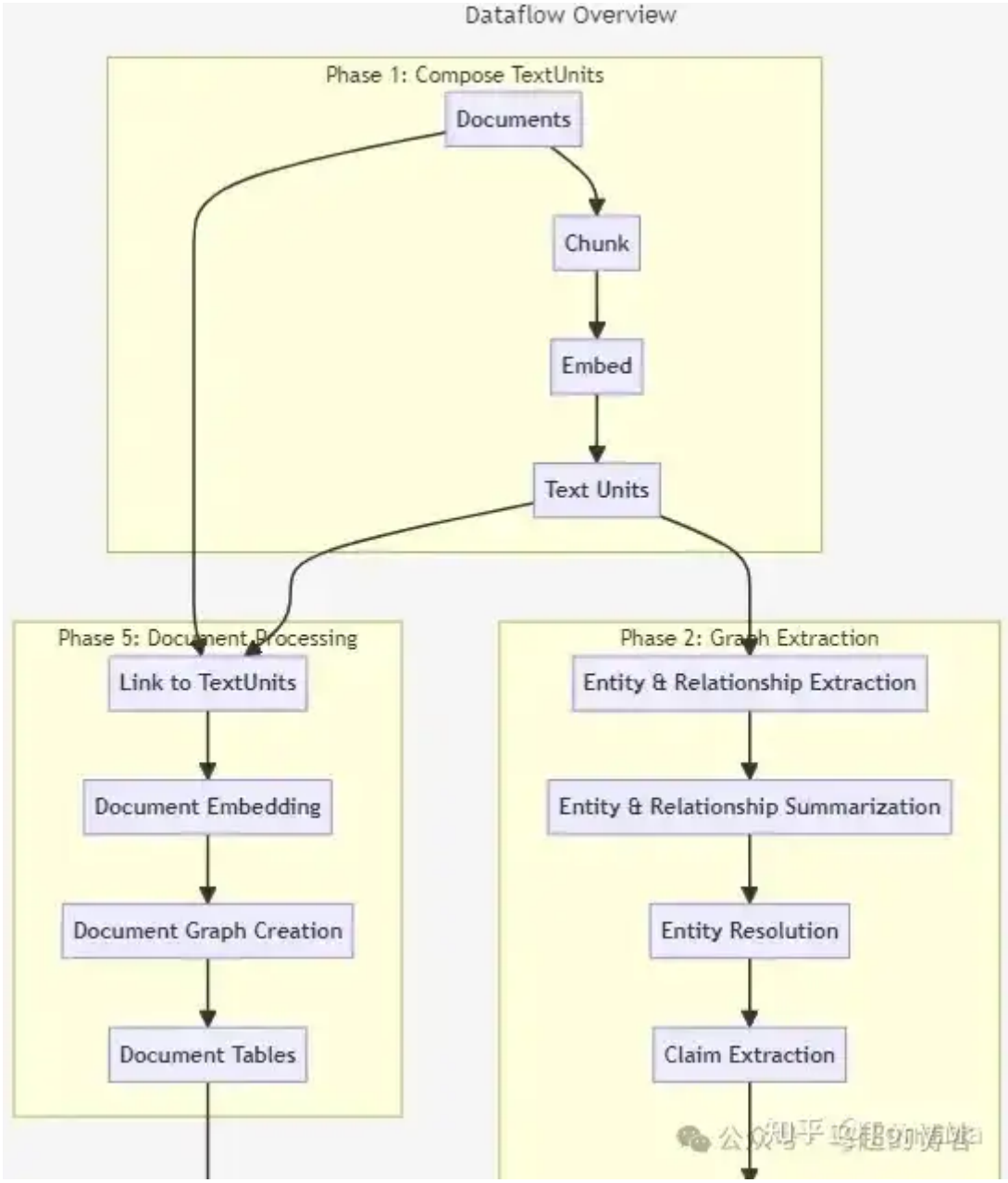
查询

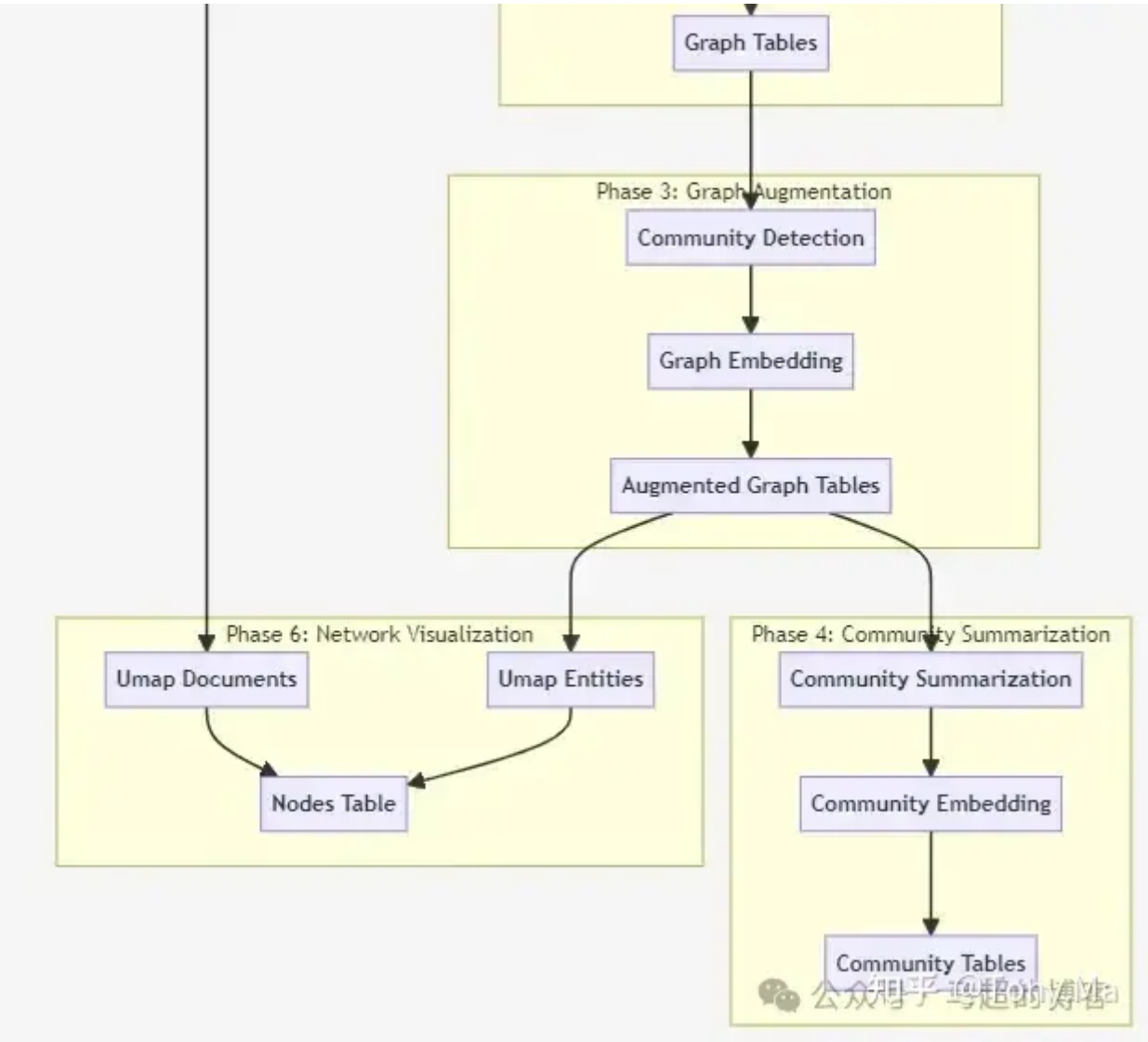
在查询时，这些结构用于在回答问题时为 LLM 上下文窗口提供材料。主要查询模式包括：

- · 利用社区摘要对语料库的整体问题进行全局搜索推理[6]。
- · 本地搜索通过向邻居和相关概念展开来推理特定实体[7]。

索引过程深入解析

本节主要介绍工作流如何将文本文档转换为 GraphRAG 知识模型，主要分为六个阶段[8]。索引工作流的完整配置，请查看配置文档[9]。

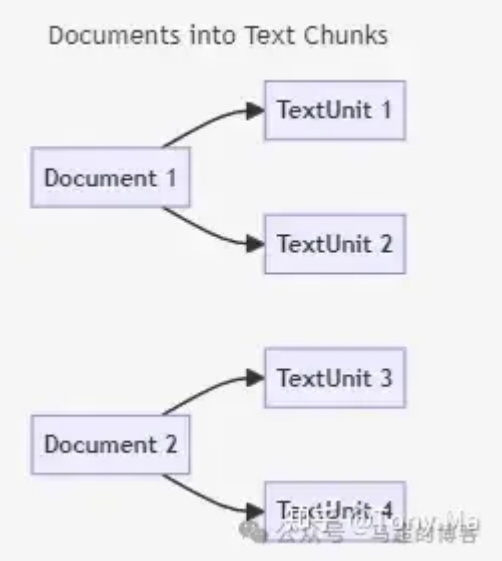

Dataflow Overview
Phase 1: Compose TextUnits
Documents
Chunk
Embed
Text Units
Phase 5: Document Processing
Link to TextUnits
Document Embedding
Document Graph Creation
Document Tables
Phase 2: Graph Extraction
Entity & Relationship Extraction
Entity & Relationship Summarization
Entity Resolution
Claim Extraction



六阶段数据处理工作流预览

步骤 1：处理文本块

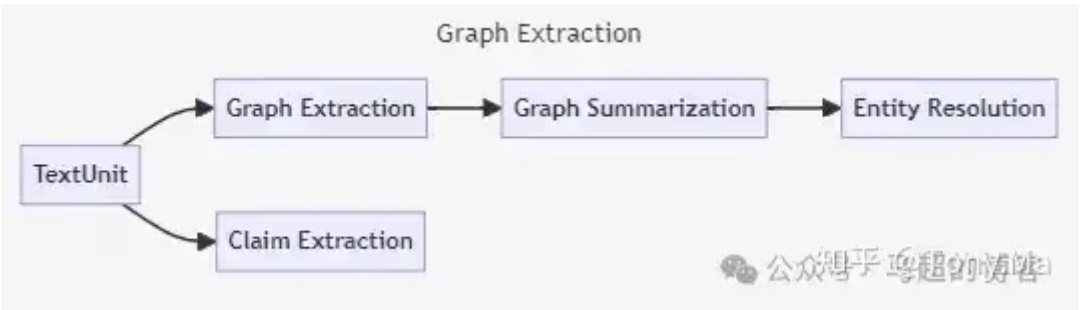
将输入文档转换为 TextUnits。使用文本分段技术将文档进行分块，文本块还作为实体和关系的来源被提取的图引用，便于追溯到原始文本。对于文本块大小的设置需要考虑具体使用情况，一般较大的块会导致输出保真度较低，参考文本意义较小；但是，使用较大的块可以大大缩短处理时间。文档和文本单元之间一般存在严格的一对多关系，在极少数情况下，这可以变成多对多关系（当文档很短并且我们需要其中几个来组成一个有意义的分析单元）。



