Application of RAG to Institutional Documents of the Federal Institute of São Paulo (IFSP)

Ramon Abílio and Vagner I. de Oliveira

28 June 2024

Abstract

The Federal Institute of São Paulo (IFSP) is specialized in Professional and Technological Education, managing an extensive array of regulatory documents dispersed across its 37 campuses. These documents, stored in PDF format on institutional or Google Drives, pose significant retrieval challenges due to their varied of information, structures and formats over time. This project aims to develop a Retrieval-Augmented Generation (RAG) system to enable efficient extraction of reliable information from these documents, without requiring users to be familiar with the IFSP's organizational structure or document types. The methodology involves three phases: dataset composition, development of the RAG mechanisms, end evaluation.

1 Introduction

The Federal Institute of São Paulo (IFSP)[5] is a prominent educational institution in Brazil, specialized in Professional and Technological Education. It holds a unique position within the educational landscape due to its administrative, patrimonial, financial, didactic-pedagogical, and disciplinary autonomy. To maintain and regulate its operations, IFSP disseminates a wide array of documents, including Bylaw, Normative Instructions, Ordinances, and Notices [9]. These documents are crucial for ensuring compliance, consistency, and transparency across its campuses.

However, the current storage and retrieval system for these documents presents several challenges. The documents are dispersed across multiple locations, often stored in PDF format on institutional or Google Drives. This dispersion complicates the retrieval process, as users must navigate through various sources and formats to find the necessary information. Furthermore, the documents have evolved over time, with changes in their structure and format, including shifts from plain text to tables and the inclusion of scanned documents. These variations further hinder efficient information retrieval.

To address these issues, this project proposes the implementation of a Retrieval-Augmented Generation (RAG) system tailored to the specific needs of IFSP. The goal is to enable users to extract reliable information from these documents without needing to understand the intricate organizational structure of the Institute or the various document types. A RAG method enhances Large Language Models (LLMs) by addressing their limitations in domain-specific or knowledge-intensive tasks and mitigating issues like "hallucinations" when queries extend beyond their training data. It can synthesize contextually relevant, accurate, and up-to-date information by retrieving relevant document chunks from an external knowledge base through semantic similarity calculations[1][3].

2 Methodology

The development of the Retrieval-Augmented Generation (RAG) system for the Federal Institute of São Paulo (IFSP) involved a structured approach divided into three key phases (Fig. 1): a) Phase I: dataset composition and Question and Answer (Q&A) dataset development; b) Phase II: implementation of the baseline using BM25 and Sentence Window Segmentation, and implementation and evaluation of different approaches for segmentation and retrieval; and c) Phase III: evaluation of the two RAG systems. Use utilized Google Colaboratory (Colab) in free and paid versions (due to the GPU T4 requirement), GitHub to host the project files[1], Groq platform [4] to have access to Llama3 70B large language model (LLM), and Python v3.8, the framework LlamaIndex v0.10.50 [7] as our main tools. This section details those methodology phases in the following subsections.

[1] https://github.com/rsabilio/ia024-projeto-rag

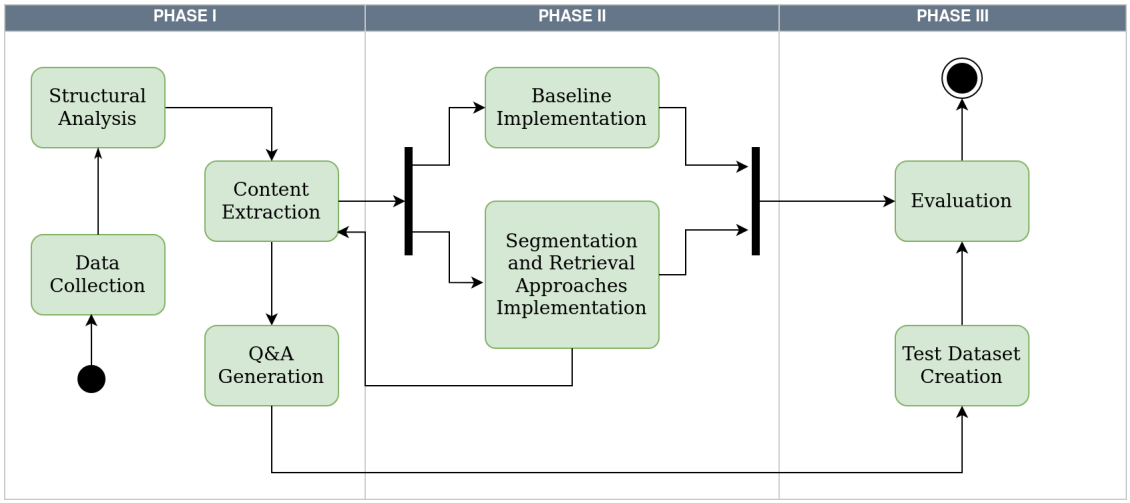

Figure 1: Methodology overview, including the three phases.

