Enhance retrieval performance on mixed type documents through LLMs in RAG systems

Alice Turrini and **Alessio Conti**
Master's Degree in Artificial Intelligence, University of Bologna
alice.turrini@studio.unibo.it, alessio.conti3@studio.unibo.it

Abstract

The growing presence of mixed-type documents, combining unstructured text and structured tabular data, poses challenges for modern information retrieval systems. While embedding models excel at encoding natural language, they often struggle to capture the semantic and structural complexity of tables. This project enhances Retrieval-Augmented Generation (RAG) by leveraging Large Language Models (LLMs) to generate semantic descriptions of tabular content. The effectiveness of this approach is evaluated by comparing standard embedding techniques with this enhanced method. Extensive experimentation with various prompting strategies demonstrates significant improvements in retrieval performance for financial documents. Results indicate a notable enhancement in preserving and utilizing crucial tabular information, leading to more accurate and contextually relevant retrievals leveraging increases of up to 15% in performance. This work helps bridge the gap between structured and unstructured data representation in modern information retrieval.

1 Introduction

Information extraction from documents containing mixed data types, particularly those combining plain text and tables, remains a significant challenge in modern information retrieval systems. Although Large Language Models (LLMs) have demonstrated remarkable capabilities in processing natural language, their effectiveness in handling tabular data within the context of Retrieval-Augmented Generation (RAG) systems is still not fully explored. Tables, being structured representations of information, often contain crucial data that may be lost or misinterpreted when processed using standard text embedding techniques. Traditional approaches to this problem typically involve treating tables as plain text. This often fails to capture structural relationships and hierarchical information inherent in tabular data. This method often fails to capture the semantic context that humans naturally infer when interpreting tables alongside text, such as the high-level meaning of the table and the potential trends within the data. Our approach addresses these limitations by leveraging LLMs to generate natural language descriptions of tabular data, which are then combined with standard text embeddings. The intuition behind this method is that LLMs can act as intelligent interpreters, extracting and articulating the semantic content of tables in a way that preserves both structural relationships and contextual meaning. By prompting LLMs to generate these descriptions through carefully crafted prompts, we aim to create richer and more contextualized embeddings. Further tests were conducted to exploit LLM's capabilities in query expansion and rephrasing to improve search systems' performances.

2 Background

Retrieval-augmented generation systems have emerged to enhance LLM responses with external knowledge, address the limitations of static training data, and reduce hallucinations. These systems rely heavily on the effectiveness of their retrieval component, which traditionally employs dense vector embeddings to identify and retrieve relevant context from a knowledge base. The retrieval process typically involves encoding queries and documents into the same vector space, where semantic similarity can be measured. Vector stores, also known as vector databases, are specialized storage systems designed to efficiently index, persist and retrieve high-dimensional vector embeddings. These systems implement approximate nearest-neighbour (ANN) search algorithms to enable efficient similarity search operations in high-dimensional spaces. These systems typically operate in two phases: an initial indexing phase, in which incoming documents are processed and organized, and a subsequent query phase, in which similarity searches be-

tween the queries and the documents in the vector store are performed using distance metrics. Modern vector stores often incorporate additional features such as metadata filtering, allowing for hybrid searches that combine semantic similarity with traditional database queries. They also offer support for multiple vector indices to accommodate different embedding models or data modalities. In this project, we utilized FAISS (Douze et al., 2024), a highly efficient library designed for indexing and similarity search through cosine similarity. FAISS enables fast and scalable retrieval by leveraging optimized indexing structures, making it an ideal choice for managing and querying large-scale embedding-based datasets.

3 System description

In this study, we propose three methodologies designed to investigate how document embeddings and retrieval can be enhanced by the use of LLMs. The techniques investigate the scenario of mixed format documents (text and tabular) with the objective of increasing the accurate recall rate in retrieval tasks.