文本转换为文本块

步骤 2：图提取

在此阶段，分析每个文本单元并提取图元素：实体、关系和协变量。



图元素提取

- · 实体和关系提取

在图提取的第一步中，我们处理每个文本单元，以便使用 LLM 从原始文本中提取实体和关系。此步骤的输出是每个文本单元的子图，其中包含具有名称、类型和描述的实体列表，以及具有源、目标和描述的关系列表。这些子图合并在一起，任何具有相同名称和类型的实体都通过创建其描述数组进行合并。同样，任何具有相同源和目标的关系都通过创建其描述数组进行合并。

- · 实体和关系摘要

现在我们有了实体和关系图，每个图都有一串描述，我们可以将这些列表汇总为每个实体和关系的单个描述。这可以通过要求 LLM 提供简短的摘要来完成，该摘要捕获每个描述中的所有不同信息。这使得我们所有的实体和关系都有一个简洁的描述。

- · 实体消歧（微软 GraphRAG 配置默认是不启用的）

图提取的最后一步是解析任何代表相同现实世界实体但名称不同的实体。由于这是通过 LLM 完成的，并且我们不想丢失信息，因此我们希望采取保守、非破坏性的方法。 然而，我们目前对实体解析的实现是破坏性的。它将为 LLM 提供一系列实体，并要求其确定哪些实体应该合并。然后，这些实体将合并为一个实体，并更新它们的关系。 我们目前正在探索其他实体解析技术。在不久的将来，实体解析

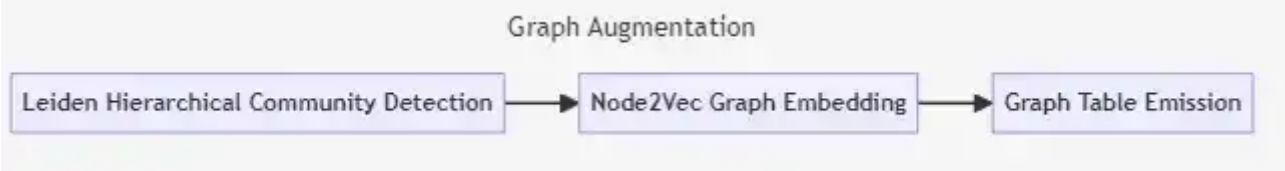
将通过在实体变体之间创建一条边来执行，表明实体已被索引引擎解析。这将允许最终用户撤消索引端解析，并使用类似的过程添加自己的非破坏性解析。

- · 协变量提取

最后，作为一个独立的工作流程，我们从源 TextUnits 中提取声明。这些声明代表具有评估状态和时间限制的积极事实陈述。它们作为称为协变量的主要工件发出（协变量在本地搜索模式中被引用）。

步骤 3：图增强

现在我们有了可用的实体和关系图，我们希望了解它们的社区结构，并用其他信息扩充该图。这分为两个步骤：社区检测和图嵌入。这为我们提供了显式（社区）和隐式（嵌入）方法来理解图的拓扑结构。



图增强

- · 社区检测

在此步骤中，我们使用分层莱顿算法生成实体社区的层次结构。此方法将对我们的图应用递归社区聚类，直到达到社区规模阈值。这将使我们能够了解图的社区结构，并提供一种在不同粒度级别上导航和总结图的方法。

- · 图嵌入

在此步骤中，我们使用 Node2Vec 算法生成图中节点的向量表示。这将使我们能够理解图的隐式结构，并提供额外的向量空间，以便在查询阶段搜索相关概念。

步骤 4：社区总结

社区总结

此时，我们有一个实体和关系的图、实体的社区层次结构以及 node2vec 嵌入。 现在，我们希望基于社区数据并为每个社区生成报告。这让我们可以从多个粒度点对图表有一个高层次的了解。例如，如果社区 A 是顶级社区，我们将获得有关整个图表的报告。如果社区是较低级别的，我们将获得有关本地集群的报告。

- · 生成社区报告

在此步骤中，我们使用 LLM 生成每个社区的摘要。这将使我们能够了解每个社区中包含的独特信息，并从高级或低级角度提供对图表的范围理解。这些报告包含执行概述，并引用社区子结构中的关键实

体、关系和声明。

- · 总结社区报告

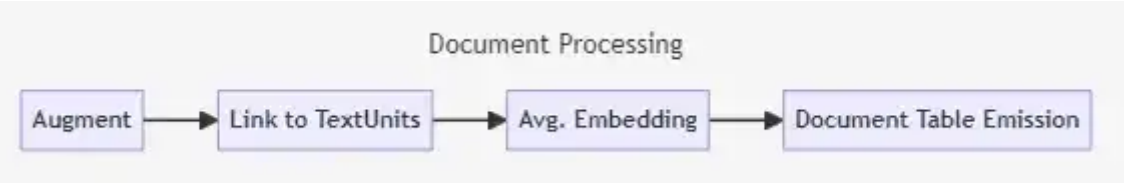
在此步骤中，每个社区报告都会通过 LLM 进行总结，以供摘要使用。

- · 社区嵌入

在此步骤中，我们通过生成社区报告、社区报告摘要和社区报告标题的文本嵌入来生成我们社区的向量表示。

步骤 5：文件处理

在工作流程的这个阶段，开始处理文档图谱核心步骤包括文本块图谱的创建和文档的向量化表示。



文件处理

- · 链接到 TextUnits

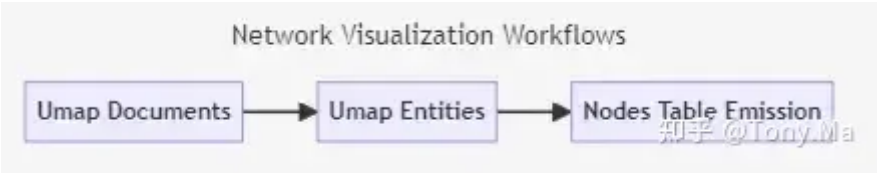
在此步骤中，我们将每个文档链接到第一阶段创建的文本单元，创建文本块之间的上下文关系。这使我们能够了解哪些文档与哪些文本单元相关。

- · 文档嵌入

在此步骤中，我们使用文档切片的平均嵌入来生成文档的向量表示。我们对不重叠的块重新分块文档，然后为每个块生成嵌入。我们创建这些块的平均值，并按标记计数加权，并将其用作文档嵌入。这将使我们能够理解文档之间的隐式关系，并帮助我们生成文档的网络表示。

步骤 6：网络可视化

在工作流程的这个阶段，我们执行了一些步骤来支持现有图中高维向量空间的网络可视化。此时有两个逻辑图表在起作用：实体关系图和文档图。



网络可视化

对于每个逻辑图，我们执行 UMAP 降维以生成图的 2D 表示。这将使我们能够在 2D 空间中可视化图并了解图中节点之间的关系。然后将 UMAP 嵌入作为 Nodes 表发出。该表的行包括一个鉴别器，指示节点是文档还是实体，以及 UMAP 坐标。

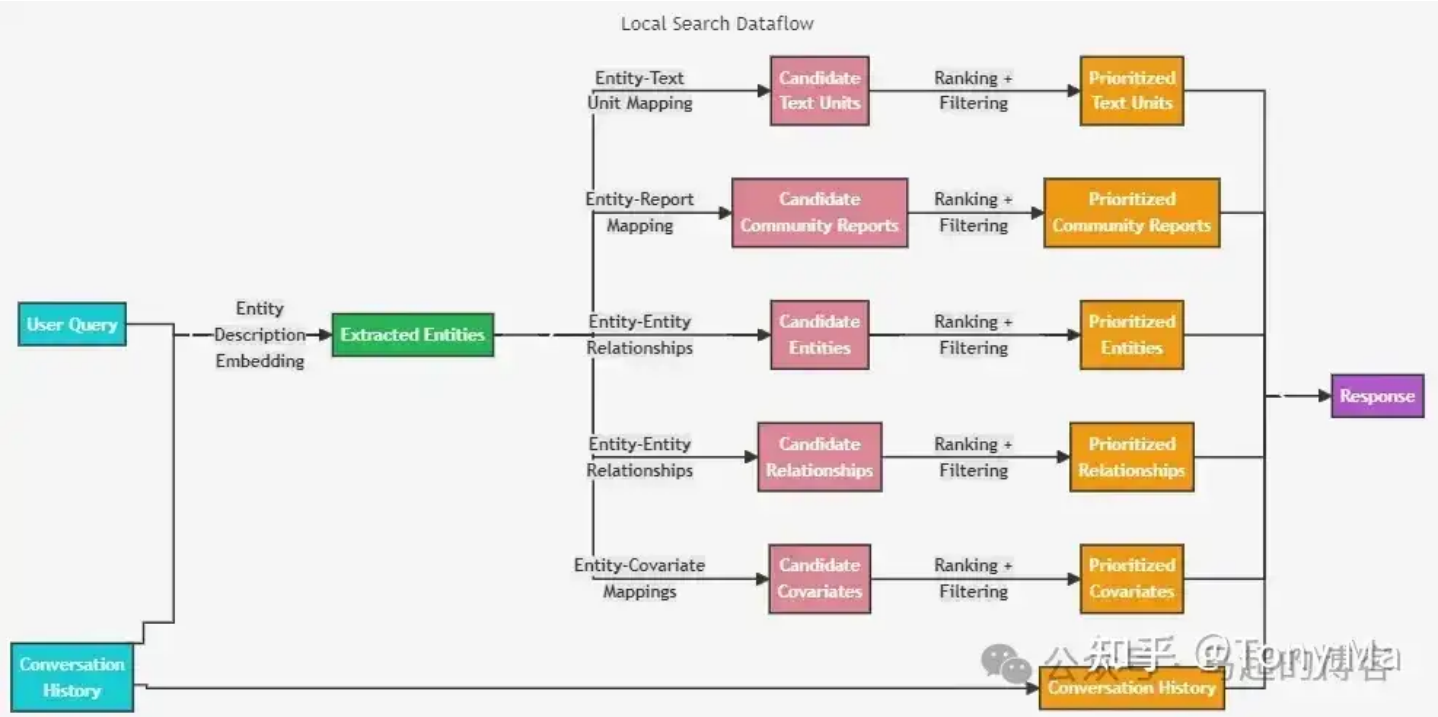
查询过程深入解析

查询引擎是 Graph RAG 库的检索模块。它是 Graph RAG 的核心组件之一。它负责以下任务：

- · 本地搜索
- · 全局搜索
- · 问题生成

本地搜索

- · 基于实体的推理 本地搜索[10]方法通过将 LLM 提取的知识图谱中的相关数据与原始文档的文本块相结合来生成答案。此方法适用于需要了解文档中提到的特定实体的问题（例如，洋甘菊的治疗功效是什么？）。
- · 方法论



