Archives available at journals.mriindia.com

International Journal on Advanced Computer Engineering and Communication Technology

ISSN: 2278-5140

Volume 15 Issue 01, 2026

Evaluating Memory Usage and Processing Time in AI-Driven Document Parsing: Extracting Information From Scanned, PDF, And Excel Formats

Prithvi Panchineni
Data engineer, Texas
Email: prithvi.dataconsulting@gmail.com

Peer Review Information

Submission: 28 Jan 2026

Revision: 20 Feb 2026

Acceptance: 06 March 2026

Keywords

Large Language Model; Retrieval Augmented Generation; Document; Performance Metrics; Chatbot.

Abstract

Recent developments in large language models (LLMs) and retrieval-augmented generation (RAG) approaches have changed information management across industries. These breakthroughs have made it possible to design document interaction systems that are intelligent, efficient, and aware of their context. The use of these technologies in corporate document management is the focus of this thesis, which also introduces a new chatbot solution that improves document retrieval and gives users accurate, context-driven help. Using this strong retrieval framework as a foundation, the Retrieval-Augmented Generation (RAG) technique is utilized to merge the optimal embedding model with five different LLMs. To measure how well these models' function, we look at how well they generate context-aware responses and how well they match up with user expectations. Metrics such as generating time are also evaluated to determine the effectiveness of the system. This thesis shows how combining LLMs, RAG, and advanced embedding methods can change the way corporate documents are managed by making access to knowledge more reliable, scalable, and quick. This paper illustrates the potential for deploying LLM-powered, RAG-driven systems that enable efficient and contextually relevant user interactions across sectors. This promise is highlighted by explaining the system's architecture, methodology, evaluative metrics, and performance benchmarks.

Introduction

Flosslab is a company that was established in 2007 as the first spin-off from the University of Cagliari. Its primary focus is on delivering innovative software solutions that are specifically designed for both the public and private sectors. The organization has competence in a wide range of important areas, including the digitalization of processes, the administration of documents, the building of websites, and blockchain technology. With the incorporation of Flosslab into the Net Service SpA Group in 2018, the company was able to greatly improve its capacity to provide digital solutions that are both scalable and innovative [1]. The dedication of Flosslab to research and development is an essential component of the company's business operations. The organization, which is supported by Sardegna Ricerche, has developed a rigorous innovation program with the intention of incorporating cutting-edge technologies into its products and services. Notable projects include CAFCHA, which uses blockchain technology to assure traceability and safety in the agrofood industry, and CASCO, which is a project focused on increasing smart contract security by discovering vulnerabilities and tracking token-related data within blockchain networks.

Both initiatives are good examples of noteworthy initiatives. Recently, Flosslab has broadened its emphasis to investigate the revolutionary possibilities of artificial intelligence (AI), which is in line with the forward-looking strategy that the company consistently takes. As part of this program, Natural Language Processing (NLP) tools will be utilized to improve document management systems [2]. These tools will enable semantic comprehension, context-aware search, and more intuitive data interaction. Flosslab's goal is to improve operational efficiency by harnessing these capabilities, as well as to make information more accessible and actionable for users. I was able to contribute to the development of an advanced solution that was created to maximize document retrieval and management through my partnership with FlossLab, which matches with this objective on a fundamental level.

To improving the efficiency and effectiveness of accessing corporate papers, this research makes use of Retrieval-Augmented Generation (RAG), a cutting-edge artificial intelligence technology that was first proposed by [3]. The problem of managing enormous information repositories is addressed by the solution, which provides streamlined and user-friendly access to essential data. This is accomplished through the incorporation of AI-driven techniques such as semantic search and chat-based support. This thesis provides documentation of the joint research and development process, as well as the results of the collaboration. It does this by analyzing the potential of RAG and AI technologies to improve corporate document management systems, which ultimately contributes to Flosslab's aim of providing new solutions that are powered by technology.