(I) The first methodology, **No-Summary (NS)**, is a straightforward approach in which tabular data undergo no preprocessing. The textual content is directly embedded using a sentence transformer without any modifications or additional processing. This is intended as the project baseline. Each document is divided into chunks of 256 tokens with an overlap of 15% among the chunks to provide context. The embedding model used is *all-MiniLM-L6-v2*, a sentence transformer able to capture the semantic information of text into a dense 384-dimensional vector space. The computed embeddings are then stored using Faiss for the creation of the vector storage database.

(II) The second methodology is inspired by LangChain's proposals to increase RAG performances (LangChain, 2023), and it consists of using LLM for the creation of **table summaries**. This approach introduces an initial extraction phase, wherein tables are isolated from the text using regular expression (regex) techniques. The inherent complexity of tabular data is characterized by its structural diversity: including rows, columns, multi-level headers and predominantly numerical content. All these complications, as observed in the documents analyzed, pose significant challenges to comprehension. To mitigate these challenges, the extracted tables are processed using an LLM, which distils the key information into a "summary" through prompting techniques. As LLM, after several explorations, was chosen *Mistral 7B Instruct v0.3* (Mistral, 2023) that seemed promising despite its modest size compared to today's state-of-the-art LLM. The model is asked to understand the concept behind the table and report it in a textual form. This approach encompasses two experimental variants:

- the Short Summary (SS)
- the Long Summary (LS)

which differ in the extent of detail and emphasis elicited through the prompt. In SS, the LLM is asked to provide a summary of the content present in the given table, as well as a title and some keywords and tags that best fit the table content. In LS, similarly, the LLM is asked to provide a summary, but this time fairly longer, then it is also asked to expand any acronym or abbreviation present. Examples of the summary in Table 1 and the detailed prompts can be found in the appendix in Table 8. The generated summaries are embedded with the same model as additional chunks, maintaining references to the related document for retrieval purposes. Notably,

Table:	in millions | 2007 | 2006 | 2005 ——- | —— | —— | —— sales | $ 5245 | $ 4925 | $ 4625 operating profit | $ 501 | $ 399 | $ 219
SS	Financial Performance Over Three Years. This table shows the financial performance of a company over three years. Key trends include a steady increase in sales, a significant increase in operating profit, and a substantial increase in net profit. **Keywords**: sales, operating profit, net profit, financial performance, trends, increase
LS:	Financial Performance Over Three Years. The table displays the financial performance of a company over a three-year period, from 2005 to 2007. The key financial metrics include sales, operating profit, and net profit, all presented in millions. The table shows a steady increase in sales, operating profit, and net profit over the three-year period. **Keyword**: sales, operating profit, net profit, financial performance, three-year period

Table 1: Example of a table summarization over the two approaches: short summary and long summary.

