什么是 RAG（Retrieval-Augmented Generation）？

RAG 是一种结合检索（Retrieval）和生成（Generation）的自然语言处理方法。

它通过引入外部知识库，使大语言模型能够在生成文本时引用最新的、任务相关的信息，从而提升模型的上下文理解能力和回答准确性。

RAG 的典型流程：

1.

用户输入查询（Query）

2.
3.

使用向量检索模型（如 BGE、GTE、M3E）将查询向量化

4.
5.

在向量数据库（如 FAISS）中检索最相关的文档片段

6.
7.

将这些文档与用户问题一并输入给大语言模型（如 Qwen、LLaMA、ChatGLM）

8.
9.

模型生成结合检索信息的自然语言回答

10.

RAG 的关键优势

-

使模型能够“查资料”而非“死记硬背”

-
-

支持个性化知识库（如公司文档、论文集、教材）

-
-

减少幻觉（Hallucination）现象，提升可靠性

-
-

可在低参数模型上通过增加知识覆盖增强能力

-

向量检索模型的常见选择：

-

thenlper/gte-small：轻量中英文通用检索模型

-
-

moka-ai/m3e-base：专为中文优化的向量模型

-
-

BAAI/bge-small-zh：开源中文向量检索模型，效果稳定

-
-

sentence-transformers/paraphrase-MiniLM：英文为主，速度快

-

如何切分文档片段?

文档切分（Chunking）是构建 RAG 系统的重要前置步骤。常用的切分策略包括:

-

固定字数（如每 300 字一个段落）

-
-

句子级分割（保持语义完整）

-
-

层次结构切分（基于标题、段落）

-

向量数据库推荐:

-

FAISS：轻量、支持本地 CPU 查询

-
-

Chroma：集成化文档库，适合快速原型

-
-

Weaviate / Milvus：适合海量数据的分布式部署

-

RAG + 本地模型的典型架构：

```
用户问题
   |
向量模型 (GTE/BGE)
   |
  FAISS
   |
匹配文档片段
   |
   ↓
本地 LLM (Qwen / LLaMA)
   ↓
结合上下文生成回答
```

RAG 应用案例

-

企业文档问答系统

-
-

医疗指南知识助手

-

-

法律法规智能检索

-
-

教材生成答疑系统