2.1 Phase I - Dataset Composition

The documents were sourced from the Rectorate's website (institutional drive) and the Boituva campus' website (Google Drive). Efforts were made to ensure that the dataset is comprehensive and representative of the various types of documents used across IFSP. We downloaded the Bylaw and the two Academic Organizations from the Rectorate's website and 1,180 ordinances from the Boituva campus's website[6], totaling 1,183 PDF files. We downloaded all ordinances available until May 2024, resulting in a range of nearly ten years (2014 to 2024). The ordinances were saved according to the year and month of publication, and we navigated through the directories, renaming them to maintain a consistent standard.

Subsequently, we conducted an in-depth analysis of the collected documents to understand their structure and content. We observed that, in general, the bylaws and academic organization documents have a hierarchical structure composed of Titles, Chapters, Sections, Subsections, and Articles (Fig. 2). Ordinances typically consist of a header (pre-articles), body (articles), and footer (post-articles), as illustrated in Fig. 2.

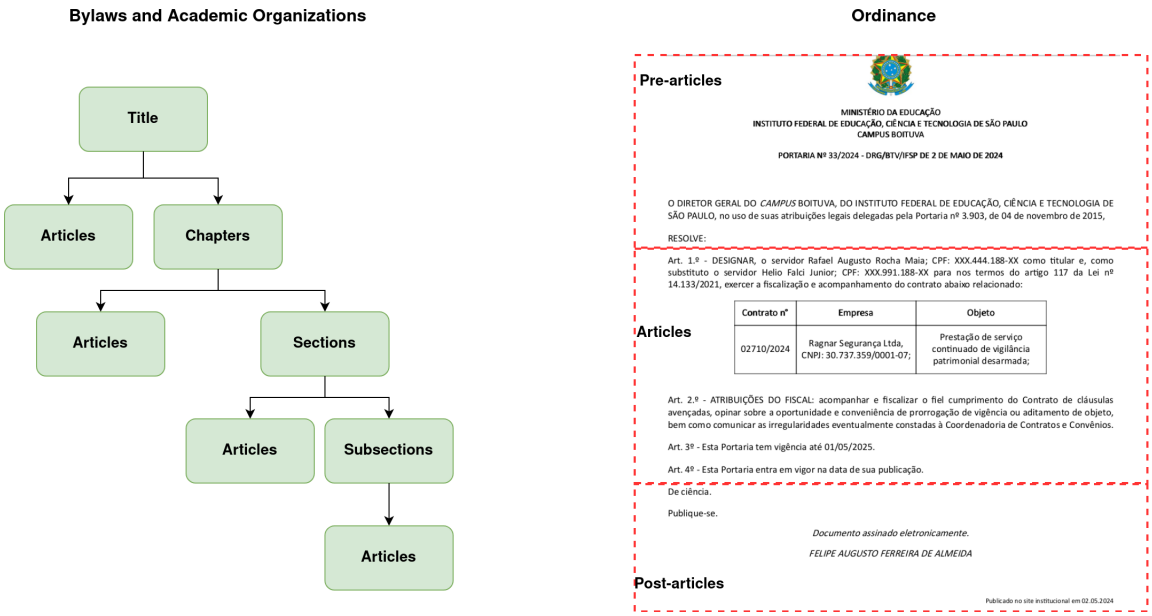

Figure 2: General structure of the documents.

Following that analysis, we utilized the LlamaParse v0.4.4 [8] package to extract the content from the documents. LlamaParse is a tool designed specifically to extract and convert PDF content into structured formats like JSON [8]. Therefore, for each PDF file, we executed LlamaParse, which returned

the extracted data in JSON format. We saved the content of each document in separate JSON files, organized within a dedicated directory structured by year and month.

We processed each of these JSON files, organizing the relevant data and removing elements such as headers, footers, and page numbers, and performing cleaning operations. The organized data was subsequently saved in JSON format. During this process, we encountered instances where LlamaParse failed to extract the content, resulting in empty JSON outputs. Upon analyzing these files, we identified that they were either scanned at low resolution, in landscape orientation, or with misaligned pages. We reprocessed these files multiple times, refining our data organization and addressing text nuances based on our segmentation and retrieval methods.