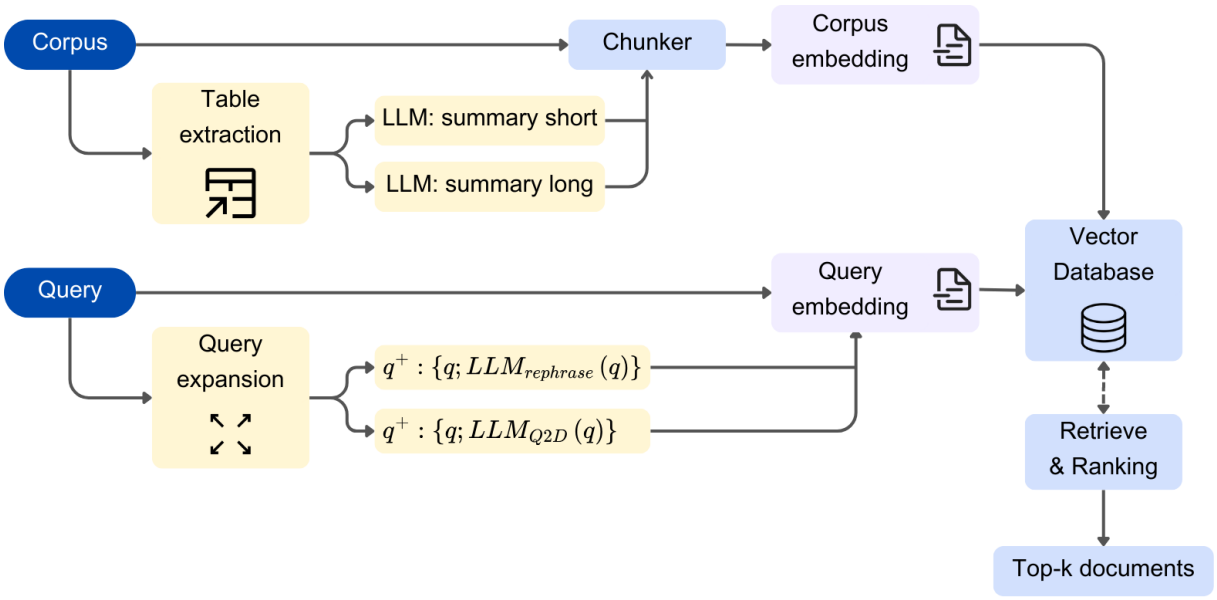

Figure 1: Schematic representation of our complete information retrieval system. There is the core baseline architecture, in blue, and the additional approaches integrated to enhance performance shown in yellow. These approaches to improve retrieval effectiveness are Table summarization and Query expansion.

also the tables within the text were retained, as their removal was observed to negatively impact performance. This is probably due to the sentence transformers' reliance on the surrounding context, enabling better semantic understanding.

(III) The third methodology is **query expansion**: introduced to improve the retrieval accuracy of the system. It is built on top of the previous best approach, which resulted in the LS, by incorporating two different expansion techniques to the query. These methods are introduced to submit as much context as possible to the embedder model, with the aim of having a better chance to match the target documents. Two specific techniques were evaluated within this framework:

- the Query Rephrasing (QR)
- the Query-to-Document (Q2D)

In particular, QR consists of rephrasing the query using synonyms and different syntactical structures from the original one. As for the tables, the model is asked to expand any acronym or abbreviation present. Q2D technique, instead of rephrasing the original query, tries to write a pseudo-document as an answer to the query itself. Examples of the query expanded can be found in Table 2, and the detailed prompts can be found in the appendix in Table 7. For both approaches, the expanded query is then concatenated to the original one before undergoing the embedding process. These strategies aim to enhance information retrieval with additional insights, details and richer context information given by the LLM.

Query	In which year was the share of net earnings in Golar Partners higher?
QR	What was the year in which the net earnings of Golar Partners, a company that specializes in the ownership, operation, and management of floating liquefied natural gas (FLNG) vessels, had a higher share compared to other years? Specifically, I am looking for the year in which the net earnings as a percentage of total revenue or total assets was greater than in other years.
Q2D	The share of net earnings in Golar Partners may have been higher in different years compared to others, but without specific data, it is not possible to definitively state which year had a higher share of net earnings. To find this information, one would need to analyze the financial statements of Golar Partners for various years.

Table 2: Example of query expansion with the two tested approaches: query rephrase and query to document.

4 Data

Dataset	Num of Doc	Num of queries	% Tables in Text	Tables per Doc
ConvFinQA	101	126	15.7%	1
FinQA	247	344	16.2%	1
MultiHeritt	876	292	51.7%	0.77
TATQA	248	498	36.1%	1

Table 3: The table reports some dataset statistics: the number of documents, queries, percentage of text containing tables and the number of tables per document for each dataset.

The experiment is tested on four datasets (Linq, 2024), each designed to test different aspects of tabular and text retrieval in the financial context. The datasets are: TATQA, FinQA, ConvFinQA, and MultiHeritt. Their complexity arises from the need for numerical reasoning, multi-step calculations, and multi-hop processing over hybrid data that combine tables with unstructured text.
These are the datasets in more detail:

- *ConvFinQA*: involves handling conversational queries based on earnings reports.
- *FinQA*: demands answering complex natural queries over earnings reports using multi-step numerical reasoning.
- *MultiHiertt*: focuses on multi-hop queries, requiring retrieving and reasoning over hierarchical tables and unstructured text from annual reports.
- *TATQA*: includes natural queries that involve especially numerical reasoning over hybrid data of tables and text.

