

基于 Qwen 和 LangChain 的 RAG 增强检索

一、系统介绍

基于 Retrieval-Augmented Generation (RAG) 的知识库问答系统通过结合信息检索与生成式模型，为知识库问答提供了一种高效且智能的解决方案。

二、主要技术与方法

1. 数据集构建

收集了近期的 30 篇新闻文本，涵盖体育、科技、民生等多个方面，文档以 markdown 形式保存；数据中包含一定的噪声文档，以提高检索的难度；根据文本内容设计了对应的问题和参考答案以进行 RAG 效果评估。

噪声文档

对新闻文本而言，要求模型生成的回答应该是最新发生的事。而在所有检索的文本中，加入了许多以前的新闻时间作为噪声数据。对于检索模型而言，这些数据很有可能因为相似度较高或关键词匹配多而成为检索文档，但生成模型应具备分别用户希望使用文本的能力。

QA 测试集

为定量评估 RAG 系统的效果，需要设计一定的 QA 对进行测试，包含提问 (query) 和参考答案 (ground truth)。为提高效率，本项目中的 QA 对利用大模型生成，然后人工进行二次检查与复筛。

评价标准

本系统中设置了两类问题。第一类是正常对话的 QA 集，共 60 个，所有内容均可在文本中找到答案，该测试侧重准确性、连贯性、引用质量；第二类为难题集，包含模糊提问、超纲知识、语义对抗（如矛盾描述）等，侧重评估模型的风险规避能力。

针对第一类问题，使用上下文精度、上下文召回、忠实度、回答相关性、回答正确性进行评估；针对第二类问题，使用拒绝率、安全响应率，且人工评价响应效果。

2. 文档分块与嵌入生成

LangChain 支持多种类型的文档输入，包含常用的 markdown, pdf, txt, doc 等；本项目中使用的文本数据经过清洗，均为 markdown 格式。

加载文本内容，使用 Langchain 提高的文本分割模块(text_splitter)并对文本进行分块处理，生成更小的知识单元，便于向量化存储和检索；本系统采用 text-embedding-v3 嵌入模型，生成高质量的语义向量表示，为检索的精确性提供支持。v3 相比 v1 和 v2，效果有明显提高。该模型默认支持最大 1024 向量维度，最大行数 20 行，最大输入长度 8192 为 token。向量化后的数据存储在向量数据库 Qdrant 中，便于进行相似度检索。

3. 检索增强流程

LangChain 提供的 retrievers 模块实现了多种检索器的封装，默认的向量数据库 vectorstore 对嵌入模型生成的语义向量进行管理，大多采用余弦相似度计算；为提高检索的效果，本系统采用了向量检索与关键词检索相结合的混合检索流程，弥补各自的不足，以提供更佳全面的搜索方案。

在混合检索的基础上，为应对复杂查询，本系统还对检索到的结果进行了优化排序。具体使用了 Cohere 提供的重排序工具，提高语义相似度高的段落的排名。

4. 大模型生成问答链

利用 LangChain 将检索与问答流程无缝集成：设计了规范化的提问模板，优化问题输入的格式和语义表达，提升回答生成的上下文相关性和准确性；结合历史对话，提高模型对提问的意图理解和多轮对话能力；推理阶段采用 Qwen-7B-Chat 模型（qwen-max）生成最终回答。

5. 模型评估

利用 RAGAs 框架对 RAG 系统的性能进行评估，评价指标包含上下文精度 (context precision)、上下文召回率 (context recall)、忠实度 (faithfulness)、答案相关性 (answer relevancy) 和答案正确性 (answer correctness)。

三、相关代码

1. 模型加载

使用 `langchain_community` 提供的接口加载模型。本系统使用到的模型包括文本嵌入模型和用于生成的大语言模型,采用 `dashscope` 提供的 `text-embedding-v3` 作为文本嵌入模型,生成模型则为 `Qwen-plus`。在加载前已提前将 `API-KEY` 配置到环境变量中。

```
from langchain_community.embeddings import DashScopeEmbeddings
from langchain_community.llms import Tongyi

# 生成模型
llm = Tongyi(model_name='qwen-plus')

# 嵌入模型
embeddings = DashScopeEmbeddings(model="text-embedding-v3")
```

