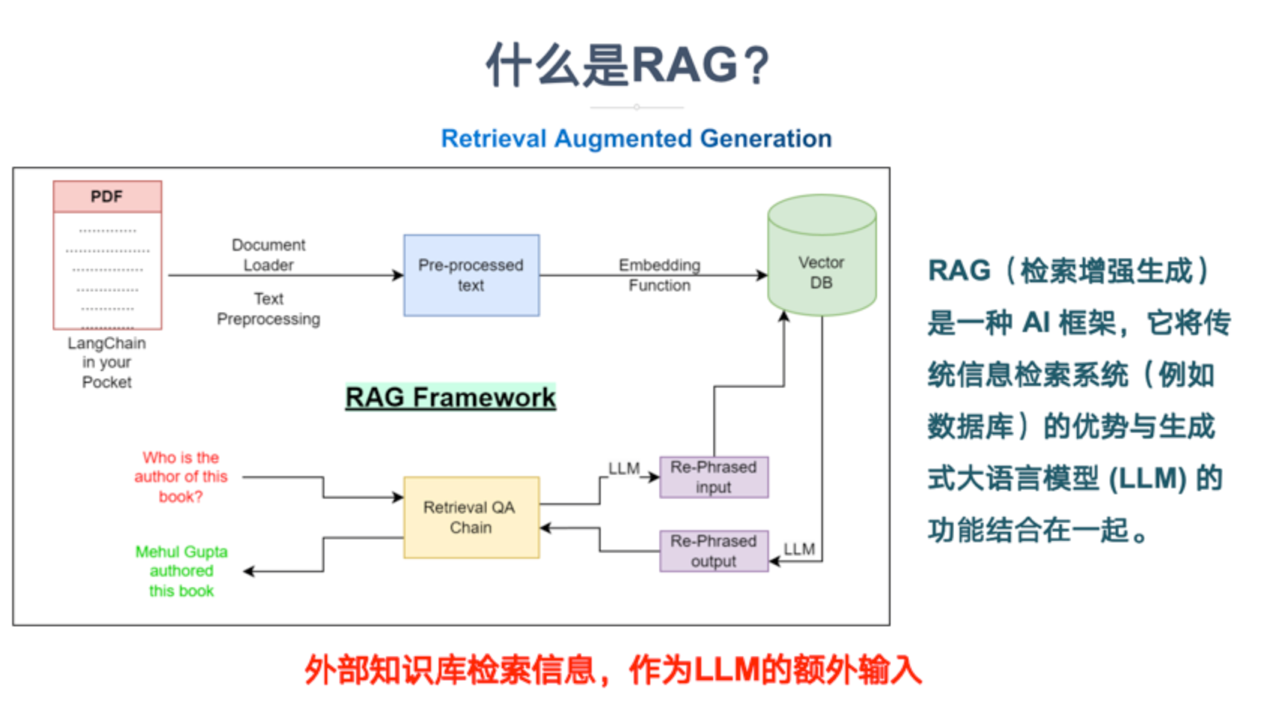
**大模型RAG（Retrieval-Augmented Generation）指南**

**一、什么是RAG？**

**RAG（检索增强生成，Retrieval-Augmented Generation）** 是一种将信息检索系统与大语言模型（LLM）结合的技术框架。与传统大模型单纯基于参数生成不同，RAG引入了“外部知识库”，模型在回答问题或生成内容时先从知识库中检索相关内容，再结合检索结果进行生成，从而显著提升了准确性、时效性和可控性。

典型的RAG系统可分为三个阶段：

1. **Query 构造**：将用户输入转化为适用于检索的查询。
2. **检索（Retrieval）**：使用向量数据库（如FAISS、Chroma、Weaviate）查找相关文档。
3. **生成（Generation）**：将检索到的文档与原始问题一起传入大模型，生成最终回答。



