ChatGPT

引入多步检索推理机制提升金融RAG系统在复杂金融查询中的性能

1. **研究背景与动机**
   在金融领域，专业用户提出的问题往往涉及最新的市场数据、法规条文或财务报告细节。传统的大型语言模型（LLM）在金融问答中容易出现事实不准确或知识滞后的问题。为提高准确性，检索增强生成（Retrieval-Augmented Generation，简称 **RAG**）系统应运而生：通过先检索相关文档，再由模型基于检索结果生成答案，以确保回答基于真实资料。然而，现有的大多数金融RAG系统主要针对单一文档或简单查询，面对**复杂金融查询**时仍存在性能不足的现象。复杂查询指的是需要跨越多个段落甚至多个文档进行推理，或需要从数据中计算数值结果的问题（例如：“比较公司A过去两年的营收增长率”这类需要跨段检索并计算的问题）。这类任务对系统提出了更高要求：模型不仅要理解金融术语和缩写（如“MS”表示Morgan Stanley，**YOY**表示同比增长），还要能够进行多步推理：先后检索不同片段的信息，并将它们关联起来推导答案。

近期发布的 **FinDER** 数据集（Financial Domain Expert Retrieval 数据集）凸显了这一挑战。FinDER由金融领域专家构建，包含5,703个真实金融问答三元组（查询-证据-答案）。其中许多查询既简短又含糊，充满行业特有的缩写和术语，要求系统具备解释缩写含义、识别查询意图的能力【例如，将“MS”解析为Morgan Stanley】。更棘手的是，不少查询的信息散落在不同段落甚至不同文件中，需要**跨段落的多步检索**和**推理计算**才能得出正确答案。据统计，FinDER中的大部分问题是质询类（如询问公司政策、财报细节），约有一小部分涉及定量计算，而这些定量问题中又有相当比例需要复杂的多步骤推理。这表明当前的金融RAG系统在真实场景下会遇到多方面挑战：如何精准检索关联内容，如何跨越多个证据片段进行逻辑推导，以及如何处理数值计算和比较。

动机： 为了提高金融问答系统在上述复杂任务中的表现，我们希望引入**多步检索推理机制**。与一次检索直接回答不同，多步检索允许系统像分析员一样逐步深入：先检索初步相关的信息，根据初步结果进一步细化查询并检索下一个相关片段，反复迭代，直到汇集足够证据支持最终回答。这种方法有望弥补传统语义检索的不足，避免遗漏那些表面不直接相关但对推理链条关键的证据。此外，在推理过程中融入显式的算术计算模块，也能提升数值类问答的准确度。**本研究的目标**是构建一个开源且可复现的金融RAG系统，通过创新的方法链路，在不借助闭源工具的前提下，实现对复杂金融查询更高的回答准确率和可靠性。项目周期有限（约两个月用于模型训练和调优），因此方案设计上将优先利用开源预训练模型及现有工具，侧重方法创新和有效整合，以在有限时间内取得显著性能提升。

1. **问题定义**
   本研究聚焦于**复杂金融查询的检索问答问题**。正式地，可以将其定义为：给定一个来自金融领域的用户查询 q（可能涉及财务指标、法规条款、公司简称等），以及一个大型金融文本语料库 D（如年度报告、财务报表、新闻稿等），系统需要返回一个准确且有依据的答案 a。这里，“复杂”指查询可能需要**多跳检索**（multi-hop retrieval）才能找到完整答案：即答案所需的信息分别存在于 D 中不同的段落/文件，需要系统通过多次查询-检索迭代进行推理整合。此外，如果查询要求数值计算（如同比增长、差值计算等），系统必须从检索结果中抽取相关数字并执行正确的算术运算。