2. 文档加载与嵌入生成

采用 `langchain` 提供的递归分块模块 `RecursiveCharacterTextSplitter` 对文本进行分块,通过预定义的文本分隔符(如换行符`\n\n`、`\n`, 句号、逗号等)迭代地将文本分解为更小的块,以实现段大小的均匀性和语义完整性。分割结果如图 1.1 所示。

```
from langchain.text_splitter import RecursiveCharacterTextSplitter
import os

# 初始化文本分割器
text_splitter = RecursiveCharacterTextSplitter(
    chunk_size=500, # 每个切片的字符数
    chunk_overlap=50, # 切片之间的重叠字符数
    separators=["\n\n", "\n", "。", " ", " "], # 切分依据
)

# 存储所有切片的列表
all_chunks = []
# 遍历 ./news 目录下的所有文件
for filename in os.listdir('./news'):
    if filename.endswith('.md'):
        file_path = os.path.join('./news', filename)

        with open(file_path, "r", encoding="utf-8") as file:
            markdown_content = file.read()

            # 输出文件字符数
            char_count = len(markdown_content)
            print(f"文件: {filename}, 字符数: {char_count}")
            # 将 Markdown 文本切片
            chunks = text_splitter.split_text(markdown_content)
            all_chunks.extend(chunks)

print(f"生成文本块数量: {len(all_chunks)}")
```

```
文件: 体育-23年欧冠.md, 字符数: 6679
文件: 体育-24年欧冠.md, 字符数: 1952
文件: 体育-东77.md, 字符数: 405
文件: 体育-内马尔.md, 字符数: 3390
文件: 体育-巴特勒.md, 字符数: 1185
文件: 体育-欧冠综合20250130.md, 字符数: 562
文件: 体育-欧冠综合20250131.md, 字符数: 1222
文件: 体育-欧冠附加赛赛制.md, 字符数: 2611
文件: 体育-福克斯.md, 字符数: 3047
文件: 体育-篮球欧冠.md, 字符数: 954
文件: 体育-英超20250131.md, 字符数: 3472
文件: 体育-金球奖.md, 字符数: 1146
文件: 健康-核桃.md, 字符数: 1510
文件: 军事-东部战区.md, 字符数: 1238
文件: 民生-春晚.md, 字符数: 5288
文件: 民生-电影20250130.md, 字符数: 273
文件: 民生-电影简介.md, 字符数: 1441
文件: 科技-deepseekr1.md, 字符数: 1823
文件: 财经-2024GDP.md, 字符数: 5831
生成文本块数量: 115
```