An example of an instance from *ConvFinQA*:

Query:	what was the average revenue from **discontinued operations in 2013**?
Relevant doc:	... **discontinued operations** as of december 31, **2013**, blockbuster had ceased all material operations. ... current assets from discontinued operations | $ 68239 noncurrent assets from discontinued operations | 9965 ...

Table 4: Example of the passage of a relevant document given a query for the dataset ConvFinQA

The MultiHeritt dataset is the biggest in terms of corpus and document lengths. It is also the most complex dataset involving multi-hop steps across various document sections. Half of the documents contain only one table, and 60 have more than one table per document. The substantial portion of text-only documents in this dataset limits the effectiveness of this project's methods. Indeed, the approaches implemented for the tables show low improvements.
The datasets are annotated for the retrieval step, with each query assigned to a set of relevant documents. It is important to highlight that all queries within the datasets TATQA, FinQA, and ConvFinQA are associated with a single relevant document; instead MultiHeritt contains multi-hop queries and indeed multiple relevant documents. These queries require reasoning across multiple document sections, making them inherently more complex and involving multiple relevant documents to answer.

5 Experimental setup and results

Various models were evaluated to identify the most suitable option for generating the document's embeddings. Initially, *FinBERT* seemed an ideal choice due to its fine-tuning in financial language; however, it demonstrated limitations in understanding sentence structures and capturing complex semantic relationships. Sentence transformer architectures including *all-MiniLM-L6-v2*, *all-mpnet-base-v2*, and *bge-base-en-v1.5*, were then explored. Sentence transformers are particularly well-suited for this task, given their ability to generate context-aware embeddings that capture complex semantic information across sentences. Among the tested models, **all-MiniLM-L6-v2** outperformed the others due to its efficient architecture. Its lightweight design allowed it to be the optimal balance between accuracy and speed, resulting in superior performance in both embedding quality and processing time.
For what concerns the usage of LLMs, various models have been evaluated, prioritizing speed, performance, and GPU efficiency. We had to face limited GPU availability: all models have been tested in a Google Colab environment on a machine mounting an Nvidia Tesla T4 with 15GB of memory GPU. For the table summarization task have been tested the following models: *GPT-2* and *Deepseek R1 Distill Qwen 1.5B* were faster in computation, but they often generated nonsensical outputs. In contrast, *Qwen2 7B* produced more structured summaries but at the cost of slower processing. *Llama 3.2 Instruct* models, both 1B and 3B, performed well in terms of output structure but lacked depth in table understanding. The final choice, **Mistral 7B In-**

Approaches	ConvFinQA			FinQA			MultiHeritt			TATQA		
	NDCG	Recall	MRR	NDCG	Recall	MRR	NDCG	Recall	MRR	NDCG	Recall	MRR
No summary	0.6482	0.8413	0.5874	0.5650	0.7587	0.5046	0.3808	0.1602	0.0908	0.5106	0.6928	0.4537
Short summary	0.7198	0.8968	0.6631	**0.6418**	**0.8140**	**0.5876**	0.4109	0.1677	0.0966	0.5391	0.7329	0.4782
Long summary	0.7348	**0.9048**	0.6801	0.6324	**0.8140**	0.5752	0.4234	**0.1748**	0.1017	**0.5489**	**0.7369**	**0.4906**
Query rephrase	**0.7554**	0.8810	**0.7134**	0.6088	0.7936	0.5509	**0.4259**	0.1696	**0.1030**	0.5302	0.7149	0.4731
Query Q2D	0.7412	0.8730	0.6982	0.5989	0.7762	0.5437	0.4142	0.1733	0.0996	0.5099	0.7048	0.4494

Table 5: Performance comparison across different datasets. All metrics are computed @10.

struct v0.3, it strucks the best balance, delivering coherent and relevant table summaries in around 4 seconds, making it the most suitable model for this task considering both performance and the GPU limitations. Zero-shot prompting was used as one-shot, and few-shot promptings were tried, but they resulted in adding confusion to the model and slowed down the timing due to the much higher quantity of tokens in input. See table 8 for more details.