Using the extracted data, we developed a Question and Answer (Q&A) dataset. For this purpose, we submitted portions of the extracted content from each document to the LLM and requested it to generate questions, answers, supporting contexts, and the corresponding references for each question. Additionally, we required the answers to be formatted in valid JSON. For the bylaws and academic organizations, we requested five questions for each portion, and for ordinances, we requested one question per portion. The prompt used is detailed in Appendix A.

At the end of this Phase, we had two datasets: i) Dataset of documents: this dataset was the foundation for the RAG system, enabling the indexing, retrieval, augmentation, and generation processes; and ii) Q&A dataset: this dataset could be used to evaluate the RAG system.

2.2 Phase II - Implementation of RAG approaches

In this phase, we implemented two different approaches to Retrieval-Augmented Generation (RAG). In the first approach (baseline), we used the BM25 technique as the retriever and sentence windowing for text segmentation. To develop the system, we utilized the LlamaIndex API, which includes implementations of BM25 and segmentation approaches, as well as a query engine. The Query Engine mechanism receives a query, calls the retriever, post-processes the retrieved documents (or nodes, as they are called in LlamaIndex), and sends the query and nodes to the LLM. The LLM then synthesizes the response and sends it back to the query engine, which delivers the answer to the user.

To define the second approach, we experimented with different segmentation techniques and retrievers. We defined ten questions to evaluate the system's answers and to determine whether a new approach was warranted. Based on our decision, a new execution of the "Content Extraction" (Fig. 1) would be performed, and adjustments would be made to the baseline approach as well as the development of the new approach.

For both approaches, we used the same data organization. Since the system has to indicate the specific document(s) and section(s) where the information was found, we adapted the default prompts of the query engine to emphasize the inclusion of the reference (document, chapter, section, etc.) in the answer, as well as the use of the same language of the query in the answer.

2.3 Phase III - Evaluation

To ensure the representativeness of the different types of documents, we created a subset of the Q&A dataset comprising 100 random stratified samples. The distribution of samples is as follows: 71 samples from ordinances, 10 samples from the Academic Organization of Basic Education, 10 samples from the Academic Organization of Undergraduate Courses, and 9 samples from the Bylaw.

To evaluate both approaches, we used that subset and calculated three metrics based on the RAGAS framework [2], as implemented in LlamaIndex: Faithfulness, Relevancy, and Correctness. Faithfulness measures system hallucination; in LlamaIndex, a Faithfulness value can be zero (if the answer was not based on the context - hallucination) or one. Relevancy assesses how well the answer and context align with the query, with a value of one indicating relevance and zero indicating irrelevance. Correctness evaluates the accuracy of the answer against the ground truth, with values ranging from 1 (incorrect answer) to 5 (perfect answer).

3 Datasets

In this work, we developed two datasets (c.f. Subsection 2.1): Dataset of Documents and Q&A Dataset. We can observe, in Fig. 3, an example of the hierarchical structure of the bylaw from the Dataset of Documents. The first title is "TÍTULO I - DA INSTITUIÇÃO"; it does not contain articles but has chapters. In this figure, we can see details of the first chapter, named "CAPÍTULO I DA NATUREZA E

DAS FINALIDADES", which contains three articles but no sections. Ordinances do not have this clear division, so we divided them into pre-articles, articles, and post-articles. By treating each article and part of the ordinance as a document, we obtained 2,113 documents.

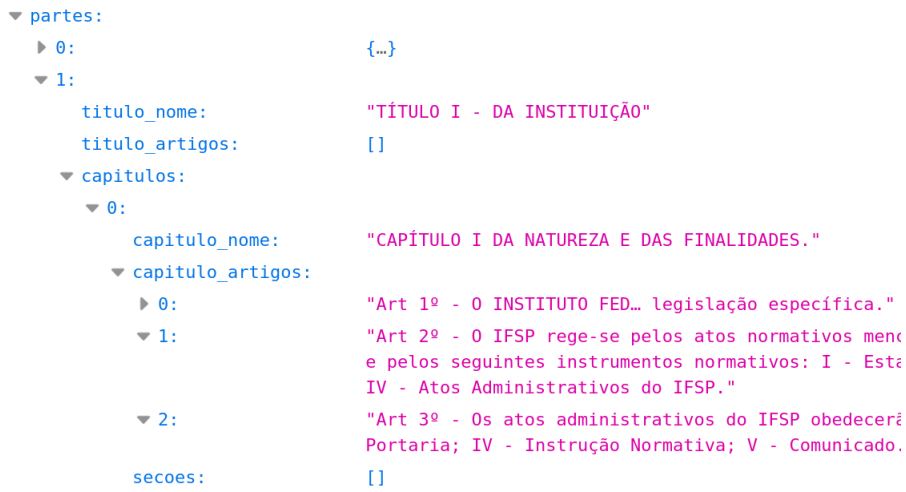

Figure 3: Example from the Dataset of Documents.