In today's data-driven business world, firms are responsible for managing large numbers of documents, including but not limited to contracts, reports, rules, and other essential information. When it comes to efficiently accessing important information, traditional document management systems frequently discover that they have limitations, particularly as the amount and complexity of documents continue to grow [4]. Keyword-based search algorithms frequently fail to capture contextual nuances and semantic relationships, which results in decreased productivity and the possibility of errors when employees are attempting to access information that is vital to their work. For instance, a physician who searches for "diabetes treatment" in a system that is based on keywords can get results that are not relevant to the topic at hand, such as out-of-date procedures or news stories that are unrelated to the topic, rather than the most recent clinical recommendations that are particular to the context. Considering the relevance of semantic search systems, which make use of contextual awareness to provide information that is correct and pertinent, this inefficiency highlights the significance of these systems. The purpose of this research is to remedy these inefficiencies by utilizing Artificial Intelligence (AI) to accomplish the advancement of document management capabilities. In particular, it investigates the ways in which embedding models, semantic search, and Retrieval-Augmented Generation (RAG) might dramatically improve the processes involved in document retrieval. The project aims to provide faster and more accurate retrieval while also decreasing the cognitive strain that users are under. This is accomplished by enabling AI-driven interpretation of the meaning and context behind user queries. The purpose of this thesis is to contribute to more efficient workflows and improved information accessibility in corporate contexts by bridging the gap between classic document management systems and new alternatives that are enhanced with artificial intelligence [5].

The project intends to integrate modern artificial intelligence approaches in order to address the issues that are associated with standard document management systems. In the following order, the goals are structured:

- A text extraction and structuring system that is efficient In order to get unstructured business documents (like PDFs) ready for semantic processing, it is necessary to investigate and execute the most effective approaches for extracting and structuring content from these documents.
- The Selection and Implementation of the Embedding Model System In order to generate semantic representations of corporate documents of a high quality, it is necessary to evaluate and choose the most appropriate embedding models.
- Optimization of Advanced Retrieval Techniques Utilize advanced retrieval tactics, such as semantic reranking and Maximal Marginal Relevance (MMR), in order to enhance the accuracy and relevance of the retrieval process [6].
- Put a variety of large language models (LLMs) through their paces Explore and test a number of different LLMs in order to create replies, with the goal of determining which models are the most appropriate for incorporation into the document management system.

- An Analysis of the System In order to test and assess the performance of the improved document management system, metrics such as retrieval accuracy with MRR, response time, and user satisfaction should be utilized.

Additionally, evaluate the quality of the outputs that have been generated by employing deterministic metrics (such as BLEU and ROUGE) as well as LLM-based measures [7].

The validation of improvements can be accomplished by comparing these outcomes to the baseline methodology. Through the accomplishment of these objectives, the purpose of this research is to provide a practical solution that has a low error rate and a deployment time that is efficient. Even when the size of the dataset and the number of active users increases, the suggested solution is designed to be scalable, which means that it will keep query response times and resource efficiency at acceptable levels. The system aims to improve corporate workflows and user productivity by increasing document accessibility and retrieval efficiency. This will result in faster job completion, higher accuracy, decreased cognitive load, more ease of use, and increased user satisfaction. Document accessibility and retrieval efficiency has been improved.

Literature Survey

Methods that have been used traditionally for document management systems When it came to retrieving information, document management systems (DMS) relied on conventional approaches before the emergence of artificial intelligence (AI). Although these methods were foundational, they had a number of shortcomings that contemporary approaches driven by artificial intelligence try to address.

1. Searching Based on Its Keywords:

To get content based on user queries, keyword-based search relies on exact string matching as its primary method of retrieval. Despite the fact that this method was computationally efficient, it lacked semantic comprehension, which frequently led to findings that were either irrelevant or incomplete. An example of this would be a query for "report" that would not receive documents that contain synonyms such as "summary" or "analysis" unless they were specifically indicated to the query [8].

2. Search Using Boolean:

The Boolean search system was the first to implement logical operators such as AND, OR, and NOT to refine queries and provide greater control over the results of the search. Despite the fact that it was accurate in certain circumstances, Boolean search required users to comprehend complicated query syntax. Furthermore, it was still restricted to exact phrase matching, which meant that it did not take into account semantic nuances [9].