要解决这一问题，传统RAG系统面临以下子任务和挑战： - **查询解析：** 金融查询常包含缩略词和专业术语，系统需要正确识别其中的实体（如公司、人名、指标）和意图。例如，将“MS去年的净利润同比增长多少”解析为“Morgan Stanley上一年度净利润同比增长率是多少”。 - **证据检索：** 从大规模金融语料 D 中检索出能回答查询的证据文段。复杂查询下，这往往需要**多步**：初步检索一个相关段落后，可能需要根据其中提取的新线索（如相关年份、子问题）再检索下一个段落。 - **证据推理与整合：** 对检索到的多个证据段落进行逻辑推理和信息聚合。有些问题需要比较不同段落的信息，有些需要将数值代入公式计算。这一步要求系统能理解证据间的关系，例如确定两个段落提到的是同一指标的不同时点，然后计算增长率。 - **答案生成：** 在掌握充分证据的基

础上，生成最终答案。答案需要准确表述所求信息，并尽可能给出基于证据的解释或出处（如果需要的话）。在生成过程中还应避免引入未验证的内容。

研究问题转化：因此，我们的问题可归纳为**构建一种包含多步检索和推理的问答系统**，该系统能够： 1. **理解复杂金融查询**（包括歧义消解和实体识别）； 2. **通过多次检索获得完整证据链**（涵盖查询所涉及的各个方面信息）； 3. **在需要时执行数值计算和跨证据推理**； 4. **给出准确且基于证据的答案**。

解决方案需在技术上满足开源和可复现性要求，即尽量使用开放的数据和模型；在时间上可行，两个月内完成模型微调与验证；在性能上相较基线方法有明确提升，尤其是在FinDER数据集定义的复杂查询子集中取得更高的准确率。

1. **方法框架**
   为实现上述目标，我们设计了一套**多步检索-推理的模块化框架**。系统整体采用流水线方式处理查询，各模块分工明确又相互衔接。下面按处理流程介绍各模块及其职责：

2. **模块1：查询理解模块**
   职责： 解析用户输入的金融查询，识别其中的关键实体和指标，澄清查询意图，并对可能存在的歧义进行消解。该模块会将原始查询转化为系统可处理的规范化形式（例如扩展简称、补全隐含信息），为后续检索做好准备。
   输入： 用户原始查询 q（文本）。
   输出： 规范化的查询表示 q′，包含明确的实体标识、属性或子问题。必要时，还输出查询的结构化信息，例如识别出的实体清单、关系类型（如“比较”、“计算增长率”）等供后续模块使用。
   实现选项：

3. 基于规则与词典的方法： 利用金融领域的专业词典和规则库对查询进行预处理。例如，将常见公司代号（Ticker）或缩写替换为全称，将“YOY”解析为“Year-over-Year（同比）”。这种方法实现简单且明确，可快速覆盖已知缩略语。
4. 命名实体识别（NER）和实体链接： 应用开源金融领域NER模型（如在FinBERT等预训练模型上微调的NER）识别查询中的公司、人物、指标等实体，然后通过知识库将缩写映射到标准实体（例如“MS”→“Morgan Stanley”）。
5. LLM语义解析： 采用一个指令微调的大模型直接生成对查询的解释。例如提示模型：“将下面的财经问题改写为清晰完整的问题：…”。模型输出扩展后的明确查询以及可能的分解。这种方法依赖模型的语言理解能力，能处理灵活表述，但需确保使用开源且可控的模型。

（模块1是可选但推荐的步骤，良好的查询理解将显著提高后续检索的准确性。例如，对于含糊查询“MS盈利如何”，模块1会将其转换为“Morgan Stanley 2023年度的净利润是多少”或类似明确的问题。）

