大模型面经——以医疗领域为例，整理RAG基础与实际应用中的痛点

原创 喜欢卷卷的瓦力 瓦力算法学研所 2024年09月12日 10:00 上海

◈ 技术总结专栏 ◈

作者：喜欢卷卷的瓦力

RAG相关理论知识与经验整理。

谈到大模型在各垂直领域中的应用，一定离不开RAG，本系列开始分享一些RAG相关使用经验，可以帮助大家在效果不理想的时候找到方向排查或者优化。

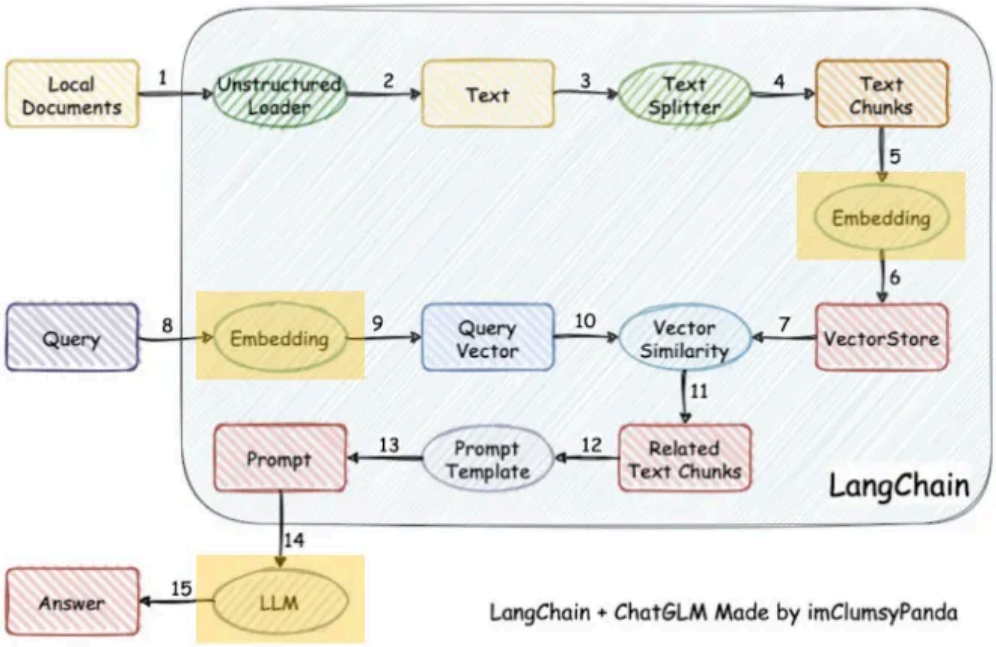
本系列以医疗领域为例，用面试题的形式讲解RAG相关知识，开始RAG系列的分享~

本篇主要是理论知识与经验；后续会结合最新的优化方法给出详细的优化代码，和实践中衍生的思考。

下面是本篇的快捷目录。

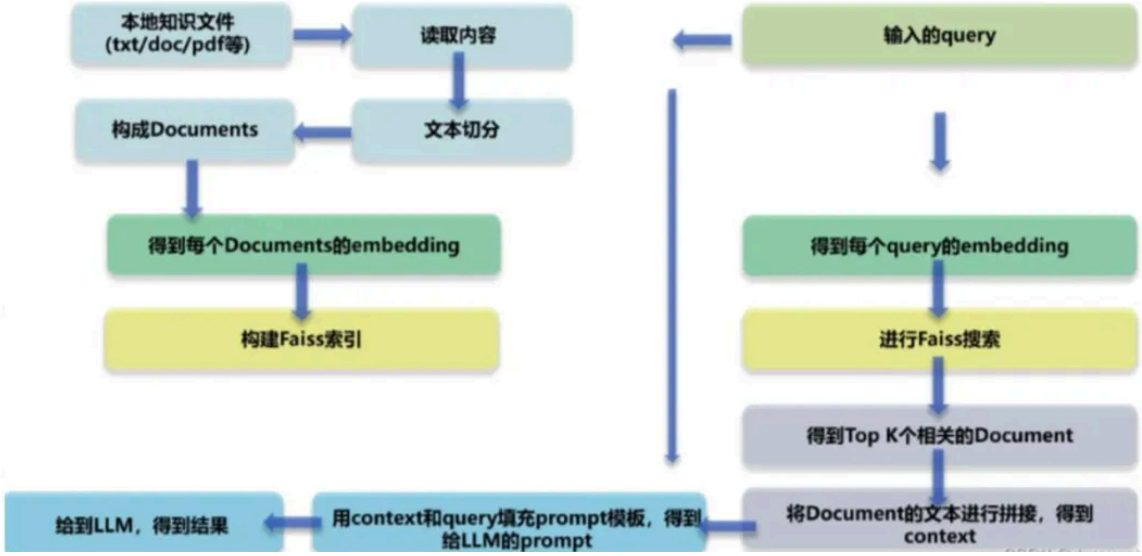
一、RAG思路

这里有一张经典的图：



具体步骤是：

- 加载文件
- 读取文本
- 文本分割
- 文本向量化
- 问句向量化
- 在文本向量中匹配出与问句向量最相似的top k个
- 匹配出的文本作为上下文和问题一起添加到 prompt 中
- 提交给 LLM 生成回答