本地搜索数据流

给定用户查询和（可选）对话历史记录，本地搜索方法会从知识图谱中识别出一组与用户输入在语义上相关的实体。这些实体可作为知识图谱的访问点，从而提取更多相关详细信息，例如连接实体、关系、实体协变量和社区报告。此外，它还会从与已识别实体相关的原始输入文档中提取相关文本块。然后对这些候选数据源进行优先排序和筛选，以适应预定义大小的单个上下文窗口，该窗口用于生成对用户查询的响应。

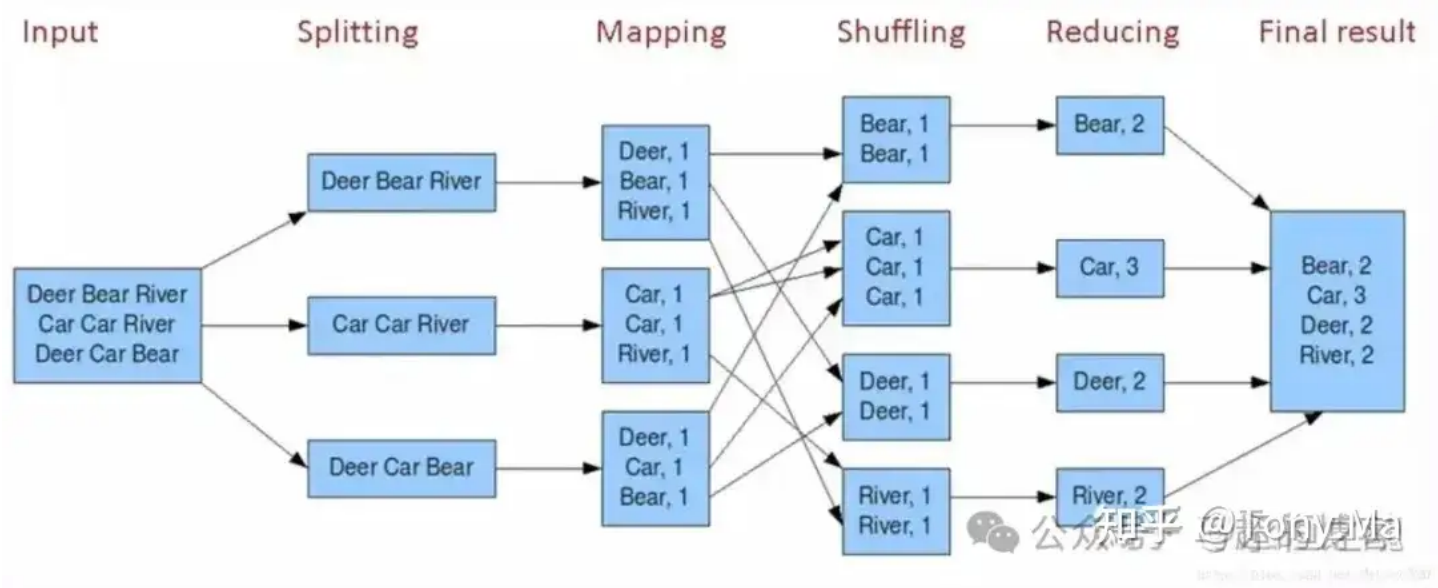
问题生成

- · 基于实体的问题生成 问题生成[11]方法将知识图谱中的结构化数据与输入文档中的非结构化数据相结合，生成与特定实体相关的候选问题。此功能获取用户查询列表并生成下一个候选问题。这对于在对话中生成后续问题或为调查员生成问题列表以深入研究数据集非常有用。

- · 方法论 给定先前用户问题的列表，问题生成方法使用与本地搜索相同的上下文构建方法来提取和优先处理相关的结构化和非结构化数据，包括实体、关系、协变量、社区报告和原始文本块。然后，这些数据记录被放入单个 LLM 提示中，以生成代表数据中最重要或最紧急的信息内容或主题的候选后续问题。

全局搜索

- · 基于整个数据集推理 全局搜索方法通过以 map-reduce 方式搜索所有 AI 生成的社区报告来生成答案。这是一种资源密集型方法，但通常可以很好地回答需要了解整个数据集的问题（例如，本笔记本中提到的药材最重要的价值是什么？）。



MapReduce原理图

上图是一个经典的 MapReduce 原理图结构，包含以下主要步骤（包含全局 GraphRAG 流程的简单注解）。

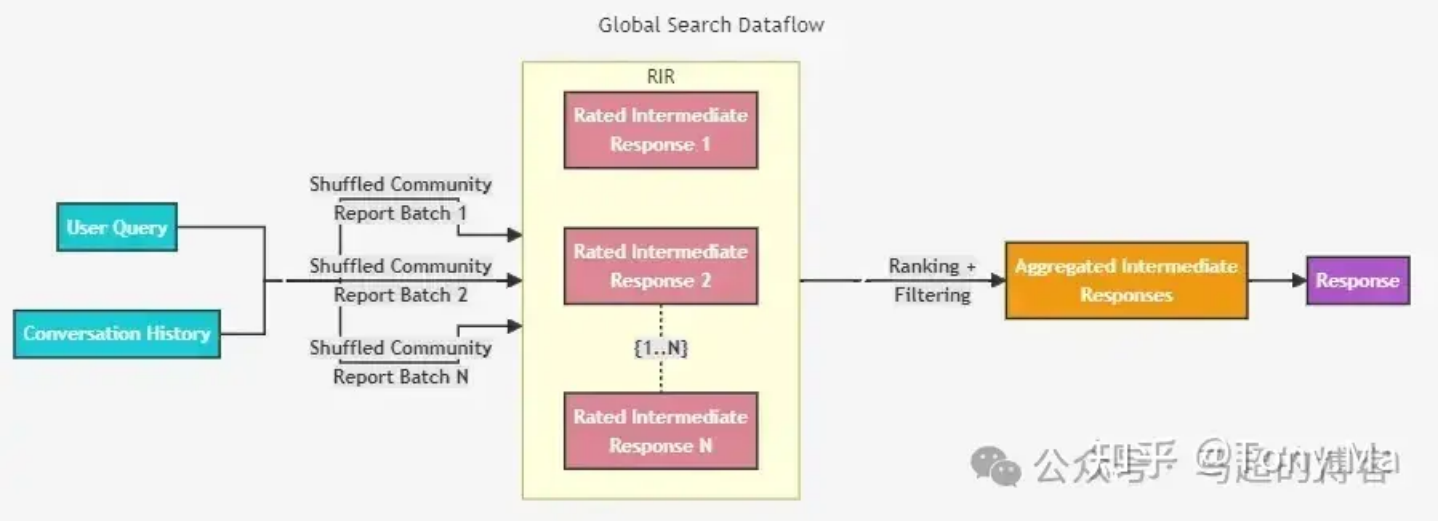
1. 将 input 的文件拆分成 splits，由于测试用的文件较小，所以每个文件作为一个 split，并将文件按行分割。这一步由 mapreduce 框架自动完成。（通过向量检索语义相关的社区）
2. 将分割好的文件交给用户定义的 map 方法进行处理，生成<key,value>对。（生成社区和用户问题相关程度的分值，和社区节点本身组成 MAP）
3. 得到 map 方法输出的<key,value>对后，shuffle 过程，会把相同 key 值相同的放到一起。（社区结果基于与问题的相关度做 Shuffle，可以理解成分组）
4. Reduce 过程，把 key 值相同 value 值累加，得到新的<key,value>对，并作为 word count 的输出结果。（组内综合打分后排序过滤 0 分的结果，然后由 LLM 生成回复）

Baseline RAG 很难处理需要汇总整个数据集的信息才能得出答案的查询。诸如“数据中的前 5 个主题是什么？”之类的查询表现不佳，因为 Baseline RAG 依赖于对数据集内语义相似的文本内容进行向量搜索。查询中没有任何内容可以将其引导至正确的信息。

但是，使用 GraphRAG，我们可以回答这些问题，因为 LLM 生成的知识图谱的结构告诉我们整个数据集的结构（以及主题）。这允许将私有数据集组织成预先汇总的有意义的语义集群。使用我们的全局

搜索[12]方法，LLM 在响应用户查询时使用这些集群来总结这些主题。

- · 方法论



全局搜索数据流

给定用户查询和（可选）对话历史记录，全局搜索方法使用来自图的社区层次结构指定级别的 LLM 生成的社区报告集合作为上下文数据，以 map-reduce 方式生成响应。在此 map 步骤中，社区报告被分割成预定义大小的文本块。然后使用每个文本块生成一个中间响应，其中包含一个要点列表，每个要点都附有数字评级，表明该要点的重要性。在此步骤中，reduce 从中间响应中筛选出的一组最重要的要点被汇总并用作上下文来生成最终响应。

全局搜索响应的质量可能在很大程度上受到选择用于获取社区报告的社区层级的影响。较低层级的报告较为详细，因此往往会产生更全面的响应，但由于报告数量较多，生成最终响应所需的时间和 LLM 资源也可能会增多。

三.综述

1.引言

尽管大语言模型这两年各方面表现非常出色，但是由于缺乏特定领域知识、实时更新信息，导致模型存在一定局限性。这些不足容易引发幻觉现象，即模型生成不准确甚至是虚构的信息。

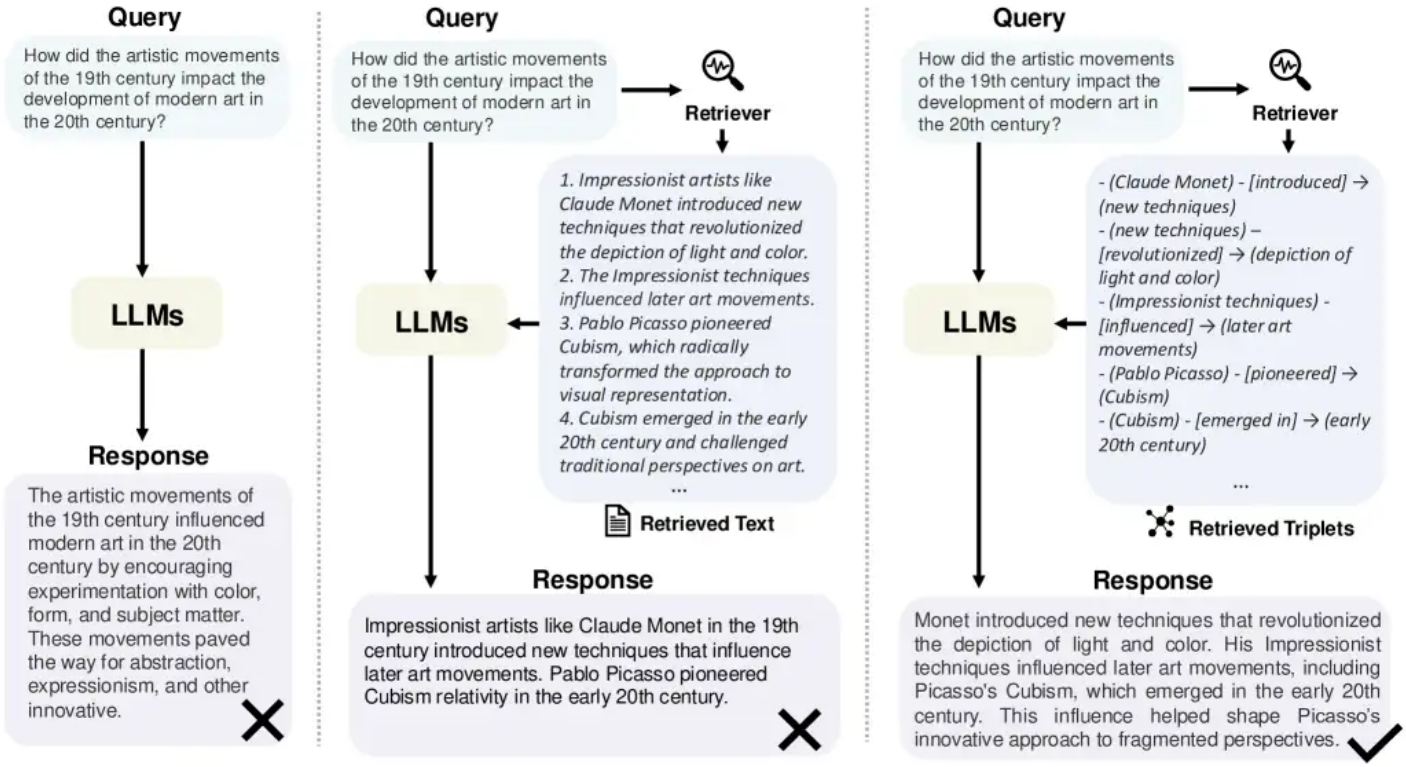
所以，用外部知识来补充大型语言模型以缓解幻觉问题势在必行。检索增强生成（Retrieval-Augmented Generation, RAG）作为一种重要解决方案应运而生。