- **模块2：多步检索与推理模块**
  职责： 基于解析后的查询 q′，从金融文档语料库中检索相关证据段落。不同于一次性检索直接完成，本模块采用**迭代检索**结合推理：初步检索获取一批候选证据，然后根据需要进行分析推理，形成新的检索线索，执行下一轮检索。该模块相当于系统的大脑，负责规划检索策略和整合跨步的信息，以构建完整的证据链。
  输入： 规范化查询 q′，以及文档语料库 D（已建立索引，供快速查询）。
  输出： 一个经过挑选和组织的**证据集合** E = {e1, e2, …, en}，其中包含能够共同回答查询的多个文档段落，每个 e 都带有出处信息。还可能输出一个简单的推理路径或证据间关系的说明，供下游生成参考。
  内部流程： （此模块内含多个子步骤，可能以循环方式执行）：
- **初始检索（子步骤2.1）：** 根据 q′ 执行首次检索，获取与查询语义相关的前若干条文档段落（例如使用向量检索得到前10个候选段落）。采用**稀疏检索**（BM25等关键词检索）或**密集检索**（基于句向量相似度）或两者结合的方式，确保初步检索具有较高召回率。
- **证据分析与推理（子步骤2.2）：** 对初始检索结果进行检查，判断是否已经涵盖回答所需的所有信息。如果答案需要多个证据，则识别当前缺失的部分。例如，在查询“公司A过去两年的营收增长率”中，

如果初次检索到了去年营收数据但缺少前年的数据，系统将识别这一缺口。此分析可以通过规则（检查是否所有所需年份都有数据）或通过LLM产生**推理链**来完成。一个实现策略是使用LLM生成“解题思路”，例如：①找到公司A去年营收，②找到公司A前年营收，③计算增长率。对照已检索的证据，看哪些步骤尚未完成。

- **迭代检索（子步骤2.3）：** 针对上一步发现的证据缺口，构造新的检索查询并再次查询语料库。新查询可以利用已获取的信息（如上一段找到的公司A去年的营收值）来更准确定位剩余信息（如该公司前年的营收段落）。通过多轮检索-推理循环，系统逐步补全所有所需证据。当再次检索不再发现新的有用内容，或已满足查询的所有子需求时，循环终止。
- **证据筛选与排序（子步骤2.4）：** 在迭代结束后，可能收集到了较多相关片段。本步骤对证据集合 E 进行过滤和排序，淘汰噪声或冗余信息，仅保留最有支撑力的若干段落（例如3-5段），并按逻辑顺序组织（例如按照推理顺序或信息来源先后排列）。可以通过简单的评分函数（如检索分数或与查询相关度）结合启发式规则（如覆盖查询不同方面）来完成，也可以训练一个**证据评估模型**对每个候选片段的重要性打分。

实现选项：
- 检索模型： 使用开源的密集向量检索模型，如 **Sentence-BERT** 或近期表现优异的 **E5** 系列模型，将文档段落嵌入向量空间，实现高召回率的语义检索。亦可尝试**混合检索**（Hybrid Retrieval），即先用BM25获取候选再用向量模型重排序，以兼顾关键词匹配和语义匹配的优点。针对金融领域语言，可考虑对检索模型进行语域自适应微调（例如利用FinDER提供的查询-证据对进行fine-tune），以提升处理金融缩略语和行话的能力。
- 推理机制： 考虑两种路线：其一，**基于模板的规则推理**，预定义一些常见问题类型的分解模式（如同比类问题分两步检索不同年份数据再计算），系统按照模板检查并执行迭代检索。这种方法明确易控，但对无法覆盖的新问题类型可能无能为力。其二，**LLM Agent（代理）方法**，即使用一个大型语言模型作为智能代理，在检索过程中交替执行“思考”和“行动”指令——“思考”步骤由LLM生成下一步要查找的线索，“行动”步骤由系统根据该线索执行检索并将结果反馈给LLM，循环往复。这类似于链式思维（Chain-of-Thought）与Actuator结合的思路，能灵活适应多种推理需求，但需注意控制模型的错误引导和成本。
- 证据整合: - 如果使用LLM Agent，上述Agent在每轮检索后可保留一个**工作记忆**，逐步构建对问题的解答思路，最终汇总所有证据。这种方式天然地产生一个推理链，可用于解释模型的思路。 - 如果采用流水线方式，也可以引入一个简单的**逻辑回推模块**：例如在取得两个年份的数据后，用代码计算增长率，或在获取多个段落后用规则验证它们是否来自同一主题。这一模块既可以在检索阶段作为检查（如验证不同段落提及的是同一实体），也可以在下游生成中使用。