We tracked the source of each document, including metadata such as the name of the document (e.g., "bylaw" or the ordinance identification) and its hierarchical structure. Using the example in Fig. 3, the article "Art 1° - ..." would be a document, and its metadata would include: Document = "Bylaw"; Title = "TÍTULO I - DA INSTITUIÇÃO"; Chapter = "CAPÍTULO I DA NATUREZA E DAS FINALIDADES". This way, when the LLM receives the documents to generate the answers, it has access to the document's origin or reference.

In Fig.4, we can observe a sample from the Q&A Dataset. This dataset contains 523 samples, is saved in CSV format, and includes the following fields: Question, Answer, Context, Reference, and the type of document (bylaw, academic organization of basic education, academic organization of undergraduate courses, and Boituva ordinance). During the evaluation, we combined the Answer and Reference fields to create the ground truth answer. We used the type of document to analyze the source of the correct or incorrect RAG answers.

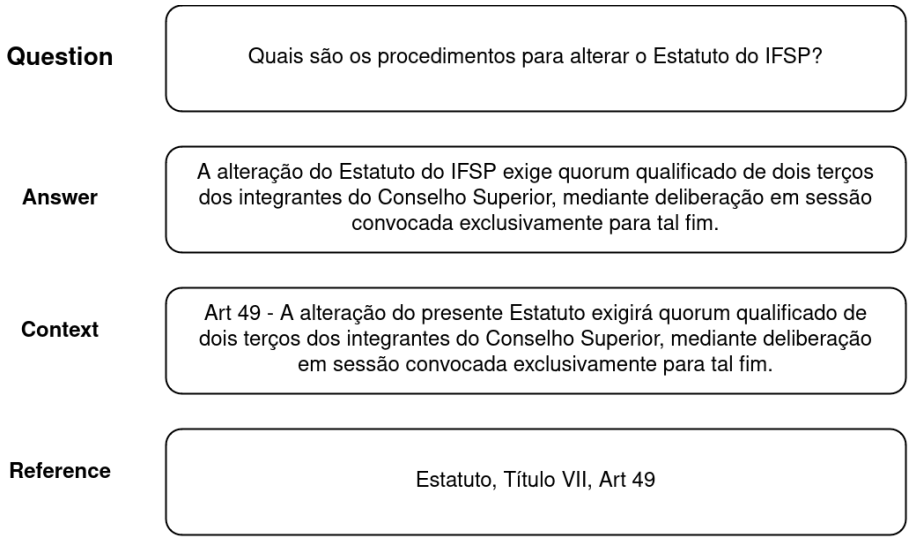

Figure 4: Example from the Q&A Dataset.

These datasets were used in the experimental and evaluation phases detailed in the following sections.

4 Experiments and Results

The conducted experiments on the RAG systems focused on assessing their performance in accurately retrieving and generating information from IFSP's institutional documents. The experiments were based on both naive RAG and advanced RAG approaches. For the preliminary evaluation, we defined the following ten queries:

1. Qual o nome do IFSP?
2. Quais são os campi do IFSP? Além de responder à pergunta, informe o documento, o Capítulo e o Artigo de referência.
3. Quantos campi o IFSP tem? Informe onde você encontrou essa informação
4. O IFSP tem só ensino médio? Além de responder à pergunta, informe o documento, o Capítulo e o Artigo de referência.
5. Quais artigos do Estatuto dizem que o IFSP oferece mestrado e doutorado?
6. O que diz o Art 6º do estatuto do IFSP?
7. O Art 4º do estatuto do IFSP é sobre o que?
8. Sou aluno de graduação e gostaria de saber o que são as transferências especial e ex oficio.
9. O que é Estudante Especial?
10. Em qual portaria Ramon Abilio foi designado para compor a Comissão de Avaliação de Atividades Docentes - CAAD?

For each approach, we executed these queries and manually analyzed the results. We observed whether the implemented approach was able to answer the queries and made decisions on altering the data organization, text segmentation, or retrieval mechanism in the advanced approach.