RAG通过在生成过程中融入检索组件来提升生成内容的质量和相关性。RAG 的核心在于能够动态查询大型文本语料库，将相关的事实知识纳入底层语言模型生成的响应之中。这种融合不但丰富了上下文深度，还保证了事实准确性和特异性。因其出色的表现和广泛的应用，RAG 备受关注，成为大语言模型应用领域的重点。

虽然 RAG 成果斐然，并在各个领域得到广泛应用，但在实际场景中仍面临一些局限：

- · **忽视关系**：实际上，文本内容并非孤立存在，而是相互关联的。传统的 RAG 无法捕获仅靠语义相似性无法呈现的重要结构化关系知识。比如，在通过引用关系连接论文的引用网络中，传统的 RAG 方法侧重于依据查询找到相关论文，却忽略了论文之间重要的引用关系。
- · **冗余信息**：RAG 在连接成提示时，常以文本片段的形式重复内容，致使上下文过长，陷入“Lost in the Middle”的困境。
- · **缺乏全局信息**：RAG 只能检索文档的子集，无法全面掌握全局信息，因而在诸如查询聚焦摘要（Query-Focused Summarization，QFS）等任务中表现不佳。

图检索增强生成（GraphRAG）作为创新的解决方案出现，可以用来解决以上传统RAG的困局。

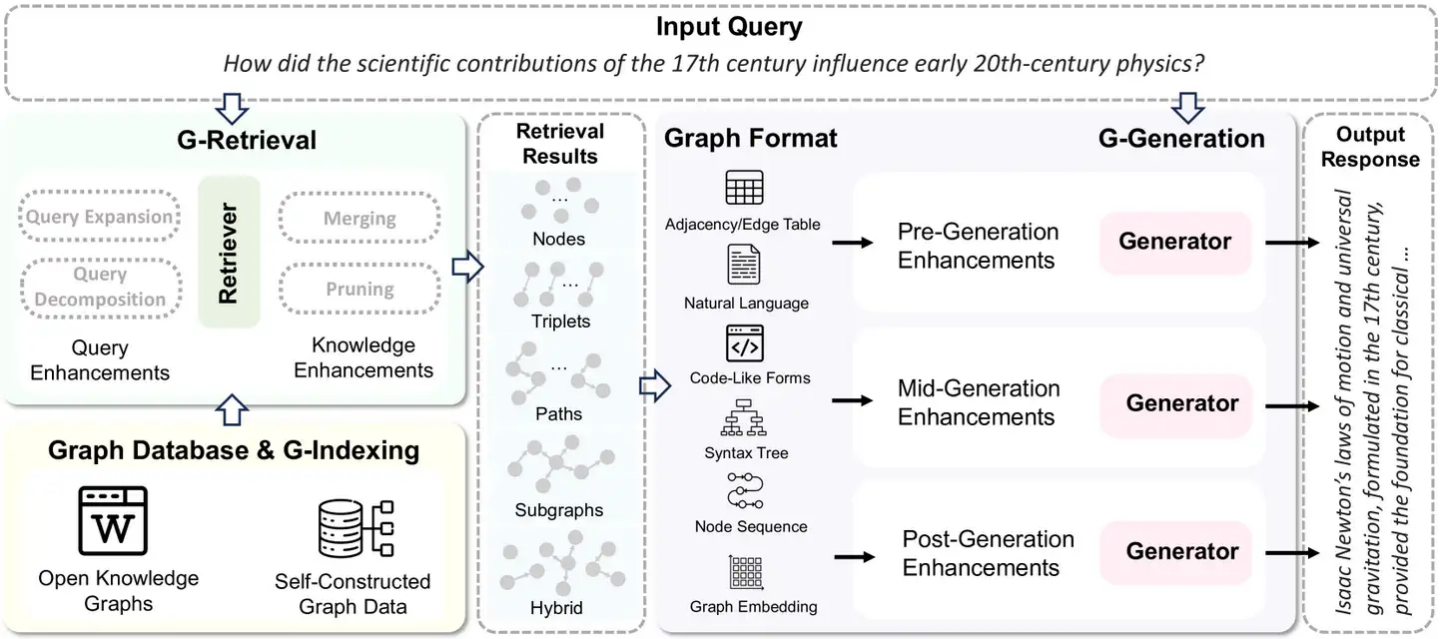


与传统的 RAG 不同，GraphRAG 从预先构建的图数据库中检索与给定查询相关且包含关系知识的图元素，如上图所示。这些元素可能包括节点、三元组、路径或子图。

GraphRAG 考虑了文本之间的联系，能够更准确、更全面地检索关系信息。

此外，图数据，如知识图，对文本数据进行了抽象和总结，大幅缩短了输入文本的长度，减少了冗长的问题。通过检索子图或图社区，能够获取全面的信息，借助捕获图结构中更广泛的上下文和相互联系，有效解决 QFS 挑战。

2.GraphRAG 概述



GraphRAG 是借助外部结构化知识图谱来增进语言模型的上下文理解，并生成更具洞见响应的框架。GraphRAG 的目标在于从数据库中检索出最为相关的知识，进而提升下游任务的答案质量。

鉴于候选子图的数量会随着图的规模呈指数增长，故而需要有效的近似方法。由此，运用图检索器提取最优子图 ，而后生成器依据检索到的子图生成答案。

所以，将 GraphRAG 的整个流程拆解为三个主要阶段：基于图的索引、图引导检索和图增强生成。

2.1 基于图的索引（G-Indexing）

基于图的索引构成了 GraphRAG 的初始阶段，旨在识别或构建图数据库 G，并在其基础上建立索引。

图数据库可以源于公共知识图谱、图形数据，或者依据专有数据源（如文本或其他形式的数据）来构建。索引过程通常涵盖映射节点和边的属性，在相互连接的节点间建立指针，以及组织数据以支持快速遍历和检索操作。索引决定了后续检索阶段的精细程度，在提升查询效率方面发挥着关键作用。

2.2 图引导检索（G-Retrieval）

在基于图的索引之后，图引导检索阶段的重点在于根据用户的查询或输入从图形数据库中提取相关信息。

具体来说，给定以自然语言表述的用户查询，检索阶段旨在从知识图谱中提取最相关的元素（例如，实体、三元组、路径、子图）。

2.3 图增强生成（G-Generation）

图增强生成阶段涉及基于检索到的图数据合成有意义的输出或响应。这可能包括回答用户的查询、生成报告等。在这个阶段，生成器将查询、检索到的图形元素以及可选的提示作为输入来生成响应。

1. 基于图的索引

图数据库的构建与索引乃是 GraphRAG 的基石，其质量直接左右 GraphRAG 的性能表现。

3.1 图数据

GraphRAG 在检索和生成过程中运用了各式各样的图数据。在此，依据其来源将这些数据划分为两类，分别是开放知识图和自建图数据。

3.1.1 开放知识图谱（Open Knowledge Graphs）

开放知识图指的是源自公开可用的存储库或数据库的图数据。运用这些知识图能够大幅缩减开发与维护所需的时间和资源。依据其范畴进一步将它们分为两类，即**通用知识图和领域知识图**。

3.1.1.1 通用知识图谱（General Knowledge Graphs）

通用知识图谱主要存储通用的、结构化的知识，通常仰仗全球社群的集体输入与更新，确保信息库全面且持续更新。

百科知识图属于典型的通用知识图，涵盖了从人类专家和百科全书采集的大规模现实世界知识。例如：

- · Wikidata是一个免费且开放的知识库，存储了维基媒体姐妹项目（如维基百科、维基旅行、维基词典等）的结构化数据。
- · Freebase是一个协作编辑的知识库，从各种来源（包含个人贡献和来自维基百科等数据库的结构化数据）整合数据。
- · DBpedia借助维基百科文章中的信息框和类别来呈现有关数百万实体（包括人物、地点和事物）的信息。
- · YAGO 从维基百科、WordNet 和 GeoNames 收集知识。

常识知识图是另一类通用知识图。它们涵盖抽象的常识知识，例如概念之间的语义关联和事件之间的因果关系。典型的常识知识图包括：

- · ConceptNet 是一个由代表单词或短语的节点通过表示语义关系的边连接而成的语义网络。
- · ATOMIC 对事件之间的因果关系进行建模。

3.1.1.2 领域知识图谱（Domain Knowledge Graphs）

特定领域的知识图谱对于强化 LLMs 解决特定领域问题至关重要。这些 KG 在特定领域提供专业知识，助力模型获取更深入的见解和对复杂专业关系的更全面理解。

在生物医学领域：

- · CMeKG 涵盖疾病、症状、治疗、药物以及医学概念之间的关系。
- · CPubMed-KG 是一个中文的医学知识数据库，构建于 PubMed 丰富的生物医学文献库之上。