图 1.1 文本分割结果

2. 向量数据库存储与文本检索

将生成的文本块以向量形式储存在向量数据库 Qdrant 中, 为避免过往数据的干扰, 该系统只将数据保存在了内存中, 每次使用都要重新加载。针对大量文本的情况, 可以将数据库保存在本地, 方便后期使用。首先使用 Qdrant 创建 collection 实例, 然后定义 embedding 模型和 client 完成向量数据库 vector_store 的创建。向量检索的测试结果如图 2.1 所示。

```
from qdrant_client import QdrantClient
from qdrant_client.models import VectorParams, Distance
from langchain_qdrant import QdrantVectorStore

print("加载 Qdrant 向量数据库中")
client = QdrantClient(":memory:")
# if not exists then create collection
if not client.collection_exists("rag_collection"):
    # create collection
    client.create_collection(
        "rag_collection",
        vectors_config=VectorParams(
            size=len(embeddings.embed_query("hello world")), # size:1536
            distance=Distance.COSINE
        )
    )
vector_store = QdrantVectorStore(client=client, collection_name="rag_collection",
embedding=embeddings) # 初始化向量数据库
print("加载 Qdrant 向量数据库完成")
vector_store.add_texts(all_chunks) # 将切分好的文本嵌入到向量数据库中 (每次运行即将数据加入一遍)
print("嵌入生成完成, 向量数据库存储完成.")
print("索引过程完成.")

# 定义向量检索器
vector_retriever = vector_store.as_retriever(search_kwargs={"k": 5}) # 只检索 5 条
```

```
queries = ["欧冠直通16强淘汰赛的球队有哪些?", "曼城在附加赛的对手是谁?"]

vector_retriever = vector_store.as_retriever(search_kwargs={"k": 5}) # 只检索5条
# 批量查询, 每个 query 都会返回 k 个最相关的文档
results = [vector_retriever.get_relevant_documents(query, k=5) for query in queries]

# 展示所有查询结果
for i, result in enumerate(results):
    print(f"🔍 查询: {queries[i]}")
    for i, doc in enumerate(result):
        print(f"检索到的段落{i+1}: {doc.page_content[:100]}")
    print("-" * 50)

🔍 查询: 欧冠直通16强淘汰赛的球队有哪些?
检索到的段落1: ##### 欧冠综合 | 利物浦和巴萨直通16强
新华社 2025-01-30 11:12 北京
新华社柏林1月29日电 (记者刘畅) 本赛季欧冠联赛第一阶段29日结束全部8轮比赛, 利物浦、巴萨、阿森纳等积分榜
检索到的段落2: 意甲: 8支球队参加欧战尚存7支, 5支欧冠2支欧联1支欧协联。其中参加欧冠的5支球队中仅有国米以联赛第4名直接晋级淘汰赛, 亚特兰大、AC米兰和尤文图斯都需要通过附加赛来争
检索到的段落3: 在欧冠淘汰赛附加赛中, 常规赛排名最高的第9或第10位球队, 将对阵排名最低的第23或24位球队; 排名第11或第12位的球队, 将对阵排名第21或第22位的球队; 排名第13或第14位的
检索到的段落4: 面对36支球队采取联赛总排名的欧冠全新赛制, 有的传统强队有点懵圈, 曼城、皇马、拜仁都表现不佳。最后一轮联赛开始前, 竟然仅有利物和巴萨锁定16强入场券。9支种子队除了
检索到的段落5: 也就是说, 曼城必对皇马和拜仁之一。尽管曼城本赛季整体表现不佳, 但在冬季转会窗引进三名强援, 分别是前锋马尔穆什以及两名中卫库萨诺夫和雷斯。曼城三名强援没有
-----
🔍 查询: 曼城在附加赛的对手是谁?
检索到的段落1: **曼城对决**
在抽签仪式上, 首先抽出第23、第24名球队, 然后依次抽出剩余非种子球队。随后, 首先抽出第15、16名球队, 然后依次抽出种子球队。附加赛没有回避规则, 球队可以对阵同一足协球队以及联赛阶
检索到的段落2: ##### 欧冠附加赛抽签: 曼城再战皇马, 5年4次决战, 拜仁大巴黎好签
2025-01-31 19:29:28 来源: 奥拜尔
北京时间1月31日19时, 2024-25赛季欧冠淘汰赛附加赛抽签在瑞士尼
检索到的段落3: 也就是说, 曼城必对皇马和拜仁之一。尽管曼城本赛季整体表现不佳, 但在冬季转会窗引进三名强援, 分别是前锋马尔穆什以及两名中卫库萨诺夫和雷斯。曼城三名强援没有
检索到的段落4: AC米兰客场1:2不敌萨格勒布迪纳摩, 穆萨上半场吃到第二张黄牌, 被罚下场。拜仁慕尼黑主场3:1战胜布拉迪斯拉法。多特蒙德主场3:1击败顿涅茨克矿工。
附加赛将采取两回合制, 首回合将于2月11日打响。
检索到的段落5: ##### 欧冠新赛制格外刺激, 曼城将在附加赛对阵皇马或拜仁
2025-01-30 11:30 发布于: 广东省
史上第一次, 欧冠正赛在北京时间1月30日凌晨4时同时开始了多达18场比赛, 堪比足球春晚,
```

图 2.1 向量相似度检索

使用 LangChain 提供的 BM25Retriever 进行关键词检索。由于底层默认使用英文空格分词, 因此需要使用 jieba 分词库对中文文本进行分词处理, 否则将导致整个句子被视为单个词项, 导致检索效果不佳。如下修改 preprocess_func 参数, 传入 jieba 分词函数。

```
from langchain.retrievers import BM25Retriever, EnsembleRetriever
import jieba # 导入 jieba 库, 用于对中文文本进行分词处理

bm25_retriever = BM25Retriever.from_texts(all_chunks, preprocess_func=lambda
text: " ".join(jieba.cut(text)))
bm25_retriever.k = 5 # Retrieve top 5 results
```