4.1 Naive Approach - BM25 with Sentence Windowing Segmentation

The naive approach was based on the BM25 as retriever and on sentence windowing segmentation. Fig. 5 illustrates the pipeline of this approach, including the stages of query processing, BM25 retrieval, sentence windowing segmentation, and response synthesis.

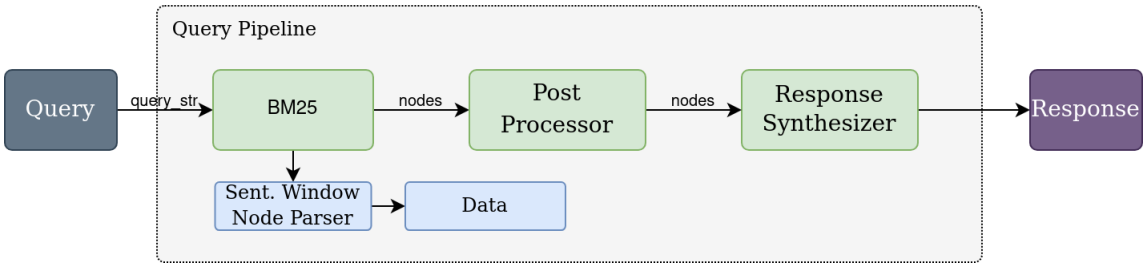

Figure 5: Pipeline of the Naive Approach.

The approach leverages the LlamaIndex library, which provides a suite of tools for building and querying document indexes. Specifically, the BM25 retriever is utilized in conjunction with a language model (LLM) to synthesize retrieved content and enhance the quality of the results.

The Dataset of Documents was loaded from a specified directory into LlamaIndex Document objects, resulting in a list of 2,113 documents. Each document's metadata, including titles, chapters, sections, and subsections, was preserved to maintain context during retrieval. The SentenceWindowNodeParser method segmented the text into sentences and created overlapping windows, with each window containing the current sentence, the preceding sentence, and the following sentence ($windowsize = 3$). This method returned a list of LlamaIndex Node objects comprising 4,672 nodes. Each node contains, among other attributes, the original sentence and the window.

Subsequently, that list of nodes was provided to BM25, and a query engine was constructed using the BM25 retriever, which indexed the nodes using the original sentences and a metadata replacement

post-processor. As illustrated in Fig. 5, the query engine receives the user query, BM25 retrieves relevant nodes, the retrieved nodes are post-processed by replacing the original sentence with its corresponding window, and the response synthesizer (an LLM) synthesizes the content into an answer (response). The system was tested using those ten queries and subsequently evaluated using the Q&A subset.

When testing the system, BM25 took 6 seconds to index the nodes and an average of 1.08 seconds ($\pm$ 0.27 seconds) to answer the ten queries. The system correctly answered (answer + reference) four queries (4, 8, 9, and 10), and for query 1, the answer was correct, but the reference was incorrect. The evaluation of the system with the 100 Q&A samples, using the faithfulness, relevancy, and correctness metrics, took nearly 2 hours.

Regarding the metrics' values, faithfulness was 0.82 ($\pm$ 0.39), relevancy was 0.81 ($\pm$ 0.39), and correctness was 0.62 ($\pm$ 0.39)[2]. Fig. 6 presents a histogram for each metric. We can observe that the system demonstrates nearly 80% faithfulness and relevancy. This means that the system may have hallucinated or not provided relevant context in 20% of the answers. On the other hand, the system correctly answered (correctness $\geq$ 0.875 or 4.5 on the original scale) 53% of the queries.

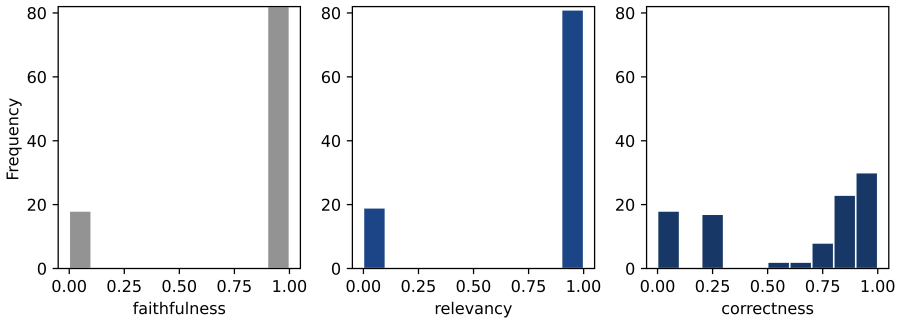

Figure 6: Naive Approach Results.