For the query expansion task, we compared the best model used for the tables summarization task, Mistral 7B Instruct, and *Flan-T5* guided by the article (Rolf Jagerman, 2023). Different sizes of *Flan-T5* were tested, but the model struggled to generate clear responses, often repeating the query instead of expanding it with different words. In contrast, Mistral 7B Instruct once again emerged as the best choice, producing more coherent and contextually accurate query expansions. In this case, we used a one-shot prompting as we discovered that having an example of output to follow guided the model to better responses. Look at table 7 for more details about the prompt.

	ConvF	Fin	MultiH	TAT	AVG
LS	24.4s	22.7s	30.5s	27.4s	26.2s
SS	16.8s	17.1s	24.4s	18.5s	19.2s
QR	14.1s	13.8s	14.4s	13.5s	13.9s
Q2D	13.2s	15.2s	16.3s	14.2s	14.7s

Table 6: Average LLM processing time for one document, in seconds

An analysis of the execution times of the LLMs revealed that summarizing the textual table was the most time-consuming task. Specifically, the long summary task (LS) required an average duration exceeding 26 seconds per table. Summarization in the MultiHiertt dataset was the slowest task, taking approximately 30% longer than other datasets for both SS and LS. This delay is attributed to the greater complexity and larger size of the tables in MultiHiertt. In contrast, both query expansion tasks were fast, taking an average of under 15 seconds per query. This discrepancy is likely due to the nature of the input processed by the model: queries consisted of short phrases, whereas summaries involved unstructured and complex texts.

To evaluate the performance of the models, standard retrieval metrics were employed, including NDCG@k, Recall@k, and MRR@k, with k=5 and k=10. Those metrics are used to evaluate both matching rates and satisfy relevance ordering.

The most comprehensive one is the Normalized Discounted Cumulative Gain (NDCG), which measures the quality of the ranking by considering both the relevance and position of the retrieved documents (with higher relevance scores discounted logarithmically as the rank increases). It is defined as:

$$\text{NDCG@k} = \frac{DCG@k}{IDCG@k}$$

$$\text{where} \quad DCG@k = \sum_{i=1}^{k} \frac{rel_i}{\log_2(i+1)}$$

and $IDCG@k$ represents the ideal DCG for perfect ranking. Recall@k quantifies how many relevant documents are retrieved within the top k results for a query, providing the measure of the model's ability to retrieve the correct documents. It is computed as:

$$\text{Recall@k} = \frac{\text{retrieved relevant doc in the top } k}{\text{total relevant doc}}$$

In the case of FinQA, ConvFinQA and TATQA, where the correct document is only one per query, recall can also be easily interpreted as the exact number of times the model retrieves the correct document. Recall does not consider the relevant document's position in the ranking; it just focuses on coverage in the top k. To also consider this aspect, we used the Mean Reciprocal Rank (MRR), which evaluates the position where each relevant

document is found, measuring retrieval ranking effectiveness. For a given query, it is indeed defined as:

$$\text{MRR@k} = \frac{1}{\text{rank}}$$

where rank is the rank position of the first relevant document.

These metrics collectively offer a comprehensive assessment of both ranking quality and retrieval effectiveness.

Table 5 reports all the metrics for each approach across the different datasets.

6 Discussion

The evaluation results across different datasets provide interesting insights into the effectiveness of the various approaches implemented. LLM-generated summaries revealed significant improvements in enhancing retrieval performance: the addition of the table summary led to a performance improvement of around 15%, as is shown in Figure 2. Among the two summary approaches, the long summary is the best performer. This is likely because the knowledge inferred from the table by the LLM helps contextualize the textual content, enabling the model to explain from the simplest acronyms to more complex meanings in the table. As a result, the RAG system retrieves more relevant documents and ranks them higher. In the case of *FinQA*, the results of LS are slightly under the SS's one. This is because the tables of this dataset are shorter and simpler. Thus, LS tends to be too verbose, and this could result in noise.

In general, recall metrics are relatively strong for most datasets, suggesting that relevant documents are often retrieved. The MRR is also high, meaning that these relevant documents are also retrieved in a high-ranking position.