- **模块3：证据整合与计算模块**
  职责： 接收模块2输出的证据集合 E，对证据进行深入处理，包括必要的数学计算和跨证据推理，形成对查询的完整解答思路。该模块确保所有需要的推理都在答案生成前完成，以降低语言模型出错的风险。
  输入： 证据集合 E（经过筛选的若干段相关文本），以及查询解析结果（可能含结构化任务描述，如要求计算同比等）。
  输出： 一个结构化的**推理结论**或中间结果 R，其中包括：如果需要计算，给出计算过程和结果；如果需要比较，给出比较的逻辑关系；并将各部分结论和对应的证据段落建立映射关系。简而言之，R 是对答案内容的框架性说明，可被视为生成答案的蓝本。
  实现选项：
- 程序化计算： 对于数值类查询，优先使用代码计算而非让语言模型推断。例如编写简单的计算函数，从 E 中抽取相关数值字段，然后按照业务逻辑（差值、比率等公式）计算出结果。这可避免LLM的算术错误，并使结果更可解释和可重复验证。
- 逻辑推理: 对于需要比对文本证据的查询（如“公司A与公司B在某方面有何不同”），可以在 E 基础上用规则或简单NLP方法（如对比两个段落的关键信息）得出差异点摘要。
- LLM推理: 亦可利用大型模型本身进行证据综合：将 E 拼接成提示，要求模型先输出一个分步推理过程（理由链），再给出结论。这相当于让模型担任本模块功能。不过此方法需谨慎使用，以防模型产生不忠于证据的内容，通常会结合提示工程来督促模型引用 E 中的信息而非编造成果。

（模块2和模块3的界限在实现中可以视情况调整：有的方案可能将多步检索和部分推理糅合在一起，例如LLM Agent直接在对话中既检索又计算；也有方案将所有计算留到生成阶段。但为了清晰起见，这里将检索得到证

据、对证据进行计算推理与最终生成分别划归不同模块。模块3强调的是让确定性的推理先于文字生成，从而提高答案准确性和可靠性。）

- **模块4：答案生成模块**
  职责： 根据经过推理整合的结果 R，生成最终对用户查询的答案文本。该模块是与用户直接交互的部分，需要用专业且简洁的语言给出回答，并确保回答内容可溯源至提供的证据。对于金融领域，还需注意数值的表达格式和单位，措辞准确性等。
  输入： 中间推理结论 R，证据集合 E（可选，用于引用或佐证），以及原始查询（用于保持问答的上下文相关性和措辞风格）。
  输出： 最终答案 a（文本），通常是一两段话，直接回应用户问题。理想情况下，答案应包含必要的数据或事实，并在措辞中反映出基于证据推理而来的结论。如果是部署在交互系统中，答案还可以附带引用标记，指向相应证据出处。
  实现选项：
- 微调的生成模型： 选用开源的指令微调大型语言模型作为生成器，例如LLaMA-2系列、Bloom或ChatGLM等，并结合本任务的需要进行微调训练。微调数据可以来自FinDER（三元组中的查询-答案对，证据作为附加上下文）以及其他相似金融问答数据。这种专门微调能教会模型在答案中引用证据、执行金融领域特有的表述。由于项目时间有限，可优先微调中等规模模型（如7B或13B参数量级），在保证性能的同时降低训练难度。
- 提示范式（Prompting）： 如果微调资源有限，也可以直接使用现成的开源大模型，通过精心设计提示来引导生成。提示中包括：重述问题、提供证据要点 R 和 E、要求模型给出依据证据的回答等。需要确保提示明确要求模型不要引入未提供的知识，并严格依据 R 进行回答。
- 答案形式:
  - 对于**定量问题**，答案应给出具体数值和计算结果，并尽量说明计算过程或依据（例如：“根据2020年年报，公司X在2021年的净利润为Y亿元，较2020年的Z亿元增长了N%。”）。
  - 对于**定性问题**，答案应当概括证据内容并直接回应提问。例如问“公司A的主营业务是什么”，答案可表述为“公司A的主营业务包括...（信息源自其年报第X页）”。在更开放的生成场景，可以在答案中加上引述的证据片段或出处标记，以增加可信度。