图 2.2 和 2.3 分别展示了分词对检索效果的对比, 可以看出未分词时检索到的段落和问题完全无关, 分词后检索的结果显著提高。

```
查询: 欧冠直通16强淘汰赛的球队有哪些?
BM25 检索最相似的前 5 个文本块:
检索到的段落1: 事实上, 根据历年GDP最终核实数, 重庆经济总量在2020年就超过了辽宁, 2020年, 辽宁和重庆GDP分别为25011.4亿和25041.4亿元, 重庆领先30亿元; 2021年重庆继续领先辽宁; 但

检索到的段落2: ##### 内马尔回归母队, 这一次绝不是为了钱
澎湃新闻记者 陈均 2025-02-01 17:14 来源: 澎湃新闻
北京时间2月1日, 内马尔召开了回归巴西桑托斯俱乐部后的第一次新闻发布会, 会上内马尔坦

检索到的段落3: 内马尔表示: “巴西人民一直对我充满爱戴, 我也一直努力在球场上回报他们, 我需要找回踢球的信心和快乐, 这是我回归的目的, 我很高兴人们都在谈论如何拯救巴西足球的问题。”

检索到的段落4: 有消息称, 内马尔并不排除在这次短约结束后重返欧洲的可能性, 但一切要根据实际情况进行取舍, 有一点可以肯定, 眼下的内马尔更在乎的是如何恢复状态, 能够在下届世界杯到来前

检索到的段落5: ##### 巴特勒一心去太阳, 表态不会续约勇士, 谈判告吹
外媒报道, 金州勇士队与迈阿密热火队关于吉米-巴特勒的交易谈判已经结束, 因为巴特勒不肯与勇士续约告吹。金州勇士队在2024-25赛季的战绩并不是很
```

图 2.2 关键词检索（未对中文分词）

查询：欧冠直通16强淘汰赛的球队有哪些？
BM25 检索最相似的前 5 个文本块：
检索到的段落1：#### 五大联赛欧战积分排名，英超无愧第一联赛，意甲西甲相差无几
新欧洲三大杯联赛阶段已经全部结束，每个杯赛各有8支球队直接晋级下一阶段淘汰赛，16支球队将参加淘汰赛附加赛争夺另外8个晋级名额，还有
检索到的段落2：意甲：8支球队参加欧战尚存7支，5支欧冠2支欧联1支欧协联。其中参加欧冠的5支球队中仅有国米以联赛第4名直接晋级淘汰赛，亚特兰大、AC米兰和尤文图斯都需要通过附加
检索到的段落3：德甲：7支球队参加欧战，4支欧冠2支欧联杯1支欧协联。其中参加欧冠的勒沃库森直接晋级淘汰赛，多特蒙德和拜仁将参加淘汰赛附加赛，斯图加特和莱比锡被直接淘汰。参加
检索到的段落4：面对36支球队采取联赛总排名的欧冠全新赛制，有的传统强队有点懵圈，曼城、皇马、拜仁都表现不佳。最后一轮联赛开始前，竟然仅有利物浦和巴萨锁定16强入场券。9支种
检索到的段落5：#### 欧冠新赛制格外刺激，曼城将在附加赛对阵皇马或拜仁
2025-01-30 11:30 发布于：广东省
史上第一次，欧冠正赛在北京时间1月30日凌晨4时同时开始了多达18场比赛，堪比足球春晚，

图 2.3 关键词检索（分词后）

最后使用 EnsembleRetriever 对两种检索方法的结果进行合并，实现混合检索。该方法中可以对两种检索方式的权重进行分配，可以看出与 query:" 曼城在附加赛的对手是谁？"排在了前面。

```
ensemble_retriever = EnsembleRetriever(retrievers=[bm25_retriever, vector_retriever], weights=[0.4, 0.6])

processed_query = ". ".join(jieba.cut(query))
ensemble_retriever.get_relevant_documents(processed_query)
```