In summary, this method performs well in terms of faithfulness and relevancy. However, there are significant challenges in achieving consistent correctness, as indicated by the varied distribution. Enhancements are needed to ensure that retrieved documents are not only faithful and relevant but also correct.

4.2 Advanced Approach - Dense Search

The Advanced approach comprises a recursive retriever using dense search combined with a reranking mechanism using the monoPTT5-large model. Fig. 7 illustrates the pipeline of this approach. It shows the flow from the initial query input, through the Recursive Retriever and the Reranker, to the Response Synthesizer, which generates the final response. The Index Store and Data components are crucial for storing and retrieving the nodes required at each step.

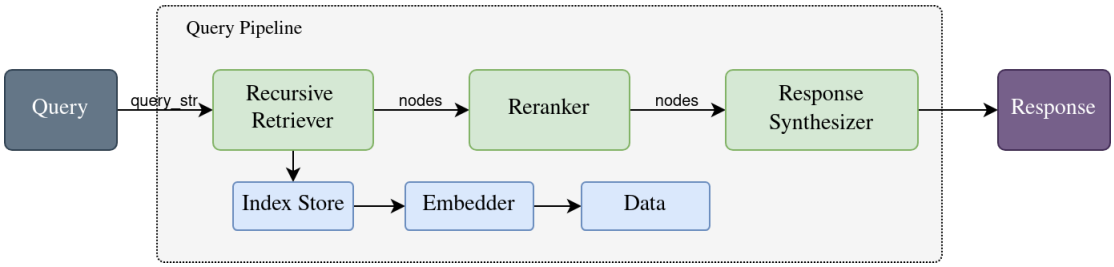

Figure 7: Pipeline of the Advanced Approach.

The primary objective was to enhance the retrieval and ranking of relevant documents in response to user queries by leveraging advanced natural language processing techniques. We utilized the "alfaneo/bertimbau-base-portuguese-sts" model from HuggingFace[3] as an embedder, the llama3-70b-8192 powered

[2]We scaled correctness values between 0 and 1 for comparison with the other metrics' values
[3]https://huggingface.co/alfaneo/bertimbau-base-portuguese-sts

by Groq for query response synthesis, and the "unicamp-dl/monoptt5-large" model[4] for reranking the retrieved documents based on their relevance to the query.

In the Recursive Retriever approach, a search on small chunks leads to larger chunks that contain more context. This part involves conducting multiple iterative searches to obtain richer and higher-quality content [10]. Regarding the data loading, the Dataset of Documents was loaded into LlamaIndex Node objects, resulting in a list of 2,113 nodes. Each node's metadata, including titles, chapters, sections, and subsections, was preserved to maintain context during retrieval. The nodes were chunked into sub-nodes using chunks of 256 and 512 sizes with an overlap of 20 tokens, and linked to their original source node to ensure thorough text segmentation. The nodes and their sub-nodes were indexed using VectorStoreIndex, which employs the embedder and provides a dense retrieval method. The Recursive Retriever was implemented to fetch the top 20 similar nodes for a given query using the vector-based retriever.

Following the retrieval, a reranking step was introduced to enhance the quality of the results. The reranker, powered by the "unicamp-dl/monoptt5-large" model, evaluates the relevance of each retrieved document to the query, assigns scores, and reorders the results accordingly. This ensures that the most pertinent documents are prioritized in the final response. A custom query engine was developed to integrate the Recursive Retriever and reranking functionality, processing user queries, retrieving relevant nodes, and refining the results using reranking to deliver high-quality answers.

In this approach, we had to use the T4 GPU because the embedder required nearly 1.5 hours to create the embeddings, and the reranker also consumed a significant amount of time with each call. With the T4, the time decreased from 1.5 hours to nearly 5 minutes. During the tests with the 10 queries, this approach took an average of 6.75 seconds (± 5.16 seconds) to answer the queries and provided correct answers to five queries (2, 3, 4, 8, and 9). We observed that the larger the context sent to the reranker and the LLM, the more time was required for the final answer.

Due to the token rate limit per minute of Groq, we divided the evaluation of the Q&A subset into two steps. First, we performed the queries and saved the responses. Then, with the saved responses, we calculated the metrics. The entire process took nearly 4 hours. We observed that the time increased as the process continued. We suspect that Groq imposes some delay when it receives numerous requests from the same API key. We saved the results in a CSV file for further analysis.

Upon analysis, we observed that the average faithfulness was 0.56 (± 0.50), relevancy was 0.67 (± 0.47), and correctness was 0.42 (± 0.43). Fig. 8 presents the histogram of each metric. We noted an overall poor performance. Given the tests with the 10 queries, we expected better performance compared to the naive approach. We conjecture that there were possible issues with text segmentation and embeddings because the embedder may have truncated the nodes' texts, or we needed to provide more context from the retriever to the reranker.