流程概述： 综上，系统从接收到查询到输出答案的大致流程如下： 1) **解析查询：** 模块1将用户查询标准化，提取出关键金融实体和任务类型。
2) **检索证据：** 模块2进行首次检索获取初步文档段落；若答案涉及多个方面，模块2根据初步结果迭代检索补充其它证据。
3) **推理计算：** 模块3汇总所有证据，在需要时执行跨段推理和数学计算，形成答案要点和结论。
4) **生成回答：** 模块4据此撰写最终答案文本，确保内容准确、专业，并与查询对应。

整个框架强调**方法创新**：通过在第2步引入智能的多轮检索决策和在第3步引入显式计算逻辑，使RAG系统具备更强的复杂推理能力。这些模块均以开源工具为基础实现，可方便地共享和复现结果。

1. **数据与开源模型**
   数据集： 本项目将使用 FinDER 数据集作为主要评测基准。FinDER由真实的金融文件（如上市公司10-K年报等）及专家提出的问题组成，贴近实际分析师的工作场景。数据集中每条样本包含：(a) 用户查询，(b) 文本证据片段（通常摘自一个或多个文件），以及 (c) 对应的正确答案。语料库涵盖多家金融机构的公开披露文件，内容丰富多样，包括财务报表、管理层讨论、监管披露等。这为我们的RAG系统提供了真实的知识源。使用FinDER有两方面好处：一是数据集中问题本身经过了清洗和标注，能用于训练或微调模型（如检索模型和生成模型）；二是自带的证据-答案对使我们可以自动评估系统性能。此外，如果需要拓展训练数据，我们也可以考虑其它开源金融QA数据集（如 FinQA、TAT-QA 等涉及财务报告的问答），以及金融领域的文本资源（例如证券分析师报告、财经新闻）来增强模型的金融知识面。所有使用的数据都将是公开获取的，以确保方案的可复现性。

文档预处理： 由于金融文档往往篇幅很长（如年报可达上百页），我们将预先对语料库 D 进行分段索引处理。具体做法是将文档按段落或页面切分为独立的文本块，并为每个块附加元数据（如所属文档名、年份、章节标

题）。然后使用向量数据库（如FAISS或Milvus等开源方案）对这些文本块建立索引。这种预处理能加快在线检索速度，并允许我们方便地获取某段文本所在的上下文（例如在提供答案时引用来源）。

开源模型选择： - **检索模型：** 我们考虑使用现有开源的文本向量化模型作为起点，例如 SentenceTransformer提供的**msmarco-distilbert**系列，或近期专为信息检索训练的**E5-base** 等等。这些模型无需从零训练，可直接产生段落级别的向量用于语义检索。在项目初期，我们将评估哪种预训练向量模型对金融文本的嵌入效果较好。如果通用模型效果不理想，则会利用FinDER的训练集对其进行轻量微调（fine-tune）：以查询-相关文段为正样本，查询-无关文段为负样本，训练双塔编码器，提高模型对金融术语和上下文的敏感度。整个检索组件力求使用开放库实现，比如使用HuggingFace的**Transformers**加载模型，向量索引使用**FAISS**，以避免任何闭源依赖。 - **生成模型：** 在答案生成阶段，我们选用开源的指令遵循型LLM。候选包括**LLaMA-2 Chat**模型（Meta发布的LLaMA2变体，在Apache许可下开放）、**Bloom**（适合多语言的大模型）、或**ChatGLM**系列（开源的中文对话模型，若涉及中英文混合）。考虑到金融领域的专业性和英文资料的占比，优先选择在英文上表现好的模型并支持金融领域微调的。LLaMA-2（13B或7B参数）是一个平衡选项：参数规模适中且社区支持丰富，我们可以针对FinDER问答对其进一步微调，使其掌握金融问答的格式和证据引用习惯。如果运行资源有限，也可考虑更小巧的**Flan-T5**模型（例如Large版），虽然单次生成能力略逊，但经过任务微调后也能胜任有据可依的回答生成。此外，本项目严格避免使用需要付费的闭源API（如OpenAI的GPT-4、BloombergGPT等未开放模型），所有模型都将运行在本地环境，以确保方案完全开源可复现。训练和推理所使用的框架将基于开放源码库（如PyTorch、Transformers、LangChain等）。 - **其他工具：** 为支持多步推理，我们可能集成一些开源的工具组件：例如用于单位换算或财务指标计算的Python库，用于解析时间序列数据的简单脚本等。如果采用LLM Agent方案，可能用到 **LangChain** 或 **Haystack** 等开源框架来编排提示（Prompt）与检索动作。实体识别若需要预训练模型支持，可采用 **spaCy** 金融模型或 HuggingFace上的金融领域NER模型。这些工具的选择都以开源为前提，并会在报告中记录具体版本，确保他人能够安装相同环境运行。