在电影领域：

- · Wiki-Movies 从与电影相关的维基百科文章中提取结构化信息，将有关电影、演员、导演、类型和其他相关细节的数据编译成结构化格式。

此外，构建了一个名为 GR-Bench 的数据集，其中包含跨越学术、电子商务、文学、医疗保健和法律领域的五个领域知识图。

3.1.2 自建图数据（Self-Constructed Graph Data）

构建自定义图数据是将特定领域知识融入检索过程的巧妙手段。

对于那些不直接涉及图数据的下游任务，研究者们常提出从多元数据源（如文档、表格、数据库等）构建图谱，并借助GraphRAG技术提升任务表现。

为捕捉文档间的**结构性联系**，有提议构建异构文档图，涵盖共同引用、共同主题、共同发布场所等多种文档级关系，并通过共享关键词建立段落间的联系。

在捕捉文档内**实体间关系**，利用命名实体识别工具提取实体，再通过语言模型深入挖掘实体间关系，构建出知识图谱。

此外，针对特定下游任务，还需设计相应的映射方法。例如，在专利短语相似性推断任务中，将专利数据库转化为专利短语图，通过专利中出现的短语建立专利节点与短语节点间的联系，而专利节点间的联系则基于引用关系。在客户服务技术支持场景中，有提议**将历史问题转化为知识图谱**，通过树状结构保持问题内部联系，并通过语义相似度和阈值维护问题间的联系。

3.2 索引

基于图的索引对于提升图数据库查询操作的效率和速度至关重要，直接影响到后续的检索方法和细节程度。常见的基于图的索引方法包括图索引、文本索引和向量索引。

3.2.1 图索引（Graph Indexing）

图索引作为最常用的方法，保留了图的全部结构，确保任何节点及其边和邻近节点都能轻松访问。在后续检索阶段，可以应用**广度优先搜索**和**最短路径算法**等经典图搜索算法来促进检索任务。

3.2.2 文本索引（Text Indexing）

文本索引通过将图数据转化为文本描述来优化检索过程。这些描述存储于文本语料库中，可以使用各种基于文本的检索技术，如稀疏检索和密集检索。一些方法通过**预设规则或模板，将知识图谱转化为易于理解的文本**。例如，使用预设模板将知识图谱中的三元组转换为自然语言，同时将相同主实体的三元组合并为段落。此外，还有方法将子图信息转化为文本描述，如**通过社区检测在图上生成每个社区的摘要**。

3.2.3 向量索引（Vector Indexing）

向量索引通过将图数据转化为向量表示来提升检索效率，便于快速检索和有效处理查询。例如：

- · 通过查询嵌入可以无缝应用实体链接，同时可以利用局部敏感哈希（LSH）等高效的向量搜索算法。
- · G-Retriever利用语言模型对图中每个节点和边关联的文本信息进行编码。
- · GRAG 使用语言模型将跳数自我网络转换为图嵌入，更好地保留了结构信息。

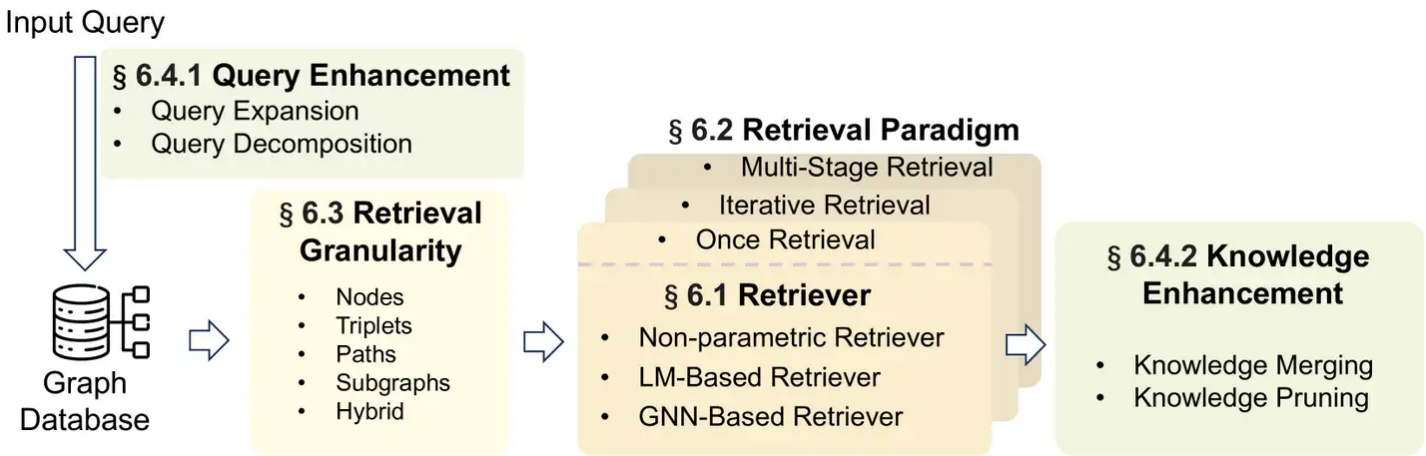
3.2.4 小结

以上三种索引方法各有独特优势：

- · 图索引便于轻松获取结构信息
- · 文本索引简化了文本内容的检索
- · 向量索引能实现快速高效的搜索。

在实际应用中，往往更青睐结合这些索引方法的混合方式，而非单纯依赖其中一种。

2. 图引导检索（Graph-Guided Retrieval）



在 GraphRAG 中，检索过程对于从外部图数据库提取相关且高质量的图数据，以保障生成输出的质量与相关性而言，至关重要。

然而，检索图数据面临两大显著挑战：

-（1）**候选子图激增（Explosive Candidate Subgraphs）**：随着图规模的扩大，候选子图的数量呈指数式增长，这就需要启发式搜索算法来高效地探寻和检索相关子图。

-（2）**相似度测量欠缺（Insufficient Similarity Measurement）**：要精准测量文本查询与图数据之间的相似度，就需要开发能够理解文本和结构信息的算法。

4.1 检索器（Retriever）

在 GraphRAG 中，各类检索器在应对检索任务的不同方面均具有独特优势。依据其基础模型将检索器分为三类：非参数检索器、基于语言模型（LM）的检索器和基于图神经网络（GNN）的检索器。

预处理步骤中所使用的模型，如查询编码和实体链接等，此处不予考虑，因为这些模型在不同方法中有所差异，并非重点。

4.1.1 非参数检索器（Non-parametric Retriever）

非参数检索器基于启发式规则或传统的图搜索算法，不依赖深度学习模型，从而实现了高检索效率。

4.1.2 基于 LM 的检索器（LM-based Retriever）

由于具备强大的自然语言理解能力，语言模型在 GraphRAG 中是高效检索器。

这些模型在处理和解读各类自然语言查询方面表现出色，使其在基于图的框架内适用于广泛的检索任务。

将语言模型分为两类：判别式和生成式语言模型。

- · Subgraph Retriever 训练 RoBERTa 作为检索器，其从实体展开，并在顺序决策过程中检索相关路径。
- · KG-GPT 采用大型语言模型（LLM）生成特定实体的前相关关系集。 利用微调后的 GPT-2 生成推理路径。
- · StructGPT 借助 LLM 自动调用若干预定义函数，借此检索并整合相关信息以辅助进一步推理。

4.1.3 基于 GNN 的检索器（GNN-based Retriever）

GNN 善于理解和利用复杂的图结构。基于 GNN 的检索器通常对图数据进行编码，而后依据与查询的相似度对不同的检索粒度进行评分。例如：

- · GNN-RAG 首先对图编码，为每个实体赋予一个分数，并基于阈值检索与查询相关的实体。
- · EtD 多次迭代以检索相关路径。在每次迭代中，它首先使用 LLaMA2 选取连接当前节点的边，然后运用 GNN 获取新层节点的嵌入，以供下一轮 LLM 选择。

4.1.4 小结

在检索过程中，非参数检索器检索效率不错，但因缺乏下游任务的训练，可能存在检索不准确的问题。

同时，尽管基于语言模型的检索器和基于图神经网络的检索器检索准确率更高，却需要大量计算开销。

鉴于此互补性，许多方法提出了**混合检索**方式，以提升检索效率与准确性。众多方法采用**多阶段检索策略**，每个阶段运用不同模型。比如：

- · RoG 先利用大型语言模型生成规划路径，再从知识图谱中提取符合规划路径的路径。
- · GenTKGQA 借助大型语言模型从查询中推断关键关系和约束，并依据这些约束提取三元组。

4.2. 检索范式

在 GraphRAG 中，不同的检索范式，包括单次检索、迭代检索和多阶段检索，对于提高检索信息的相关性与深度发挥着关键作用。

- · 单次检索旨在通过一次操作获取所有相关信息
- · 迭代检索基于先前的检索结果进行进一步搜索，逐步聚焦到最相关的结果。将迭代检索分为自适应检索和非自适应检索，区别在于检索的停止是否由模型决定。
- · 多阶段检索是把检索划分为多个阶段。每个阶段可能会运用不同类型的检索器，以获取更精准和多样的搜索结果。

4.2.1 单次检索（Once Retrieval）

单次检索旨在通过一次查询获取所有相关信息。一类方法利用嵌入相似度来检索最相关的信息片段。另一类方法设计预定义的规则或模式，直接从图数据库中提取特定的结构化信息，如三元组、路径或子图。例如：

- · G-Retriever 利用扩展的 PCST 算法来检索最相关的子图。
- · KagNet 提取所有主题实体对之间长度不超过k的路径，并提取包含所有主题实体及其2跳邻居的子图。

4.2.2 迭代检索（Iterative Retrieval）

在迭代检索中，采用多个检索步骤，后续的搜索取决于先前检索的成果。这些方法旨在通过连续迭代深化对检索信息的理解或完整性。进一步将迭代检索分为两类：（1）非自适应和（2）自适应检索。

4.2.2.1 非自适应检索（Non-Adaptive Retrieval）

非自适应方法通常遵循固定的检索顺序，检索的终止由设定最大时间或阈值来决定。例如，PullNet 通过T次迭代检索与问题相关的子图。

4.2.2.2 自适应检索（Adaptive Retrieval）

自适应检索的一大特色在于让模型自主决定完成检索活动的最佳时机。

例如，借助语言模型进行跳步预测，以此作为结束检索的指标。还有一批研究人员利用模型生成的特殊标记或文本作为检索过程的终止信号。

比如：ToG 提示大型语言模型代理探索多种可能的推理路径，直至大型语言模型根据当前推理路径判定问题能够得到解答。训练一个 RoBERTa 从每个主题实体拓展路径。在此过程中，引入一个名为“[END]”的虚拟关系来终止检索流程。

另一种常见方式是将大型模型视作代理，使其能够直接生成问题的答案，以此标志迭代结束。例如，提出基于大型语言模型的代理在图上进行推理。这些代理能够自主确定检索的信息，调用预先设定的检索工具，并依据检索到的信息停止检索过程。

4.2.3 多阶段检索（Multi-Stage Retrieval）

多阶段检索将检索过程线性地划分为多个阶段，在这些阶段之间还存在诸如检索增强甚至生成过程等额外步骤。在多阶段检索中，不同阶段可能采用各类检索器，这使得系统能够融合针对查询不同方面定制的各种检索技术。

4.2.4 小结

在 GraphRAG 中，一次性检索通常复杂度较低且响应时间较短，适用于需要实时响应的场景。相较而言，迭代检索往往涉及更高的时间复杂度，尤其是使用大型语言模型作为检索器时，可能导致处理时间较长。