✓ 0.2s Python

, page_content='#### 欧冠附加赛抽签：曼城再战皇马，5年4次决战，拜仁大巴黎好签\n2025-01-31 19:29:28\n来源：奥拜尔 \n北京时间1月31日19时，2024-25赛季欧冠淘汰赛附加赛
, page_content='#### 欧冠新赛制格外刺激，曼城将在附加赛对阵皇马或拜仁 \n2025-01-30 11:30 发布于：广东省\n史上第一次，欧冠正赛在北京时间1月30日凌晨4时同时开始了多达18场比
, page_content='在欧冠淘汰赛附加赛中，常规赛排名最高的第9或第10位球队，将对阵排名最低的第23或24位球队；排名第11或第12位的球队，将对阵排名第21或第22位的球队；排名第13或第1
_id': '6521ea97d6ed475db8b47d368bc0c81f', '_collection_name': 'rag_collection'}, page_content='**曼城对决**\n在抽签仪式上，首先抽出第23、第24名球队，然后依次抽出剩
_id': 'f91deffe4c2d4e3eac941f786e7ccdb2', '_collection_name': 'rag_collection'}, page_content='也就是说，曼城必对皇马和拜仁之一。尽管曼城本赛季整体表现不佳，但在冬
_id': '73196a726e94aa3aef90c664a26f930', '_collection_name': 'rag_collection'}, page_content='AC米兰客场1:2不敌萨格勒布迪纳摩，穆萨上半场吃到第二张黄牌，被罚下场。
_id': 'a224317e70724c3caf19f55f164968d2', '_collection_name': 'rag_collection'}, page_content='#### 欧冠综合 | 利物浦和巴萨直通16强\n新华社 2025-01-30 11:12 北京
_id': '194418b2e38442319be40d555763247b', '_collection_name': 'rag_collection'}, page_content='#### 欧冠：曼城总分5-1淘汰皇马，3年2进决赛将战国米，8席梅开二度\n北
_id': '7f3856cd33554efa9611e6fec9af1e29', '_collection_name': 'rag_collection'}, page_content='尽管和国米一样打明牌，但一到关键战役就慢热的曼城，并没能像此前两轮赛
_id': 'de666ea5578d46d4a2a7eeb46be69a9e', '_collection_name': 'rag_collection'}, page_content='曼城半场吊打皇马，控球82开，射门13比1，射正5比0。皇马在对手禁区只触
, page_content='#### 巴尔扎雷蒂：米兰总是先丢球没进步；尤文图斯前场创造力不高\n小科爱足球 2025-01-30 14:56\n费德里科·巴尔扎雷蒂，这位前乌迪内斯高层、现负责罗马球探和租借
, page_content='更难能可贵的是，眼下瓜迪奥拉的主力阵容，固然不乏加盟时叫价惊人的存在，但打出身价，不辱投资者是大多数：哈兰德和迪亚斯都是菜鸟季即巅峰，德布劳内、8席、罗德里

图 2.4 混合检索

3. 重排序

本系统对比了两种重排序方法。首先使用了 LangChain 提供的长上下文重排序 LongContextReorder 模块，结果如图 3.1 所示。可以看出，该方法有一定效果，如之前排在第 6 名的段落，经过重排到了第三名，但效果仍然一般。

```
from langchain_community.document_transformers import LongContextReorder

reordering = LongContextReorder()
reordered_docs = reordering.transform_documents(docs)
```