数据使用策略： FinDER数据集将主要用作**评估**和**小规模微调**：由于其问题数量有限（数千量级），我们会把其中大部分用作测试验证系统效果。不过，我们也可以划分一部分作为开发集，用于检索模块及生成模块的调试。例如，挑选其中的1000个QA对用于微调检索模型，提高召回率；再用另一部分数据微调生成模型，使其学会在答案中引用证据。剩余未见过的问题则留作最终评估集。同时注意数据泄漏问题，确保用于评估的问题及答案不在训练微调过程中出现。这样的划分能保证模型既从数据集中学习到金融问答模式，又能在完全陌生的问题上测试泛化能力。

1. **实验计划**
   为在两个月内高质量完成研究，我们制定如下实验步骤和时间安排，兼顾模型开发和性能验证：
2. **步骤1：基线系统搭建（预研阶段，第1-2周）**
   我们首先实现一个基本的金融RAG原型系统，不含多步检索推理机制，以作为对照基线。检索部分可直接采用向量搜索+BM25的简单融合，生成部分使用开源模型的直接推理（不经特殊微调），仅根据检索到的前若干段落给出答案。这个基线系统将在FinDER开发集上进行初步测试，记录关键指标（如检索召回率、答案准确率）。从基线表现中，我们会分析现存主要错误类别：例如是否经常因为只检索一次导致遗漏信息、或因无法计算导致数值误差等。这些分析将为有针对性地设计多步检索方案提供依据。
3. **步骤2：多步检索模块开发（第3-5周）**
   在了解基线不足后，进入核心方法的实现阶段。首先开发**模块1查询理解**：建立金融缩写词典和基本NER模型，对一些典型缩写/简称做测试，确保能正确扩展查询。接着重点开发**模块2多步检索**：实现迭代检索逻辑。在代码上，可以利用现有框架（如Haystack的Pipeline）实现一个循环检索组件，实现伪代码：
   while 需要更多证据：检索→分析→生成新查询→再检索。
   在此过程中，我们将尝试两种实现进行对比实验：
   (a) **规则驱动迭代：** 针对有限的几类问题手动制定推理规则，验证多步检索在这些案例上的有效性；
   (b) **LLM Agent驱动：** 编写Prompt模板，让开源LLM按照ReAct框架执行检索和推理动作，并观察其表现。
   在开发和调试中，我们使用FinDER开发集的问题进行交互式测试。例如，选取一问需要两步检索的案例，运行模块2，检查它是否能成功获取所需的两个证据段。如果某轮检索检回无关结果，我们会调优检索查询生成策略（比如调整提示或增加上下文）。这个迭代开发过程也涉及**检索模型微调**：我们可能

利用开发集对向量检索模型训练若干epoch，以提升它对正确证据的得分。微调后再次验证迭代检索的效果。我们的目标是模块2对于开发集中的复杂查询都能找到完整证据链（主观判断标准是：人工核对证据集合确实包含足够信息回答问题）。