检索范式的选择应依据具体的用例和需求来平衡精度与时间复杂度。

4.3 检索粒度（Retrieval Granularity）

依照不同的任务场景和索引类型，研究人员设计了不同的检索粒度，可分为节点、三元组、路径和子图。

每种检索粒度都有自身的优势，适用于不同的实际场景。

4.3.1 节点（Nodes）

节点能够实现对图中单个元素的精准检索，对于有针对性的查询和特定信息提取十分理想。通常来讲，对于知识图谱，节点指的是实体。对于其他类型的文本属性图，节点可能包含描述节点属性的文本信息。通过检索图中的节点，GraphRAG 系统能够提供关于其属性、关系和上下文信息的详尽见解。例如，以及构建文档图并检索相关的段落节点。以及从构建的知识图中检索实体。

4.3.2 三元组（Triplets）

三元组由实体及其关系以主语-谓语-宾语元组的形式构成，为图中的关系数据提供了一种结构化的呈现方式。三元组的这种结构化形式有利于实现清晰且有序的数据检索，在理解实体间关系及上下文相关性至关重要的场景中具有优势。

检索包含主题实体的三元组以获取相关信息，并首先运用预定义模板将图数据中的每个三元组转化为文本语句，接着采用文本检索器来提取相关三元组。

直接从图数据中检索三元组可能仍存在上下文广度和深度不足的问题，以致无法捕捉间接关系或推理链。建议基于原始问题生成逻辑链，再检索每个逻辑链的相关三元组。

4.3.3 路径（Paths）

对路径粒度数据的检索可被视为捕获实体之间的关系序列，从而增强了上下文理解和推理能力。

在 GraphRAG 中，检索路径因能够捕捉图中的复杂关系和上下文依赖关系而具有显著优势。

随着图规模的增大，可能的路径呈指数级增长，这使得路径检索面临挑战，计算复杂度也随之提升。

为解决此问题，一些依据预定义规则的方法来检索相关路径。例如:

- · 先在查询中选定实体对，然后遍历查找 n-hop 范围内它们之间的所有路径。
- · HyKGE 首先定义了三种类型的路径：路径、共同祖先链和共同出现链，接着利用相应规则来检索这三类路径。

还有一些借助模型在图上进行路径搜索的方法。

- · ToG 提议提示 LLM 代理在 KGs 上执行波束搜索，以找到多个有助于回答问题的可能推理路径。并且首先利用模型生成可靠的推理计划，再依据这些计划检索相关路径。
- · GNN-RAG 首先识别问题中的实体。随后，提取满足一定长度关系的实体间的所有路径。

4.3.4 子图（Subgraphs）

检索子图具有显著优势，因其能够捕捉图内的综合关系上下文。这种粒度使 GraphRAG 能够提取和分析嵌入在较大结构中的复杂模式、序列和依赖关系，有助于更深入的洞察和对语义连接更细腻的理解。

为保证信息的完整性和检索效率，一些方法提出了一种基于初始规则的方法来检索候选子图，随后对其进行优化或进一步处理。从自行构建的专利短语图中检索专利短语的自我图。并且首先选取主题实体及其两跳邻居作为节点集，然后选择头和尾实体均在节点集中的边来构成子图。此外，还有一些基于嵌入的子图检索方法。

除了上述直接的子图检索方式，有些工作建议先检索相关路径，而后据此构建相关子图。比如，训练一个RoBERTa模型，通过顺序决策流程来识别多个推理路径，接着合并来自不同路径的相同实体，从而得出最终的子图。

4.3.5 混合粒度（Hybrid Granularties）

鉴于上述各类检索粒度的优劣，部分研究人员提议采用混合粒度，也就是从图数据中检索多种粒度的相关信息。这种粒度类型增强了系统捕获详细关系和更广泛的上下文理解的能力，进而减少噪声并提高检索数据的相关性。以往的诸多工作提议借助LLM代理来检索复杂的混合信息，运用基于LLM的代理自适应地选取节点、三元组、路径和子图。

4.3.6 小结

-（1）在实际应用中，检索粒度之间不存在清晰的界限，因为子图能够由多条路径构成，而路径又可以由若干三元组形成。

-（2）诸如节点、三元组、路径和子图之类的各种粒度在GraphRAG过程中具备不同的优势。

在选择粒度时，依据任务的具体情境，在检索内容和效率之间做好平衡至关重要。对于简单的查询或者当效率最为关键时，更精细的粒度如实体或三元组或许更受青睐，以优化检索速度和相关性。

相较而言，复杂场景往往得益于融合多种粒度的混合方法。这种方法能够确保更全面地理解图结构和关系，提升生成响应的深度和准确性。

因此，GraphRAG在粒度选择上的灵活性使其能够有效适应各个领域的各种信息检索需求。

4.4 检索增强（Retrieval Enhancement）

为保证高检索质量，研究人员提出了增强用户查询和检索所得知识的技术。把查询增强划分为查询扩展和查询分解，把知识增强分为合并与修剪。

这些策略共同优化了检索过程。虽然像查询重写之类的其他技术在RAG中常用，但在GraphRAG中应用较少。所以作者不对这些方法进行深入探讨，尽管它们有可能适用于GraphRAG。

4.4.1. 查询增强（Query Enhancement）

用于丰富信息以获取更优的检索效果。这可能涵盖查询扩展和查询分解。

4.4.1.1 查询扩展（Query Expansion）

由于查询通常较短且信息含量有限，查询扩展通过添加额外的相关术语或概念来补充或优化原始查询，从而改善搜索结果。 利用LLM基于KG生成有依据的关系路径来增强检索查询。

- · 采用SPARQL从Wikidata获取查询实体的所有别名来扩充检索查询，这些别名捕捉了同一实体的词汇变化。
- · 共识视图知识检索方法来提高检索准确性，该方法先发现语义相关的查询，然后重新为原始查询术语加权以增强检索性能。
- · HyKGE 利用一个大型模型生成问题的假设输出，将假设输出与查询相连作为检索器的输入。

4.4.1.2 查询分解（Query Decomposition）

查询分解技术把原始用户查询拆解或分解为更小、更具体的子查询。每个子查询通常聚焦于原始查询的特定方面或组成部分，有效地降低了语言查询的复杂性和模糊性。例如，把主要问题拆分为子句，每个子句代表一种不同的关系，并依次为每个子句检索相关的三元组。

4.4.2. 知识增强（Knowledge Enhancement）

在检索出初始结果后，运用知识增强策略来优化和改进检索器的结果。此阶段通常涉及知识合并和知识修剪流程，以凸显最相关的信息。

4.4.2.1 知识合并（Knowledge Merging）

知识合并所检索到的信息能够压缩和聚合信息，通过整合来自多个来源的相关细节，有助于获取更全面的视角。

这种方法不仅增强了信息的完整性和连贯性，还**缓解了模型中输入长度限制**的问题。

KnowledgeNavigator 通过三元组聚合来合并节点并压缩检索到的子图，以提升推理效率。在子图检索中，从每个主题实体检索到顶部路径以形成单个子图后，合并来自不同子图的相同实体以形成最终子图。并且基于关系合并检索到的子图，将满足相同关系的头实体和尾实体组合成两个不同的实体集，最终形成关系路径。

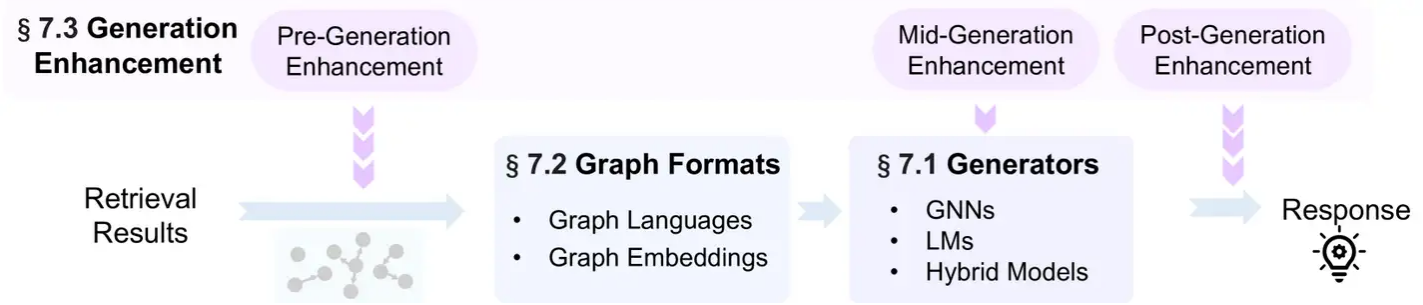
4.4.2.2 知识修剪（Knowledge Pruning）

知识修剪是指过滤掉不太相关或冗余的检索信息以细化结果。以往的修剪方法包含两个主要类别：基于重排序的方法和基于大型语言模型（LLM）的方法。

- · 重排序方法使用定制的指标或标准对检索到的信息进行重新排序或确定优先级。
  - · 一类方法引入更强大的模型进行重新排名。例如，将每个检索到的三元组与问题-选项对相连接，并采用预训练的交叉编码器对检索到的三元组重新排名。利用 FlagEmbedding 对文本进行编码，对嵌入模型“bge_reranker_large”返回的 top-k 文档重新排名。
  - · 另一类利用查询和检索到的信息之间的相似性进行排名。
  - · 第三类方法提出了用于重新排名的新指标。例如，提出了一种同时衡量检索到的文本块的影响力和新鲜度的指标。KagNet 将检索到的路径分解为三元组，并依据知识图谱嵌入（KGE）技术测量的置信度得分对路径重新排名。

基于 LLM 的方法在捕捉复杂的语言模式和语义细微差别方面表现出众，从而提高了其更准确地对搜索结果进行排名或生成响应的能力。为避免引入噪声信息，和提议通过调用 LLM 来修剪不相关的图数据。