二、RAG中的prompt模板

已知信息：{context}
根据上述已知信息，简洁和专业的来回答用户的问题。如果无法从中得到答案，请说 “根据已知信息无法回答该问题” 或 “没有提供足够的相关信息”，不允许在答案中添加编造成分，答案请使用中文。
问题是：{question}

其中 {context} 就是检索出来的文档。

三、检索架构设计

基于LLM的文档对话架构分为两部分，先检索，后推理。重心在检索（推荐系统），推理一般结合langchain交给LLM即可。

因此接下来主要是检索架构设计内容。

1. 检索要求

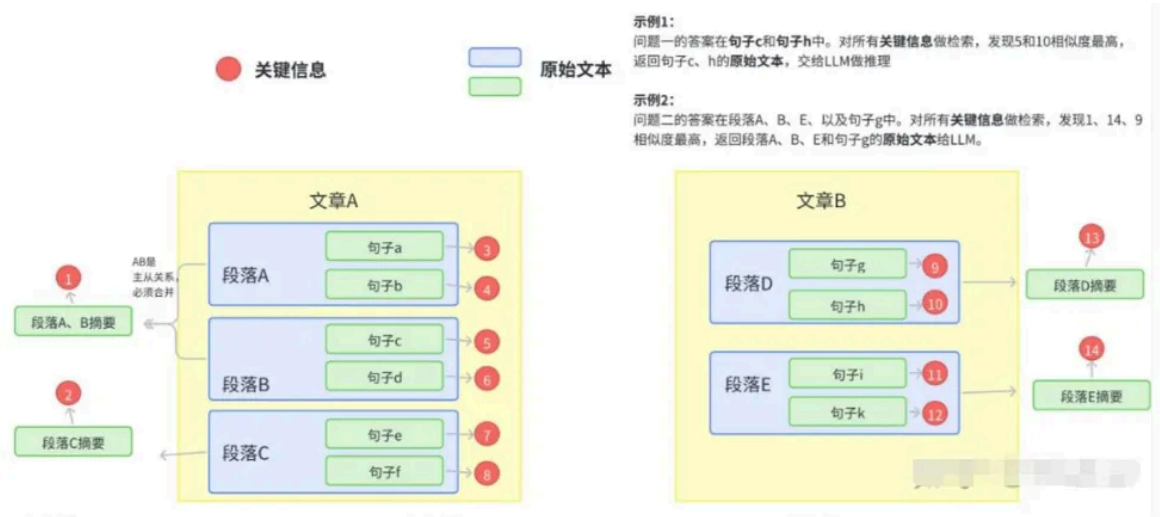
- 提高召回率
- 能减少无关信息
- 速度快

2. 检索逻辑

拿到需要建立检索库的文本，将其组织成二级索引，第一级索引是 [关键信息]，第二级是 [原始文本]，二者一一映射。 [关键信息]用于加快检索， [原始文本]用于返回给prompt得到结果。

向量检索基于关键信息embeddig，参与相似度计算，检索完成后基于关键信息与原始文本的映射，将原始文本内容作为 {context} 返回。

主要架构图如下：



3. 切分与关键信息抽取

关键信息抽取前需要先对拿到的文档进行切分。

其实文档切分粒度比较难把控，粒度过小的话跨段落语义信息可能丢失，粒度过大噪声又太多。因此在切分时主要是按语义切分。

因此拿到文档先切分再抽取关键信息，可根据实际情况考虑是否进行文章、段落、句子更细致粒度的关键信息抽取。

下面具体来讲讲方法和经验：

1）切分

- **基于NLP篇章分析（discourse parsing）工具**

提取出段落之间的主要关系，把所有包含主从关系的段落合并成一段。这样对文章切分完之后保证每一段在说同一件事情。

- **基于BERT中NSP（next sentence prediction）的训练任务**

基于NSP（next sentence prediction）任务。设置相似度阈值t，从前往后依次判断相邻两个段落的相似度分数是否大于t，如果大于则合并，否则断开。

2）关键信息抽取

- **直接存储以标点切分的句子**：只适用于向量库足够小（检索效率高）且query也比较类似的情况。
- **传统NLP工具**：成分句法分析（constituency parsing）可以提取核心部分（名词短语、动词短语……）；命名实体识别（NER）可以提取重要实体（货币名、人名、企业名……）。
- **生成关键词模型**：类似于ChatLaw中的keyLLM，，即：训练一个生成关键词的模型。在医疗领域中，这个方法是目前比较靠谱且能通用的方法。

参考资料

[1] GitHub - imClumsyPanda/Langchain-Chatchat-dev: Langchain-Chatchat 个人开发 Repo(https://github.com/imClumsyPanda/Langchain-Chatchat-dev)

[2] 基于LLM+向量库的文档对话痛点及解决方案 - 知乎 (zhihu.com) (https://zhuanlan.zhihu.com/p/651179780)

[3] LLM+Embedding构建问答系统的局限性及优化方案 - 知乎 (zhihu.com) (https://zhuanlan.zhihu.com/p/641132245)

想要获取技术资料的同学欢迎关注公众号，进群一起交流~