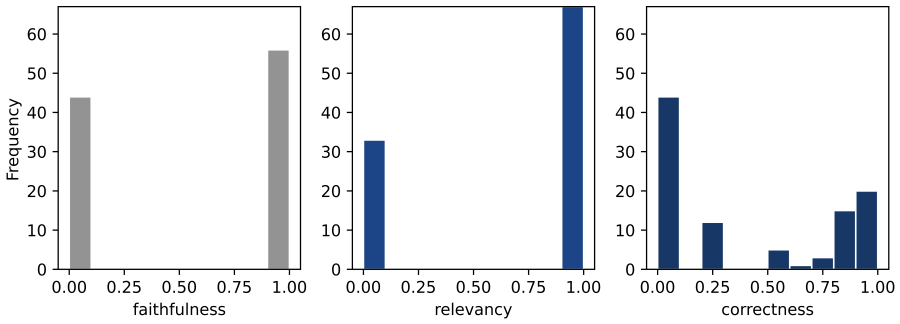

Figure 8: Advanced Approach Results.

The results, in Fig. 8, indicate that while the retrieval methods perform well in terms of relevancy, as shown by the high frequency of 1.00 scores, there are significant challenges in achieving consistent faithfulness and correctness. The varied distribution in correctness further emphasizes the need for improvement in accurately retrieving and synthesizing responses. Overall, the performance in relevancy is promising, but enhancements are needed to ensure that retrieved documents are not only relevant but also faithful to the source content and correct.

[4]https://huggingface.co/unicamp-dl/monoptt5-large

4.3 Comparative Analysis

Aiming to verify if both systems gave incorrect answers to the same queries and to identify any patterns in the queries that make them difficult to answer, we combined the answers from both systems and filtered for correctness metric values equal to 1 (or 0 in re-scaled values). Analyzing the answers, we noted that in all queries, the LLM indicated it was not possible to answer using the provided context. However, we did not perceive any pattern in the queries.

In Table 1, we can observe the values of the naive approach when the correctness of the advanced approach is equal to 1, totaling 44 samples. Both approaches scored 1 in 13 samples, but the naive approach scored 5 in 15. Additionally, in Table 2, we observe that, besides the 13 scores equal to 1, the advanced approach obtained a score of 5 in 2 samples.

Correctness	**#samples Naive Approach**
1.0	13
2.0	9
3.0	1
3.5	1
4.0	1
4.5	4
5.0	15
Total	44

Table 1: Advanced Approach - Correctness = 1.

Correctness	**#samples Advanced Approach**
1.0	13
2.0	1
3.0	2
5.0	2
Total	18

Table 2: Naive Approach - Correctness = 1.

The naive approach of BM25 combined with sentence window segmentation, while less complex, demonstrated better performance on the Q&A dataset compared to the advanced approach with Recursive Retriever and reranking. This approach, while advanced, faced challenged to capture the full context and relevance in recursive queries. The reranking step aims to enhance relevance but can introduce additional complexity and processing overhead. Despite its potential for more nuanced retrieval, the Recursive Retriever's complexity did not translate into better performance for this particular dataset, highlighting the effectiveness and efficiency of the naive approach.

5 Conclusion

In this project, the naive approach using BM25 outperformed the advanced approach based on Dense Search utilized by a Recursive Retriever and a Reranker. Specifically, for correctness scores of 4.5 and above, the naive approach achieved a rate of 53%, while the advanced approach achieved 35%. It is also noteworthy that the advanced approach required a GPU to reduce the time needed for index construction and response generation. In summary, while the naive approach has shown promising results, there is still room for improvement in multiple areas.

We encountered several challenges in building a RAG system, from content extraction to its evaluation. The LlamaIndex framework indeed supports the building process by providing various resources, from extracting content from files to evaluating the entire system. However, deciding which strategies to use has been the main challenge. To reach this setup for the advanced approach, we tested different data organization methods, text segmentation methods such as by sentence, sentence window, chunks, and semantic chunks, retrieval approaches such as naive dense search and recursive retrieval, and language models for the embeddings and reranking. We also faced challenges regarding the prompts used by the query engine due to the requirement to respond by indicating the source of the information and in the same language as the query.