3. 图增强生成



在 GraphRAG 中，生成阶段将检索到的图数据与查询相融合，以提升回答质量，需依据下游任务来选定适宜的生成模型。

把检索到的图数据转化为与生成器适配的格式，生成器会将查询和转换后的图数据当作输入，从而生成最终响应。

5.1. 生成器

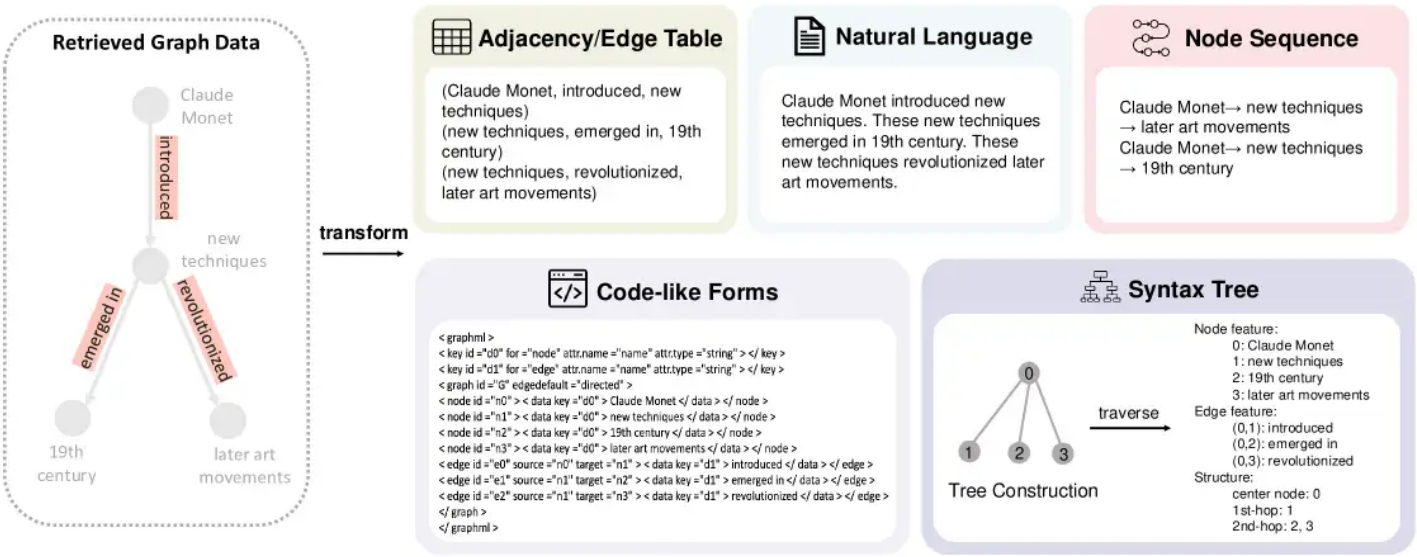
生成器的选取通常取决于当下的下游任务类型。

- · 对于判别类任务（例如，多项选择题回答）或者能够被表述为判别类任务的生成任务（例如，KBQA），能够运用 GNNs 或者判别语言模型来学习数据的表征。
- · 对于生成任务而言，仅使用 GNNs 和判别语言模型是不够的。这些任务需要生成文本，这就需要部署解码器。
- · 混合模型：鉴于图神经网络（GNNs）在展现图数据结构方面的长处，以及语言模型（LMs）对文本的强大理解能力，众多研究正在探索将这两种技术相融合，以生成连贯的回应。本文把混合生成的方式划分为两类：级联范式与平行范式。

5.2. 图格式

当把 GNNs 用作生成器时，图数据能够直接编码。但当使用 LMs 作为生成器时，图数据的非欧几里得特性构成了难题，因为它无法直接与文本数据相结合输入到 LMs 中。为解决此问题，采用图转换器将图数据转换为与 LMs 兼容的格式。这种转换通过让 LMs 能够有效地处理和利用结构化的图信息，从而增强了其生成能力。

作者总结了两种不同的图格式：图语言和图嵌入。



5.2.1. 图语言

图描述语言乃是一种专为表征和呈现图数据而构建的规范化符号系统。它设定了统一的语法和语义框架，用以描绘图中的组件及其相互连接。借由这些语言，用户能够以机器可理解的格式统一地生成、操纵和解读图数据。它们能够实现图架构的定义、节点与边的属性规范，以及在图结构上进行操作和查询。

5.2.1.1 邻接/边表

邻接表和边表乃是描述图结构的常用手段。邻接表枚举每个顶点的紧邻，为稀疏图中的连接提供了紧凑的表述方式。

5.2.1.2 自然语言

鉴于用户的查询通常以自然语言呈现，再考虑到语言模型（LMs）卓越的自然语言理解能力，运用自然语言描述检索到的图数据便成为一种颇具吸引力的方法。通过将图数据转化为描述性、易于理解的语言，LMs 能够弥合原始数据表示与用户友好型信息之间的鸿沟，促进与数据驱动应用程序更有效的交互。

5.2.1.3 类代码形式

考虑到自然语言描述和其他 1-D 序列本质上难以直接呈现图数据的 2-D 结构，同时鉴于 LMs 强大的代码理解能力，尝试采用类代码格式来表示图结构。例如，研究使用图形建模语言（GML）和图形标记语言（GraphML）来表示图形。这些标准化语言专为图数据而设计，提供了涵盖节点、边及其相互关系的全面描述。

5.2.1.4 语法树

相较于直接将图扁平化，将图转变为类似于语法树的结构。语法树具有层次结构，作为拓扑图，也保持着拓扑顺序。此方法留存了更多的结构信息，增进了对图内在属性的理解和分析。这种转变不但保留了不同图元素之间的关系动态，还推动了更复杂的图分析和处理算法。

5.2.1.5 节点序列

通过节点序列来呈现图，这些序列通常依据既定规则生成。相较于自然语言描述，这些序列更为精炼，且融入了先验知识，尤其是规则所强调的结构信息。将检索到的路径转化为节点序列，并输入到大型语言模型（LLM）中以提升任务表现。

LLaGA提出了两种模板，用以将图转化为节点序列。

- · 第一种模板，即邻域细节模板，对中心节点及其直接邻近区域进行了详尽的审视。
- · 第二种模板，称为跳数场概览模板，提供了节点邻域的概括视角，并可扩展以覆盖更广区域。GNN-RAG以节点序列的形式，将检索到的推理路径作为提示输入到语言模型中。

5.2.2. 图嵌入

图语言方法将图数据转化为文本序列，可能导致上下文过长，增加计算成本，并可能超出LLM的处理能力。此外，即便借助图语言，LLM在完全理解图结构方面仍存在挑战。

因此，利用图神经网络（GNN）将图以嵌入形式表示，成为一个充满希望的替代方案。核心难题在于如何将图嵌入与文本表示融合到统一的语义空间中。目前的研究集中在使用前文提到的提示工程方法。

将图表示输入到语言模型主要通过开源模型实现，而非像GPT-4这样的闭源模型。尽管图嵌入方法避免了处理长文本输入的问题，但它们也面临其他挑战，如难以精确保留特定实体名称等信息和泛化能力不足。

5.3. 生成增强

在生成阶段，除了将检索到的图数据转换为生成器可接受的格式，并与查询一同输入以生成最终响应外，许多研究者还探索了多种生成增强技术来提升输出响应的质量。这些技术可根据其应用时机分为三类：生成前增强、生成中增强和生成后增强。

5.3.1. 生成前增强

生成前增强技术着眼于在将数据或表示输入生成器前，提升输入数据或表示的质量。实际上，生成前增强与检索之间并无明显界限。将检索阶段定义为从原始图中检索知识，并进行合并与修剪的过程。随后的操作则视为生成前增强。

常用的预生成增强手段主要在于对检索所得的图数据进行语义充实，以达成图数据与文本查询更紧密的融合。利用大型语言模型重写检索到的图数据，增进转换后的自然语言输出的自然度与语义丰富性。此方法不但保证图数据转化为更流畅、自然的语言，还丰富了其语义内涵。

5.3.2. 生成中增强

生成中增强通常依据中间结果或上下文线索调整生成策略。TIARA 引入了约束解码以控制输出空间并减少生成错误。在生成逻辑形式时，如果约束解码器检测到当前正在生成模式项，它会将下一个生成的标记限定在包含知识库（KB）类别和关系的尝试中的选项。与波束搜索相比，这种方式确保生成的模式项必定存在于知识图中，从而降低生成错误。还有其他方法调整大型语言模型的提示以实现多步推理。例如，MindMap 不仅生成答案，还生成推理过程。

5.3.3. 生成后增强

生成后增强在初始响应生成之后进行，主要包括整合多个生成的响应以获取最终响应。有些方法专注于整合在不同条件或输入下同一生成器的输出。

例如，为每个图社区生成摘要，接着依据摘要生成对查询的响应，然后使用大型语言模型对这些响应进行评分。最终，响应依据其分数降序排列，并依次纳入提示，直至达到令牌限制。随后，大型语言模型生成最终响应。并且首先将查询分解为若干子问题，然后为每个子问题生成答案，最后合并所有子问题的答案以得到最终答案。

4. 训练

6.1. 检索器的训练策略

6.1.1. 无训练

当前，主要有两种无训练检索器在使用：

- · 第一种是由非参数检索器构成。这些检索器依赖于预先定义的规则或传统的图搜索算法，而非特定的模型。
- · 第二种利用预训练的语言模型作为检索器：
  - · 利用预训练的嵌入模型对查询进行编码，并依据查询和图元素之间的相似度直接进行检索。
  - · 采用生成式语言模型进行无训练检索。候选图元素，比如实体、三元组、路径或子图，会作为提示输入的一部分提供给大型语言模型。随后，大型语言模型借助语义关联，根据所提供的提示来选择合适的图元素。

6.1.2. 基于训练

训练检索器通常采用**自回归**的方式，即把先前的关系路径连接到查询的末尾。接着，模型依据这一连接后的输入来预测下一个关系。

然而，大多数数据集中检索内容的真实值缺失构成了一个重大难题。为解决此问题，许多方法尝试基于远程监督构建推理路径，以引导检索器的训练。还有另一类方法利用隐性的中间监督信号来训练检索器。

6.2. 生成器的训练

6.2.1. 无训练

无训练生成器主要适用于闭源大型语言模型或需要规避高训练成本的情形。检索到的图数据与查询一同输入到大型语言模型中。大型语言模型随后依据提示中给出的任务描述生成响应，极大地依赖其理解查询和图数据的固有能力。

6.2.2. 基于训练

对于生成式大型语言模型，可运用监督微调（SFT）进行微调，其中输入任务描述、查询和图数据，并将输出与下游任务的真实情况进行对比。

对于充当生成器的 GNN 或判别模型，采用针对下游任务的专门损失函数来有效训练模型。

6.3. 联合训练

同时**联合训练检索器和生成器**，凭借它们的互补优势来提升下游任务的性能。有些方法**将检索器和生成器统一到一个模型里，通常是大型语言模型**，并同时以检索和生成目标对它们进行训练。这种方法借助了统一架构的聚合能力，使模型能够在一个框架内无缝检索相关信息并生成连贯的响应。

其他方法包括先分别训练检索器和生成器，而后采用联合训练技术对两个组件进行微调。例如，子图检索器采用交替训练范式，其中检索器的参数被固定，利用图数据训练生成器。随后，生成器的参数被固定，并且使用来自生成器的反馈来引导检索器的训练。这种迭代流程有助于两个组件以协调的方式优化其性能。

5. 应用与评估

7.1. 下游任务

GraphRAG 应用于多种下游任务（尤其是 NLP 任务），涵盖问答、信息抽取等。

- · 问答任务：包含知识库问答（KBQA）和常识问答（CSQA）。 -（1）KBQA（Knowledge Base Question Answering）：GraphRAG 的基础下游任务。在 KBQA 中，问题通常与特定知识图谱相关，答案常涉及知识图谱中的实体、关系或实体集合间的操作。该任务考验系统检索和对结构化知识库进行推理的能力，这对促进复杂查询响应至关重要。 -（2）CSQA（CommonSense Question Answering）：有别于 KBQA，CSQA 主要以选择题的形式呈现。常识推理通常会给出一个常识问题以及数个答案选项，每个选项可能代表一个实体的名称或一个陈述。目的是让机器利用外部常识知识图谱，如 ConceptNet，来找到与问题和选项相关的知识，并进行恰当的推理从而得出正确答案。
- · 信息检索：分为两类：实体链接（EL）和关系抽取（RE）。 -（1）实体链接：实体链接（EL）是自然语言处理领域的关键任务，涉及识别文本段中提到的实体，并将其与知识图谱中的相应实体相链接。借助诸如 Graph RAG 这样的系统，能够从知识图谱中检索相关信息，这有助于准确推断与文本中提及相匹配的特定实体。
    - · （2）关系抽取：关系抽取（RE）旨在识别和归类文本中实体之间的语义关系。GraphRAG 能够通过使用基于图的结构对实体间的相互依赖关系进行编码和利用，从而显著增强此任务，有助于从各种文本源更精准和结合上下文细致地抽取关系数据。