问题：欧冠直通16强淘汰赛的球队有哪些？
混合检索结果：
检索到的段落1：意甲：8支球队参加欧战尚存7支，5支欧冠2支欧联1支欧协联。其中参加欧冠的
检索到的段落2：面对36支球队采取联赛总排名的欧冠全新赛制，有的传统强队有点懵圈，曼城、皇
检索到的段落3：德甲：7支球队参加欧战，4支欧冠2支欧联杯1支欧协联。其中参加欧冠的勒沃
检索到的段落4：#### 五大联赛欧战积分排名，英超无愧第一联赛，意甲西甲相差无几
新欧洲三大杯联赛阶段已经全部结束，每个杯赛各有8支球队直接晋级下一阶段淘汰赛，16支球队将
检索到的段落5：#### 欧冠新赛制格外刺激，曼城将在附加赛对阵皇马或拜仁
2025-01-30 11:30 发布于：广东省
史上第一次，欧冠正赛在北京时间1月30日凌晨4时同时开始了多达18场比赛，堪比足球春晚，
检索到的段落6：#### 欧冠综合 | 利物浦和巴萨直通16强
新华社 2025-01-30 11:12 北京
新华社柏林1月29日电（记者刘旸）本赛季欧冠联赛第一阶段29日结束全部8轮比赛，利物浦、巴
检索到的段落7：在欧冠淘汰赛附加赛中，常规赛排名最高的第9或第10位球队，将对阵排名最低
检索到的段落8：也就是说，曼城必对皇马和拜仁之一。尽管曼城本赛季整体表现不佳，但在冬
检索到的段落9：#### 欧冠附加赛抽签：曼城再战皇马，5年4次决战，拜仁大巴黎好签
2025-01-31 19:29:28 来源：奥拜尔
北京时间1月31日19时，2024-25赛季欧冠淘汰赛附加赛抽签在瑞士尼
检索到的段落10：**曼城对决**
在抽签仪式上，首先抽出第23、第24名球队，然后依次抽出剩余非种子球队。随后，首先抽出第
重排序后结果：
混合检索结果：
检索到的段落1：面对36支球队采取联赛总排名的欧冠全新赛制，有的传统强队有点懵圈，曼城、皇
检索到的段落2：#### 五大联赛欧战积分排名，英超无愧第一联赛，意甲西甲相差无几
新欧洲三大杯联赛阶段已经全部结束，每个杯赛各有8支球队直接晋级下一阶段淘汰赛，16支球队将
检索到的段落3：#### 欧冠综合 | 利物浦和巴萨直通16强
新华社 2025-01-30 11:12 北京
新华社柏林1月29日电（记者刘旸）本赛季欧冠联赛第一阶段29日结束全部8轮比赛，利物浦、巴
检索到的段落4：也就是说，曼城必对皇马和拜仁之一。尽管曼城本赛季整体表现不佳，但在冬
检索到的段落5：**曼城对决**
在抽签仪式上，首先抽出第23、第24名球队，然后依次抽出剩余非种子球队。随后，首先抽出第1
检索到的段落6：#### 欧冠附加赛抽签：曼城再战皇马，5年4次决战，拜仁大巴黎好签
2025-01-31 19:29:28 来源：奥拜尔
北京时间1月31日19时，2024-25赛季欧冠淘汰赛附加赛抽签在瑞士尼
检索到的段落7：在欧冠淘汰赛附加赛中，常规赛排名最高的第9或第10位球队，将对阵排名最低
检索到的段落8：#### 欧冠新赛制格外刺激，曼城将在附加赛对阵皇马或拜仁
2025-01-30 11:30 发布于：广东省
史上第一次，欧冠正赛在北京时间1月30日凌晨4时同时开始了多达18场比赛，堪比足球春晚，
检索到的段落9：德甲：7支球队参加欧战，4支欧冠2支欧联杯1支欧协联。其中参加欧冠的勒沃库
检索到的段落10：意甲：8支球队参加欧战尚存7支，5支欧冠2支欧联1支欧协联。其中参加欧冠的

图 3.1 LongContextReorder 重排序

第二种方法则是使用 cohere 提供的 Rerank 工具，该方法需要付费并配置 API-KEY。重排序结果如图 4.2 所示，该方法的重排序结果明显优于第一种，与问题相关的段落排在了第一名。

```
# 方法二：cohere
from langchain.retrievers import ContextualCompressionRetriever
from langchain.retrievers.document_compressors import CohereRerank
```

```
# 初始化 Cohere 重排序器
os.environ["COHERE_API_KEY"] = ""

compressor = CohereRerank(model="rerank-english-v3.0",top_n=5) # 重排序后保留前 5 个
compression_retriever = ContextualCompressionRetriever(
    base_compressor=compressor,
    base_retriever=ensemble_retriever # 使用之前的混合检索器
)
```