3. Metadata-Based Search Method:

In order to get information, a search that was based on metadata made use of structured qualities such as title, author, and date. The effectiveness of this method was limited by the availability and consistency of metadata, which frequently needed manual tagging [10]. Despite the fact that this method enabled accurate filtering, its effectiveness was limited. 29:30

The complete content of documents was indexed by full-text search, which made it possible to retrieve information based on any phrase contained inside the text. Even though it offered a more comprehensive coverage than keyword or metadata-based search, it lacked semantic comprehension, which frequently led to an excessive amount of information and results that were not relevant [11]. Hierarchical and Faceted Browsing Documents were sorted into predetermined categories or facets using hierarchical and faceted browsing, which enabled users to refine their searches in a dynamic manner. Maintaining hierarchical structures, on the other hand, required a significant amount of effort, and the strategy was less effective for data that was either poorly tagged or not structured well [12].

Retrieval of Information Based on Rules, Section 3.1.6 Specific information was extracted from documents through the application of rule-based systems, which utilized established patterns. The adaptability of these systems to new or unstructured data formats was lacking [13], despite the fact that they were useful in specialized domains like as the legal and medical fields. Document Management Systems that Make Use of Artificial Intelligence The incorporation of artificial intelligence into document management systems has resulted in the introduction of sophisticated methods for the efficient retrieval of information, awareness of contextual factors, and interactions that are friendly to users. The following is a list of examples of contemporary document management systems that make use of AI methods:

Some Examples of Artificial Intelligence Methods Retrieval-Augmented production (RAG) is a technique that integrates information retrieval with generative models, so enabling the production of responses that are both dynamic and aware of their context. When it comes to managing unstructured data and providing answers to difficult questions, RAG is particularly effective since it makes use of relevant

documents that are obtained from a knowledge base [1]. ú Searching for semantic information using dense retrieval models: The utilization of embeddings by dense retrieval frameworks, such as Dense Passage Retrieval (DPR), allows for the comprehension of the semantic connections that exist between queries and documents, which ultimately results in the provision of accurate search results [38].

Question Answering Systems: Systems that are powered by artificial intelligence make use of pre-trained large language models (LLMs) such as GPT to provide answers to user questions by extracting information directly from documents and offering responses that are both brief and pertinent [14]. Automated Metadata Extraction: Artificial intelligence systems employ natural language processing (NLP) to automatically extract metadata from documents. This includes titles, authors, and dates, among other essential information. By doing so, manual tagging efforts are reduced, and the arrangement of documents is maintained in a consistent manner [15].

A Document Clustering and Classification: Machine learning models are utilized to cluster or classify documents into categories based on their content, hence simplifying the processes of organization and retrieval. It is common practice to employ these strategies to facilitate efficient navigation in legal or medical document repositories [16]. Optical Character Recognition (OCR) systems that are combined with natural language processing (NLP) are able to extract text from scanned documents and evaluate the content for further processing. This allows the text to be accessible for search and retrieval in document management systems [17].

AI-powered chatbots communicate with users through the use of natural language, providing individualized document search and support. This type of interface is known as chat-based interfaces. These solutions improve accessibility and convenience of use in situations that take place in corporate settings [18]. Restrictions affixed to the Existing Works Despite the fact that the works that were examined demonstrate substantial progress, there are still major gaps: Adaptability to Corporate Domains: The majority of embedding and retrieval systems are evaluated on datasets that are intended for generic purposes, which makes it difficult for them to adapt to specific corporate domains [19-22].

Integration of Retrieval and Generation:

Within the realm of business environments, there are just a handful of research that investigate the utilization of retrieval and generative models in conjunction with one another. Managing Unstructured Data: Numerous methods encounter difficulties when it comes to the processing of unstructured documents, such as PDFs with different layouts, which are frequently encountered in corporate environments. The purpose of this review is to describe the progression of document management systems, beginning with conventional search methods and progressing to solutions that incorporate artificial intelligence. While foundational works like keyword-based and Boolean search laid the groundwork for information retrieval, AI-driven methods such as semantic search and RAG have significantly advanced the field. By addressing the identified limitations, this thesis contributes a novel approach to enhancing corporate document management systems using embeddings, semantic search, and RAG frameworks [23].