4. **步骤3：证据整合与计算模块开发（第5-6周）**
   完善**模块3**，使系统能对检索到的多段证据进行自动推理和计算。我们将实现简易的**计算引擎**：利用Python解析证据段落中的数字，按公式算出指标。例如，对于同比增长计算，编写函数：(今年值-去年值)/去年值。以开发集的定量问题为测试样例，验证该计算模块的正确性。并实现**证据去重与验证**逻辑：比如检查多段证据是否都涉及同一主题实体（防止串错公司或年份），如不一致则提示模块2重新检索或过滤掉不匹配证据。此阶段我们也决定证据传递给生成模块的格式：是简单拼接文本，还是结构化摘要。我们会尝试将R表示成一个包含关键结论的JSON或类似结构，然后由生成模型读取。这种结构化表示在少数样本上人工评估可读性和完整性，必要时迭代修改。
5. **步骤4：答案生成模块调优（第7周）**
   在拥有可靠证据链和推理结果后，我们微调或Prompt-tuning**模块4**的生成模型。具体做法是选取FinDER开发集的一部分（或结合其他QA数据）构造训练样本：输入由“问题+证据”组成，目标输出为参考答案。使用这些样本对选定的开源LLM进行微调训练，或者进行少量回合的反馈调教（如使用LoRA等高效微调手段，考虑项目时间和算力，这里不会训练全参数，只调整一部分权重）。如果无法进行微调，则设计多个Prompt模板，通过手工Prompt Engineering来提升输出质量——例如要求模型在答案中引用证据、先列出推理步骤再给结论等。在开发集中，我们评估生成模块输出的流畅程度和准确性：重点检查是否遗漏了推理步骤结果、是否有不依据证据的编造。针对发现的问题调整模型或提示。例如，如果模型倾向于编造成果，我们会在提示中加入更明确的要求如“不要输出未在证据中出现的内容”。该阶段也涉及确定答案的格式风格，使其专业且简洁。
6. **步骤5：综合测试与迭代优化（第8周）**
   在项目最后两周，我们将在完整系统上进行综合评测和优化。首先使用FinDER验证集/测试集运行新系统，收集各项指标（见下文评估指标）。比较这些指标与基线系统的差异，重点关注复杂查询上的改进幅度。同时，分析错误案例，将其分类：是由于检索遗漏（可能提示我们需要增加迭代次数或改进模型召回），还是由于推理逻辑错误（可能需要加强模块3的验证规则），抑或生成阶段的问题（例如语言表述不准确）。根据分析结果进行有针对性的**迭代优化**：
7. 若检索召回仍有不足，可能调整检索模型参数或增加组合检索（比如增加BM25候选）；
8. 若推理计算有偏差，完善对应规则或增加对LLM输出的检查（如验证计算结果合理性）；
9. 若生成答案有瑕疵，进一步调整微调数据或提示。
   每次修改后再跑一次评测集验证改进，直到项目结束前，在主要指标上达到预期提升目标（例如答案准确率提高若干百分点，特别是在需要多跳推理的问题上性能显著优于基线）。最后，我们将整理所有模块的配置和参数，并在报告中记录实验设置，以方便他人重现结果。

整个实验计划注重**增量开发**和**对比实验**：先有基线，再逐步引入新机制，每引入一部分都验证其效果。这确保了在有限的时间内，我们能清楚地知道哪些改进带来了性能提升，哪些可能无效，从而将精力集中在最有价值的方案上。

1. **评估指标**
   为全面评估本系统在FinDER复杂金融查询上的表现，我们将采用多层次的指标体系，包括检索阶段、推理准确性和最终答案质量三个方面：

2. **检索性能指标：**
   我们用**召回率** (Recall@K) 和 **准确率** (Precision@K) 来度量检索模块在前K个候选结果中涵盖正确证据的能力。其中，Recall@K指在每个查询中，正确答案所在的相关文档片段是否出现在前K个检索结果中，然后对所有查询计算出现比例；Precision@K则指返回的K个结果中有多少比例是真正与查询相关的（即属于人工标注的证据）。例如，我们重点关注 Recall@5 或 Recall@10，期望多步检索策略能提高这些值（尤其对复杂查询，Recall@10应该显著高于基线的一次检索Recall）。同时，我们会计算**平均检索排名** (Mean Reciprocal Rank, MRR)：对于每个问题，找到第一个相关证据在检索结果中的排名，取倒数，然后平均。这反映用户获得有效信息的快慢。高性能的检索模块应在排名靠前的位置就提供相关段落，从而取得较高的MRR值。