Future work will focus on several key areas for improvement. Firstly, enhancements to the RAG system will be prioritized to improve overall performance and accuracy. Additionally, we will study and implement techniques for extracting content from scanned documents, which remains a challenging aspect of the current system. Improving the question-answer generator to avoid broad questions that can lead to different answers is also a significant goal. Lastly, we plan to incorporate additional databases to enrich the dataset and provide more comprehensive coverage for various types of documents. Furthermore, we will develop a web application to deliver the system to end-users. This web application will ensure a

user-friendly interface and efficient access to the retrieval and question-answering functionalities, making the system more accessible and practical for everyday use.

References

[1] Scott Barnett, Stefanus Kurniawan, Srikanth Thudumu, Zach Brannelly, and Mohamed Abdelrazek. Seven failure points when engineering a retrieval augmented generation system. In *Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI*, pages 194–199, 2024.

[2] Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. RAGAs: Automated evaluation of retrieval augmented generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158, March 2024.

[3] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

[4] Groq. Why groq. `https://wow.groq.com/why-groq/`, 2024. Accessed in: 27 jun. 2024.

[5] IFSP. Federal Institute of São Paulo. `https://www.ifsp.edu.br/`, 2024. Accessed in: 27 jun. 2024.

[6] IFSP. Federal Institute of São Paulo - Boituva's Campus. `https://btv.ifsp.edu.br/`, 2024. Accessed in: 27 jun. 2024.

[7] Jerry Liu. LlamaIndex. `https://github.com/jerryjliu/llama_index`, 2022. Accessed in: 27 jun. 2024.

[8] LlamaIndex. LlamaParse. `https://docs.llamaindex.ai/en/stable/llama_cloud/llama_parse/`, 2024. Accessed in: 27 jun. 2024.

[9] IFSP Federal Institute of São Paulo. Estatuto. `https://drive.ifsp.edu.br/s/QDYsuUCkSIKJeHh/`, 2014. Accessed in: 28 jun. 2024.

[10] Penghao Zhao, Hailin Zhang, Qinhan Yu, Zhengren Wang, Yunteng Geng, Fangcheng Fu, Ling Yang, Wentao Zhang, and Bin Cui. Retrieval-augmented generation for ai-generated content: A survey. *arXiv preprint arXiv:2402.19473*, 2024.

A Prompt for the Question and Answer Generation

```
Você é um funcionário do Instituto Federal de São Paulo (IFSP) e está preparando um
documento com perguntas e respostas frequentes sobre o IFSP.

Leia os documentos e gere perguntas com suas respectivas respostas, partes completas
do documento que embasam a resposta (contextos), e referências com nome dos documentos,
título, capítulos e artigos que embasam a resposta.

No contexto, coloque as partes mais importantes para a resposta e não somente a
indicação do artigo.

Exemplos de contextos esperados:
1) 'context': 'Art 1º - O INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO
PAULO { IFSP, com sede e foro na cidade de São Paulo, criado nos termos da Lei nº.
11.892, de 29 de dezembro de 2008, constitui-se em autarquia federal, vinculada ao
Ministério da Educação, detentora de autonomia administrativa, patrimonial, financeira,
didático-pedagógica e disciplinar.'
2) 'context': '§ 4º- O IFSP possui limite de atuação territorial para criar e extinguir
cursos, bem como para registrar diplomas dos cursos por ele oferecidos, circunscrito ao
Estado de São Paulo, aplicando-se, no caso de oferta de ensino a distância, legislação
```

específica.’

Exemplos de contextos incorretos
1) ’context’: ’Art 2º - O IFSP rege-se pelos atos normativos mencionados no caput do Art 1º -, pela legislação federal e pelos seguintes instrumentos normativos:’
2) ’context’: ’Art. 4º do Estatuto do IFSP’

Ao gerar o contexto, verifique se é possível extrair a resposta a partir dele. Se não for possivel, complete o contexto.

Você deve enviar sua resposta em formato JSON válido com a seguinte estrutura:
[{\"question\": \"pergunta\", \"answer\": \"resposta\", \"context\": \"contexto\", \"reference\": \"referência\" }]

A quantidade de perguntas deve ser no máximo 1.

Formule a pergunta como um estudante de ensino médio ou outra pessoa que não conheça o IFSP.

Se possível, formule questões que envolvam dois ou mais temas específicos do IFSP.

Se possível, formule questões que envolvam dois ou mais artigos.

Atenção!
1) Não utilize seu conhecimento prévio. Suas respostas devem ser baseadas nos documentos que você receberá.
2) Responda em Português do Brasil.
3) Não adicione qualquer outro texto, explicação ou instrução em sua resposta. Limite-se à resposta no formato JSON especificado.
4) Não gere perguntas repetidas.