Research Methodology

To properly implement a Retrieval Augmented Generation (RAG) framework, the methodology for this project takes a methodical approach to the implementation process. The preprocessing of data, the selection of an optimal embedding model, the evaluation of retrieval strategies, the generation of replies utilizing Large Language Models (LLMs), and the evaluation of performance using evaluation metrics are all important phases. These actions are intended to solve the difficulties associated with managing and retrieving information from unstructured corporate documents, with the goal of ensuring that the information is both precise and relevant. A visual representation of the workflow is shown in Figure 1 & 2, which highlights the additional contributions that were added on top of the conventional RAG pipeline as in Figure 1 and 2.

The dataset was made up of PDF documents that were contributed by Flosslab. These documents were arranged in a hierarchical folder structure in a methodical manner. In order to classify documents according to projects and the specific document categories associated with certain projects, the structure was designed:

- Project Level: These folders at the top level were each named after a particular project, and they grouped together all the papers that were associated with that project.
- The category level: Documents were then sorted into three unique subfolders within each project folder using categories as follows.

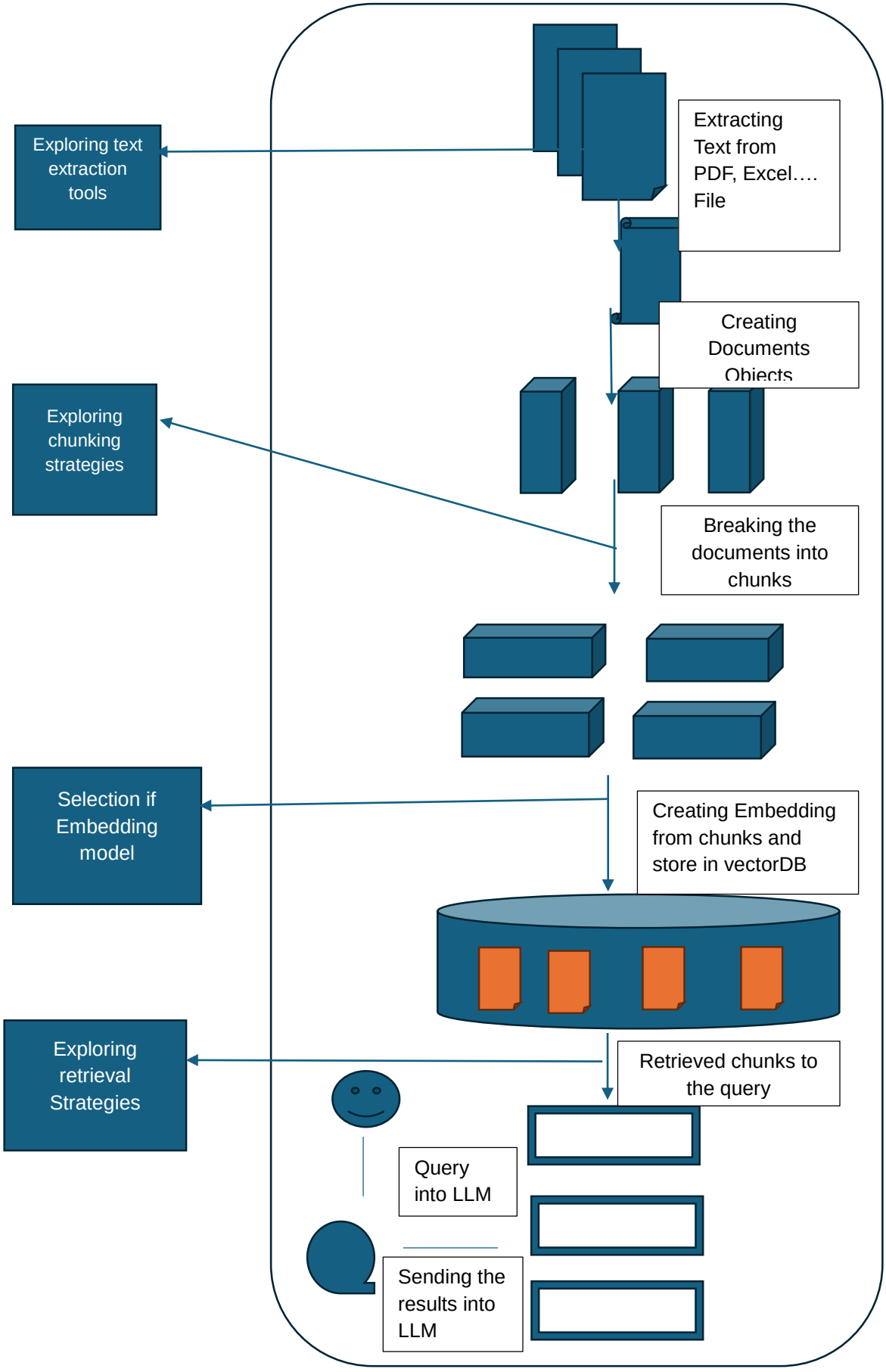

Figure 1. Proposed RAG Pipeline

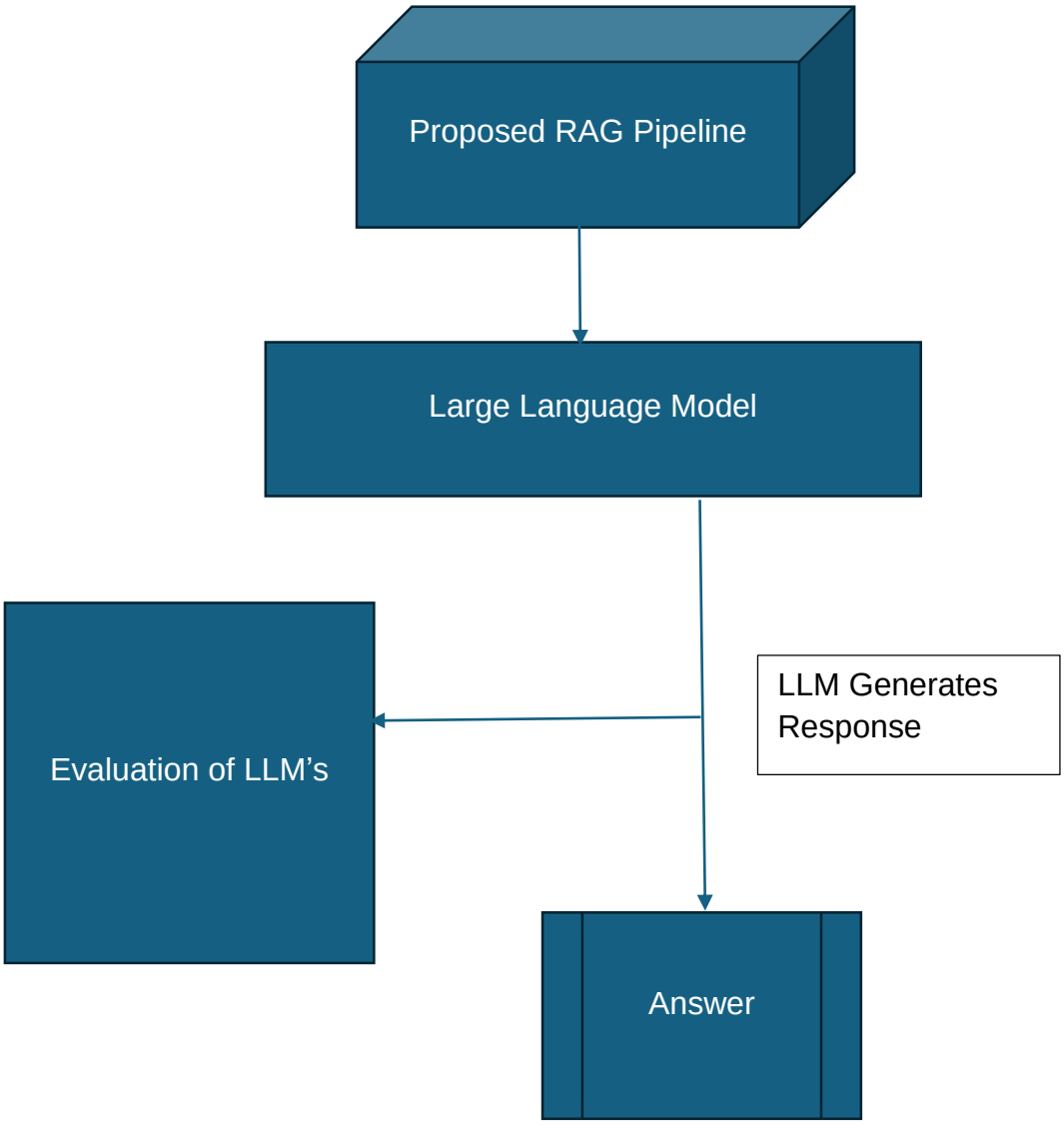

Figure 2. Integration of Proposed RAG Pipeline with LLM's

The Technical Project Documentation, also known as the Tecnica di Progetto, is a type of project documentation that often includes thorough specifications, technical reports, and project plans. Oerta Commerciale refers to commercial offers, which encompass financial proposals, price models, and other detailed information pertaining to the firm. Technical offers, which include technical evaluations, requirements analyses, and feasibility studies, are referred to as Oerta Tecnica: Technical offers.