使用cohere重排序的结果：

检索到的段落1 :#### 欧冠综合 | 利物浦和巴萨直通16强

新华社 2025-01-30 11:12 北京

新华社柏林1月29日电（记者刘旸）本赛季欧冠联赛第一阶段29日结束全部8轮比赛，利物浦、巴萨、阿森纳等积分榜

检索到的段落2 :在欧冠淘汰赛附加赛中，常规赛排名最高的第9或第10位球队，将对阵排名最低的第23或24位球队；排名第11或第12位的球队，将

检索到的段落3 :#### 五大联赛欧战积分排名，英超无愧第一联赛，意甲西甲相差无几

新欧洲三大杯联赛阶段已经全部结束，每个杯赛各有8支球队直接晋级下阶段淘汰赛，16支球队将参加淘汰赛附加赛争夺另外8个晋级名额，还有

检索到的段落4 :面对36支球队采取联赛总排名的欧冠全新赛制，有的传统强队有点懵圈，曼城、皇马、拜仁都表现不佳。最后一轮联赛开始前，竟

检索到的段落5 :德甲：7支球队参加欧战，4支欧冠2支欧联杯1支欧协联。其中参加欧冠的勒沃库森直接晋级淘汰赛，多特蒙德和拜仁将参加淘汰赛

图 3.2 Cohere 重排序

4. 生成流程设计

设计提问模板，结合 LangChain 的提示模板 ChatPromptTemplate 和检索链 create_retrieval_chain 进行标准 RAG 流程设计。使用其他检索方式只要直接替换即可。

```
from langchain.chains.combine_documents import create_stuff_documents_chain
from langchain.chains.retrieval import create_retrieval_chain
from langchain.prompts.chat import ChatPromptTemplate

# 方法一：常规 rag（不包含历史记录）
# 1. 生成回答的 prompt
def get_simple_answer_prompt():
    system_prompt = """\
你是一个问答任务的助手，请依据以下检索出来的信息去回答问题，回答的字数控制在 100 字内：
{context}
"""
    return ChatPromptTemplate([
        ("system", system_prompt),
        ("human", "{input}") # 直接使用当前问题，不添加历史记录
    ])

# 2. 创建向量索引器
vector_retriever = vector_store.as_retriever(search_kwargs={"k": 10})

# 3. 生成回答的 chain
vector_qa_prompt = get_simple_answer_prompt()
vector_qa_chain = create_stuff_documents_chain(llm, vector_qa_prompt)

# 4. 创建不使用历史的 RAG 链
vector_rag_chain = create_retrieval_chain(vector_retriever, vector_qa_chain)
```

为提高模型多轮对话和意图理解的能力，为常规的 RAG 问答链添加历史记录和问题改写的功能。首先定义改写问题的 Prompt 和改写链。

```
def get_contextualize_question_prompt():
    """
    基于历史记录来改写用户问的问题
    :return:
    """
    system_prompt = """\
    请根据聊天历史和最后用户的问题，改写用户最终提出的问题。
    你只需要改写用户最终的问题，请不要回答问题
    没有聊天历史则将用户问题直接返回，有聊天历史则进行改写
    """
    contextualize_question_prompt = ChatPromptTemplate([
        ("system", system_prompt),
        MessagesPlaceholder("chat_history"),
        ("human", "{input}")
    ])
    return contextualize_question_prompt

# contextualize question
question_prompt = get_contextualize_question_prompt()
# 改写链：结合上下文改写用户问题
history_aware_retriever = create_history_aware_retriever(llm, retriever,
question_prompt)
```

问答链的提示词和常规 RAG 相同，但也结合了历史记录。

```
def get_answer_prompt():
    system_prompt = """\
    你是一个问答任务的助手，请依据以下检索出来的信息去回答问题，回答的字数控制在 100 字内：
    {context}
    """
    qa_prompt = ChatPromptTemplate.from_messages([
        ("system", system_prompt),
        MessagesPlaceholder("chat_history"),
        ("human", "{input}")
    ])
    return qa_prompt

# qa chain
qa_prompt_template = get_answer_prompt()
# 问答链：根据问题和参考内容生成答案
qa_chain = create_stuff_documents_chain(llm, qa_prompt_template)
```

最后创建 RAG 链。

```
rag_chain = create_retrieval_chain(history_aware_retriever, qa_chain)
```

定义一个字典 store 用于管理聊天历史，然后再 RAG 链中添加历史记录。

```
store = {}
# 获取历史记录
def get_session_history(session_id:str):
    if session_id not in store:
        store[session_id] = ChatMessageHistory()
```

```

    return store[session_id]

# 在 rag_chain 中添加 chat_history
conversational_rag_chain = RunnableWithMessageHistory(
    rag_chain,
    get_session_history,
    input_messages_key="input",
    history_messages_key="chat_history",
    output_messages_key="answer"
)

# 改写用户内容部分（隐式）
contextualize_question_chain = RunnableWithMessageHistory(
    question_prompt | llm,
    get_session_history,
    input_messages_key="input",
    history_messages_key="chat_history"
)

```

5. 模型评估

首先将xlsx中的QA对导入程序中,保存在pandas的数据表dataframe中,分别命名为'question'和'ground_truth'。然后遍历问题,并将生成的预测(答案)和检索到的文本写入数据表中,分别命名为'answer'和'retrieved_contexts'。然后将该数据表封装为Dataset格式,方便Ragas使用。

```

# 构建测试数据集
import pandas as pd
file_path = "qa_dataset.xlsx"
df = pd.read_excel(file_path, sheet_name="positive")

questions = df["question"].astype(str).tolist()
ground_truths = df["ground_truth"].astype(str).tolist()

from datasets import Dataset

answers = []
contexts = []

for query in questions:
    # 方法一: 向量检索
    # answer = vector_rag_chain.invoke({"input": query})
    # answers.append(answer['answer'])
    # # 只存储相关文本, 确保格式匹配
    # context = vector_retriever.get_relevant_documents(query)
    # contexts.append([doc.page_content for doc in context])

    # 方法二: 混合检索
    # answer = hybrid_rag_chain.invoke({"input": query})
    # answers.append(answer['answer'])
    # context = ensemble_retriever.get_relevant_documents(query)