The ConFinQA and FinQA show the strongest performances. On the other hand, for the MultiHeritt dataset, performances are notably lower, reflecting the complexity of multi-hop queries. The recall is low also because the metric is penalized when it returns an incorrect number of documents; this leads us to the conclusion that probably only a few documents out of the total relevant are returned. Again, the MRR in MultiHeritt further highlights the issue of frequent misranking of relevant documents. A low-ranking position for the first relevant document indicates that incorrect documents are often placed wrongly higher in the retrieval order. In general, both table approaches (SS, LS) helped the retrieval system on this dataset, but it needs to be considered that half of the corpus of this dataset (around 350 documents) does not contain any tables, limiting the effects of our approaches.

Regarding the query expansion approaches, both QR and Q2D are built on top of the best-resulting table settings, the long summary. The query rephrase outperforms Q2D across all datasets. This indicates that simple rephrasing helps the model capture semantic variations without introducing unnecessary noise, while Q2D struggles due to its tendency to generate irrelevant content, thus diluting the original query intent. Probably this is because the LLM used does not have a wide background knowledge, specifically for this finance domain, considering also its small size of 7B parameters.

Although query expansion techniques do not excel overall, a real improvement can be seen in *ConvFinQA*. In this dataset the query structure is simple and informal, thus benefits from rephrasing. In all the other cases, no improvements are registered. This can be attributed to query formulation, which exactly matches the document corpus words, as shown in table 4. As expected, it can be deduced that rephrasing highly specific queries introduces only additional noise.

These results show only the metrics computed over the top 10 documents retrieved. Additionally, we analyzed a narrower window of the top 5 documents to gain deeper insights into the models' performance. Figure 2 shows the values on top of each column which refer to NDCG@10, but in a slightly vibrant tone is reported the NDCG@5, and they are almost identical, underlying the quality of the retrieval. This suggests that the most relevant documents were retrieved within the top 5 results. These observations reinforce the model's effectiveness in ranking relevant documents early in the retrieval process.

7 Conclusion

This study explores three methodologies for embedding documents containing both textual and tabular data, addressing the challenges posed by the integration of these formats in information retrieval tasks. The first approach (NS) serves as a baseline for comparison. The second methodology enhances comprehension by extracting and summarizing tables using a Large Language Model (LLM). The experiments show superior performance increases by introducing detailed long summaries

Figure 2: Plot of the different approaches tested on tables: NS, SS, and LS. The values reported on top of each column refer to NDCG@10. In a slightly vibrant tone, the NDCG@5 is reported. It can be seen they are almost identical, underlying the quality of the retrieval.

Figure 3: The plot shows a performance comparison of the different approaches, NS, SS, LS, QR, and Q2D, across the four datasets. The evaluation metric reported here is the NDCG@10.

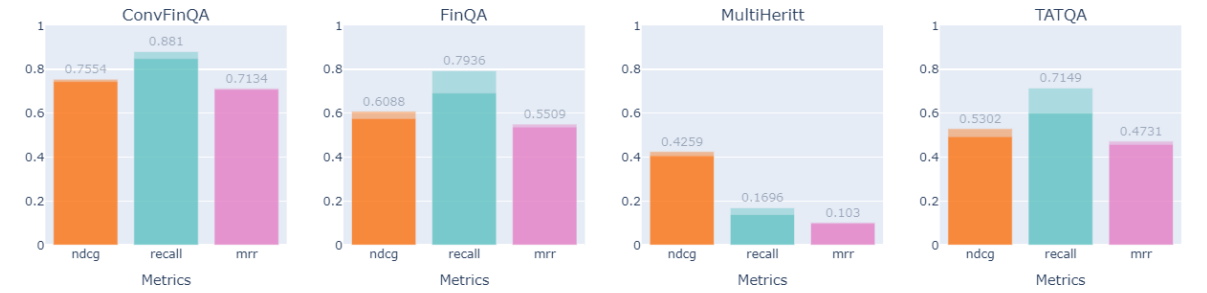

Figure 4: The plot focuses on QE approach (query expansion technique and table's long summaries combination) across all datasets. Each column in the histogram shows different metrics. Colours emphasise the same approach over different datasets, thus comparing performances. The values reported on top of each column refer to NDCG@10, and in a slightly vibrant tone, the NDCG@5 is reported.