- Document Level: Each category subfolder had the PDF documents that were pertinent to that category, and each document had its own unique format, which reflected the variety of styles and levels of complexity that are typical of business documentation.

A wide variety of data types were included in the documents that were examined, including the following: Narrative sections, headings, and technical descriptions are examples of text, which frequently employs a variety of font styles, sizes, and alignment strategies.

Tables are a type of structured data that can range from straightforward grids to intricate multi-page tables that contain merged cells and embedded annotations. Pictures and diagrams are visual content that are frequently used to complement written data. Examples of this type of content include charts, schematics, and pictures that have been annotated. Segments of programming code that are styled with indentation, syntax highlighting, or embedded within paragraphs are referred to as code snippets. It was impossible for a single extraction approach to effectively handle all the layouts due to the variety of these documents, which provided a new set of issues throughout the preprocessing stage. Each data type necessitated customized methodologies to guarantee precise extraction while maintaining semantic integrity and structure, rendering the preprocessing

pipeline an indispensable component of the methodology.

Recognizing the Components of the Document:

Several techniques were utilized to extract and organize the essential components of the papers. Particular attention was paid to the titles and subtitles, subscripts and superscripts, headers and footers, tables, and the table of contents and their respective sections. The purpose of these strategies was to guarantee accurate extraction while guaranteeing that the structural integrity of the original information would be preserved: Titles: The size of the font was utilized as a heuristic for the purpose of identifying titles. It was presumed that higher font sizes, such as those exceeding 12 points, may be indicative of headings or titles [24]. By doing an analysis of the patterns that were present in the first few pages of the document, additional rules were created in order to differentiate titles from other elements, such as entries in the table of contents.