```



```

# contexts.append([doc.page_content for doc in context])

# 方法三：重排序
answer = rrf_rag_chain.invoke({"input": query})
answers.append(answer['answer'])
context = compression_retriever.get_relevant_documents(query)
contexts.append([doc.page_content for doc in context])

data = {
    "question": questions,
    "ground_truth": ground_truths,
    "answer": answers,
    "retrieved_contexts": contexts
}

# Convert dict to dataset
dataset = Dataset.from_dict(data)

```

使用 Ragas 对结果进行评估，指标包含上下文精度和召回，忠实度，回答相关度和回答正确性。

```

from ragas import evaluate
from ragas.metrics import (
    faithfulness,
    context_recall,
    context_precision,
    answer_relevancy,
    answer_correctness
)

result = evaluate(
    dataset = dataset,
    metrics=[
        context_precision,
        context_recall,
        faithfulness,
        answer_relevancy,
        answer_correctness
    ],
)

df = result.to_pandas()
df

```

四、实验结果与系统测试

1. 定量 QA 测试

从表 1 结果来看，混合检索取得了比混合检索重排序更高的指标，尤其在召回率和回答忠实度两方面。

	context_precision	context_recall	faithfulness	answer_relevancy	answer_correctness
向量检索	83.26%	90.82%	89.04%	77.37%	62.58%
混合检索	80.76%	93.60%	93.33%	79.48%	60.04%

混合检索重排	78.23%	88.94%	89.84%	78.37%	58.71%
--------	--------	--------	--------	--------	--------

表 4.1. 常规 RAG 的不同检索方法对比

2. 多轮对话与模糊输入测试

在多轮对话常见下，使用了模糊指代的提问，对比图 4.1 和 4.2，不难发现常规 RAG 面对指代不清的问题，回答不够肯定坚决，但结合历史记录后，RAG 系统能够坚定地回答且正确。

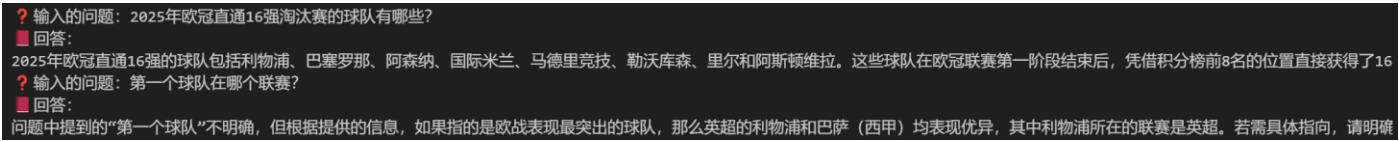


图 4.1 常规 RAG 对话

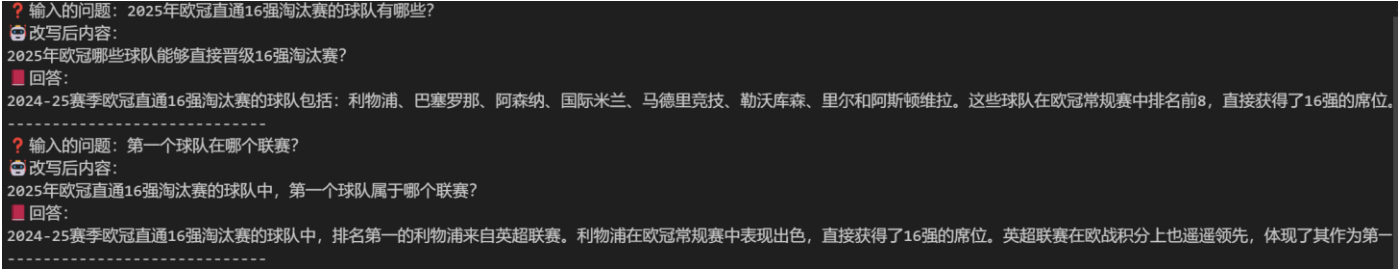


图 4.2 结合历史记录的 RAG 对话