(LS), especially effective in extracting complex information. The third approach further builds on this by incorporating query expansion techniques: query rephrasing (QR) and Query to Document (Q2D). These methods do not show any improvements because of the high specificity of queries, and expanding them results in just adding useless verbosity. Our findings highlight the importance of preserving tabular data within the text. As its removal was associated with performance degradation, adding summarized tables instead proved to be an effective strategy among all the tested datasets. Future research could focus on refining these methodologies by exploring alternative summarization techniques exploiting bigger LLMs, as well as assessing methodologies' effectiveness across different domains. Additionally, further experiments could evaluate the scalability of these approaches in large-scale document retrieval systems and the correlated cost-benefit ratio. Overall, our study contributes to the ongoing efforts in document embedding by providing insights into the

integration of textual and tabular data for improved information retrieval.

8 Links to External Resources

- GitHub repository: https://github.com/alessiocnt/FinanceRag
- Dataset: https://huggingface.co/datasets/Linq-AI-Research/FinanceRAG

References

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library.

LangChain. 2023. Multi-vector retriever for rag on tables, text, and images. https://blog.langchain.dev/semi-structured-multi-modal-rag/. Accessed: 2024-12.

Linq. 2024. Linq-ai financerag. https://huggingface.co/datasets/Linq-AI-Research/FinanceRAG. Accessed: 2024-12.

Mistral. 2023. Mistral 7b instruct v0.3. https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3. Accessed: 2025-01.

Zhen Qin Xuanhui Wang Michael Bendersky Rolf Jagerman, Honglei Zhuang. 2023. Query expansion by prompting large language models. *arXiv preprint arXiv:2305.03653*.

Appendix: prompts

Here are the reported prompts used in this project for the two tasks: the creation of a "summary" of a given table and the query expansion. In each prompt, the {text in red} denotes the element (table or question) that the LLM has to work on performing summarization (tables) or expansion (queries). With {green text}, we referred to the example given to the LLM in the case of one-shot prompting.

ID	Prompt
QR	You are a finance expert, with the ability to rewrite the queries. Rephrase a query so that it contains more information, use synonyms and your knowledge to understand the possible context. Read carefully the query, understand the concept that is asked and explain it. If acronyms are used, expand and explain their meaning. This is an example query: {query} And this is the corresponding query expanded: {query expanded} Return the query expanded in a dictionary format: {"query":"", "answer":""} Your input query: {query}
Q2D	You are a fincance expert. Write a passage that answers the given query, if you find acronyms, expand them and explain them. If there are specific finance concepts in the query, try to explain them briefly. Don't be too verbose, just one paragraph. This is an example query: {query} This is an example query: {query expanded} Return the query paragraph in a dictionary format: {"query":"", "answer":""} Your input query:{query}

Table 7: Query expansion prompts for the two approaches used: Rephrase, Q2D.

ID	Prompt
SS	You are a finance expert specialized in analyzing tables. ### Task ### Summarize the table in a dictionary with the following keys: - '**title**': A concise title for the table. - '**description**': A brief description of the table's contents and key trends. - '**keywords**': A list of relevant keywords. **Note**: Do not return the table, prompt, or any extra text. Return only the dictionary. ### Table ### {query}
LS	You are a finance expert specialized in analyzing tables. ### Task ### Analyze the table and transcribe its contents into a well-structured summary. Generate a detailed paragraph that captures all relevant information, including numerical data, trends, and observations. If the table contains acronyms, provide their concise definitions where possible. Return the output in the form of a dictionary with the following keys: - '**title**': A concise title for the table. - '**description**': A brief description of the table's contents and key trends. - '**keywords**': A list of relevant keywords. **Note**: Do not return the table, prompt, or any extra text. Return only the dictionary. ### Table ### {query}

Table 8: Table summaries prompts for the two approaches used: SS, LS.