3. **推理与计算正确性指标：**
   针对涉及数值计算或多段整合的查询，我们将引入定制的正确性检查。例如，对需要计算同比增长的问题，评估**计算准确率**：即系统计算出的结果与标准答案的差异（可用**相对误差**或**绝对误差**衡量）。若答案要求从多段证据推理得到，我们可以检查系统是否找全了必需的证据（这在FinDER数据中可通过验证系统使用的证据是否覆盖了标注的所有证据）。定义一个**证据覆盖率**指标：系统返回的证据集合与标准证据集合的重合比例（以召回/精确率形式）。理想情况下，多步检索能使证据覆盖率接近100%，即系统找到了答案所需的全部证据。另一个角度是**推理链准确度**：如果我们让系统输出显式的推理步骤（例如Chain-of-Thought文本），可以人工或规则校验这些中间推理是否合逻辑、每步是否正确利用了证据。这些指标主要用于分析系统的推理过程是否可靠，不一定都量化为一个分数，但通过抽样检查，我们可以确保系统在推理环节未引入明显错误。

4. **答案质量指标：**
   最终输出答案将通过**准确性**和**可读性**两方面评估。准确性方面，使用FinDER提供的参考答案，我们采用**精确匹配率** (Exact Match) 和 **松散匹配度** 来衡量：前者指系统答案与标准答案文本完全一致的比例，后者考虑语义上的正确（可以用**F1得分**或文本相似度衡量，计算答案与参考答案在关键字上的precision和recall）。在定量问答上，准确性也体现为数值是否正确（对数值答案我们会直接比较差异）。对于定性描述类答案，我们引入**ROUGE-L**或**BLEU**等自动指标，评测系统答案覆盖了参考答案信息的程度。例如ROUGE-L着重最长公共子序列，可以反映答案包含了多少参考答案中的要点。此外，为了评估模型在真实使用中的表现，我们可能进行**人工评价**：邀请数名对金融背景熟悉的同事，对模型答案按照正确性、解释充分性、语言专业性打分。人工评价能补充自动指标无法捕捉的细节，如答案是否措辞专业、有无不当臆测。

此外，我们特别关注**复杂查询子集**的表现。因此，会将测试集中标记为需要多步推理的问题单独计算指标。例如比较我们的系统和基线系统在这些问题上的Exact Match准确率差异。如果实现了预期效果，我们应当看到本系统对多步问题的准确率明显提升，而在简单直答型问题上不低于基线。在报告中，我们会列举一些典型案例对比——包括系统给出的推理过程和答案，与基线的结果对照分析。这将有助于读者直观了解多步检索推理机制带来的改进。

- **效率指标（次要）：**虽然准确性是主要目标，但我们也记录系统的响应效率，如平均每个查询耗时，检索调用次数等。由于多步检索必然增加一定时间成本，我们希望通过优化减少不必要的检索轮数。在2个月的开发中，我们不会专门针对效率进行优化，但会确保在合理范围内（例如一次完整问答在数秒至几十秒内完成，以满足互动需求）。如果发现多轮迭代过多导致延迟，我们可能引入**检索步数上限**或**并行检索**等策略作为权衡。在最终报告中，可提供系统复杂度分析，说明多步机制增加的计算开销和可扩展性。

小结： 通过上述指标体系，我们将能够全面评估本研究方案的有效性。成功的标志是在FinDER数据集上，本系统对于复杂金融查询的回答准确率超过基线方法，同时检索证据充分且推理链合理。定量指标的提升（如Exact Match提高、Recall@K增加）将证明方法创新的价值，而定性分析（如案例展示推理链）将进一步说明系统如何更好地应对现实的金融问答需求。所有评测过程中产生的数据和代码也将随项目开源发布，以便社区复现和检验我们的结果。