- · 其他：GraphRAG 还能应用于自然语言处理领域的其他各类任务，如事实核查、链接预测、对话系统和推荐系统。 -（1）事实核查：通常需要借助知识图谱来评估某一事实陈述的真实性。模型的职责在于通过利用结构化的知识储备库来判定给定事实断言的有效性。GraphRAG 技术能够用于提取实体间的证据关联，以此提升系统的效率与准确性。 -（2）链接预测：对图中实体之间缺失的关系或者潜在连接进行预测。GraphRAG 凭借其从图中检索和分析结构化信息的能力被应用于该项任务，通过揭示图数据中的潜在关系和模式来提高预测的精准度。 -（3）对话系统：运用自然语言与人类进行交流，处理像回答问题、提供信息或者促进用户互动之类的各种任务。通过在基于图的框架中构建对话历史和上下文关系，GraphRAG 系统能够增强模型生成连贯且与上下文相关响应的能力。 -（4）推荐系统：在电子商务平台的情境中，用户与产品之间的购买关系自然而然地形成了一个网络图。这些平台中的推荐系统的主要目标是预测用户未来的购买意向，有效地预测此图中的潜在连接。

7.2. 应用领域

7.2.1. 电子商务

通过**个性化推荐和智能客户服务**来改善客户的购物体验并提高销售额。

用户和产品之间的历史交互能够自然地形成一个图，它含蓄地封装了用户的行为模式和偏好信息。然而，鉴于电子商务平台数量的增多以及用户交互数据量的持续增长，运用 GraphRAG 技术提取关键子图变得至关重要。集成不同类型或具有不同参数的多个检索器以提取相关子图，随后对其进行编码以用于预测用户的时间行为。为了提升客户服务问答系统的模型性能，构建具有问题内和问题间关系的过去问题图。对于每一个给定的查询，检索类似过去问题的子图以提高系统的响应质量。

7.2.2. 生物医学

GraphRAG 技术在生物医学问答系统中的应用愈发广泛，实现了先进的医疗决策表现。在这一领域，每种疾病都与特定症状相关联，每种药物都包含针对和治疗特定疾病的某些活性成分。一些研究人员为特定任务场景构建知识图谱，而另一些研究人员则利用诸如 CMeKG 和 CPubMed-KG 等开源知识图谱作为检索源。现有的方法通常从非参数检索器的初始搜索开始，接着设计通过重新排序来过滤检索内容的方法。此外，有些方法提议使用检索到的信息重写模型输入以增强生成效果。

7.2.3. 学术

在学术研究领域，每一篇论文由一位或多位研究人员创作，并与某个研究领域相关联。作者隶属于机构，并且作者之间存在诸如合作或者共同的机构隶属关系等关联。这些元素能够构建为图的形式。在这个图上使用 GraphRAG 有助于学术探索，包括为作者预测潜在的合作者、识别特定领域内的趋势等等。

7.2.4. 文学

类似于学术研究，在文学领域能够构建知识图谱，其节点代表书籍、作者、出版商和系列，边则标注为“由……撰写”“在……出版”“图书系列”等。GraphRAG 可用于增强智能图书馆等实际应用。

7.2.5. 法律

在法律情境中，案例与司法意见之间存在着广泛的引用关联，因为法官在做出新决策时常常参考以往的意见。这自然形成了一个结构化的图，其中节点代表意见、意见集群、案件记录和法院，边涵盖了诸如“意见引用”“意见集群”“集群案件记录”“案件记录法院”等关系。GraphRAG 在法律场景中的应用能够助力律师和法律研究人员完成各类任务，比如案例分析和法律咨询。

7.2.6. 其他

除上述应用之外，GraphRAG 还应用于其他实际场景，如情报报告生成和专利短语相似性检测。首先构建事件情节图（Event Plot Graph，EPG）并检索事件的关键方面，以辅助生成情报报告。创建专利短语图并检索给定专利短语的自我网络，以协助判断短语相似性。

7.3. 基准和指标

7.3.1. 基准

用于评估 GraphRAG 系统性能的基准可分为两类。

- · 第一类是下游任务的相应数据集。
- · 第二类由专为 GraphRAG 系统设计的基准构成。这些基准通常涵盖多个任务领域，以提供全面的测试结果。

7.3.2. 指标

GraphRAG 的评估指标大致可分为两种主要类型：下游任务评估（生成质量）和检索质量。

7.3.2.1 下游任务评估（生成质量）

在大多数研究中，下游任务评估指标是评估 GraphRAG 性能的主要方式。例如，在 KBQA 中，精确匹配（EM）和 F1 分数通常用于衡量回答实体的准确性。此外，也有使用 BERT4Score 和 GPT4Score 来减少 LLM 生成与真实答案同义但不完全匹配的实体的情况。

在 CSQA 中，准确性是最常用的评估指标。对于诸如 QA 系统之类的生成任务，通常使用 BLEU、ROUGE-L、METEOR 等指标来评估模型生成文本的质量。

7.3.2.2 检索质量评估

在基于下游任务性能来评估 GraphRAG 可行的情况下，直接衡量检索内容的准确性颇具挑战。所以，许多研究运用特定的指标去衡量检索内容的精准度。比如，当有真实实体可用时，检索系统需要在检索信息的数量和答案覆盖范围之间找到平衡。因此，一些研究借助答案覆盖范围与检索子图大小之间的比率来评估检索系统的性能。此外，还有些研究探索了像查询相关性、多样性和保真度得分之类的指标，分别用以评估检索内容与查询的相似性、检索内容的多样性以及检索信息的保真度。

7.4. 工业中的 GraphRAG

- · GraphRAG（由微软）：此系统运用大型语言模型构建基于实体的知识图谱，并预先生成相关实体组的社区摘要，能够捕捉文档集合内的局部和全局关系，从而强化以查询为中心的摘要（QFS）任务。利用开源的 RAG 工具包快速实现，例如LlamaIndex、langchain等。

- · 网址：https://github.com/microsoft/graphrag

- · GraphRAG（由 NebulaGraph）：该项目是首个工业 GraphRAG 系统，由 NebulaGraph 公司开发。此项目将大型语言模型融入 NebulaGraph 数据库，旨在提供更智能、更精准的搜索结果。
  - · 网址：https://www.nebula-graph.io/posts/graph-RAG
- · GraphRAG（由 Antgroup）：该框架基于诸如 DB-GPT、知识图谱引擎 OpenSPG 和图形数据库 TuGraph 等数个 AI 工程框架开发而成。先是利用大型语言模型从文档中提取三元组，接着将其存于图形数据库中。在检索阶段，它从查询中识别关键字，在图形数据库中定位相应节点，并使用 BFS 或 DFS 遍历子图。在生成阶段，将检索到的子图数据格式化为文本，并与上下文和查询一同提交给大型语言模型处理。
  - · 网址：https://github.com/eosphoros-ai/DB-GPT
- · NallM（由 Neo4j）：NaLLM（Neo4j 和大型语言模型）框架将 Neo4j 图形数据库技术与大型语言模型相融合。它致力于探索并展示 Neo4j 与大型语言模型之间的协同作用，重点关注三个主要用例：知识图谱的自然语言接口、从非结构化数据创建知识图谱以及利用静态数据和大型语言模型数据生成报告。
  - · 网址：https://github.com/neo4j/NaLLM
- · LLM 图形构建器（由 Neo4j）：这是 Neo4j 开发的用于自动构建知识图谱的项目，适用于 GraphRAG 的图形数据库构建和索引阶段。该项目主要借助大型语言模型从非结构化数据中提取节点、关系及其属性，并利用 LangChain 框架创建结构化知识图谱。
  - · 网址：https://github.com/neo4j-labs/llm-graph-builder

6. 未来展望

- · **动态与自适应图：** 大多数 GraphRAG 方法基于静态数据库构建；然而，随着时间推移，新的实体和关系必然涌现。快速更新此类变化既前景广阔又充满挑战。纳入更新信息对于获取更优结果以及应对需要当下数据的新兴趋势至关重要。研发动态更新和新数据实时集成的有效方法，将显著提升 GraphRAG 系统的有效性和相关性。
- · **多模态信息整合：** 大多数知识图谱主要涵盖文本信息，缺少图像、音频和视频等其他模态，而这些模态有可能大幅提升数据库的整体质量和丰富程度。融入这些多样化的模态，能够对所存储的知识提供更全面、细致的理解。然而，此类多模态数据的整合面临着相当大的挑战。随着信息量的增加，图的复杂性和规模呈指数级增长，使其管理和维护愈发困难。这种规模的升级需要开发先进的方法和精良的工具，以高效处理并无缝将各种数据类型整合进现有的图结构，确保丰富的知识图谱的准确性和可访问性。
- · **可扩展且高效的检索机制：** 工业场景中的知识图谱可能包含数百万甚至数十亿个实体，规模庞大且错综复杂。然而，多数当代方法专为小规模知识图谱定制，此类图谱可能仅包含数千个实体。在大规模知识图谱中高效、有效地检索相关实体仍是一个切实且重大的挑战。开发先进的检索算法和可扩展的基础设施对于解决此问题至关重要，确保系统在维持高性能和实体检索准确性的同时，能够管理海量的数据量。

- · **与图基础模型相结合：** 近期，能够有效解决各类图任务的图基础模型取得显著成功。部署这些模型以增强当前的 GraphRAG 流程是一个关键问题。图基础模型的输入数据本质上是图结构的，使其能比 LLM 模型更高效地处理此类数据。将这些先进模型集成到 GraphRAG 框架中，能够极大提升系统处理和利用图结构信息的能力，从而提高整体性能和效能。
- · **检索上下文的无损压缩：** 在 GraphRAG 中，检索到的信息被组织成包含实体及其相互关系的图结构。随后，此信息被转换为 LLM 能够理解的序列，导致上下文极长。输入如此长的上下文存在两个问题：LLM 无法处理过长的序列，且推理过程中的大量计算可能对个人造成阻碍。为解决这些问题，长上下文的无损压缩至关重要。这种方法能去除冗余信息，将冗长的句子压缩成更短但仍有意义的句子。它有助于 LLM 捕捉上下文的关键部分并加速推理。然而，设计无损压缩技术颇具挑战性。当前的工作在压缩和保留信息之间进行权衡。开发有效的无损压缩技术对 GraphRAG 至关重要但困难重重。