**二、RAG 的价值**

* **实时性增强**：不依赖模型预训练内容，支持最新数据。
* **可靠性提高**：回答基于文档，降低幻觉。
* **轻量更新**：只需更新知识库，无需重新训练模型。
* **合规与可控性**：可限制生成内容在知识边界内。

**三、RAG 的技术架构**

一个标准的 RAG 系统通常由以下模块组成：

csharp

复制编辑

[User Question]

↓

[Query Embedding] ← 使用 Embedding 模型（如 OpenAI Embedding、BGE、text2vec）

↓

[Vector Search] ← 基于向量数据库检索（如 Chroma、FAISS、Milvus）

↓

[Context Documents]

↓

[Prompt Template] ← 构造 Prompt，例如：“根据以下资料回答问题：…”

↓

[LLM Response] ← 使用 GPT-4、Claude、Gemini 等大模型生成最终回答

**技术选型建议：**

| **模块** | **推荐方案** |
| --- | --- |
| Embedding 模型 | OpenAI ada-002 / BGE / text2vec |
| 向量数据库 | Chroma / FAISS / Weaviate / Milvus |
| LLM 模型 | GPT-4 / Claude / LLaMA2 / Qwen |
| 中间件框架 | LangChain / LlamaIndex / Haystack |

**四、实现策略：如何构建一个 RAG 系统？**

**1. 文档处理（Document Preprocessing）**

* 格式支持：PDF、Markdown、DOCX、PPTX、HTML、TXT 等
* 分块（Chunking）：按段落或滑动窗口方式切片文本
* 元数据处理：保存来源、页码、时间戳等信息

**2. Embedding 与向量化**

* 使用Embedding模型生成每个chunk的向量表示
* 建议统一维度和归一化方式

**3. 向量数据库构建**

* 建立索引（Index）
* 支持相似度检索（通常为余弦相似度）

**4. 检索方式**

* **Similarity Search**：基于相似度的最近邻搜索（top-k）
* **MMR（Max Marginal Relevance）**：提升多样性，减少重复信息
* 可选 rerank 模块，如 Cohere Reranker、bge-reranker

**5. Prompt 构造**

基础模板示例：

makefile

复制编辑

你是一个专业助手。请根据以下提供的资料，准确回答用户的问题。

【资料】:

{context}

【问题】:

{question}

【回答】:

提示工程还可加入：

* 回答风格（简洁 / 专业 / 类似某产品）
* 回答来源引用（source tagging）

**6. 响应生成**

* 模型接收 Prompt 后输出答案
* 支持多轮对话：需带入历史问答（需要对上下文进行摘要或限制 token）

**五、进阶技巧**

**多轮对话记忆**

* 维护对话历史（LangChain Memory）
* 使用摘要压缩上下文，节省 token

**评估与优化**

* 使用评估指标如BLEU、ROUGE、GPT-4评审、人工评审
* 模拟问答对集做检索率（Recall@K）评估

**增强方式**

* **工具增强（Tool Use）**：RAG + 工具调用，如查天气、查股票
* **视觉增强（Visual RAG）**：支持图文并用的检索和问答
* **结构化RAG**：可对数据库、表格、API返回值等结构化内容进行检索

图示

AI 生成的内容可能不正确。

**六、常见问题解答（FAQ）**

1. **RAG 是否需要微调模型？**  
   不需要，模型使用零样本（zero-shot）能力即可，降低成本。
2. **RAG 是否能实时回答？**  
   只要文档及时入库，就可以实现实时性。
3. **检索失败怎么办？**  
   可设定 fallback 策略，例如直接 LLM 回答、提示用户无法回答等。
4. **如何避免生成错误引用？**  
   使用引用标签，并限制回答不能超出context内容。

**七、适用场景**

* 企业内部知识问答（SOP、HR政策、技术文档）
* 客服机器人（RAG 版 ChatBot）
* 智能文档摘要与分析
* 法律、医疗等专业领域的问答助手