The identification of subscripts and superscripts was accomplished by analyzing the location of characters along the vertical axis (y-axis) in relation to the characters that came before them. Through the use of this method, the objective was to identify variations in text alignment and differentiate these pieces from regular text. A number of libraries, including PyMuPDF and PDFPlumber, are utilized for the purpose of extracting tabular data. Analysis of line separations, grid patterns, and text alignment were the methods that these tools utilized in order to determine table structures.

The objective was to maintain the logical structure of the tables so that they could be processed further down the line. Headers and Footers: In order to identify headers and footers, the text that was found inside the top and bottom 10% of each page was retrieved. This included features such as page numbers, document titles, and watermarks, all of which were separated out for the purpose of possible removal or independent processing. The existence of recurring patterns and entries written in large type on the first pages of the document was studied in order to determine the parts that would be included in the table of contents. With the intention of distinguishing them from the primary text, these entries were f lagged [25]. Approach to Text Extraction,

Several libraries were put through a series of tests to determine how well they handled a variety of document layouts and formats. The purpose of these tests was to extract text from documents. PDFPlumber, PyMuPDF, and LangChain PyPDFLoader were among the tools that were considered for testing. All of the libraries were evaluated based on their capacity to maintain structural elements such line breaks, text alignment, and overall page formatting. While this procedure, efforts were taken to tag the items that were retrieved. These efforts included tagging titles, tables, and other important components with tags that could be identified. These tags were first developed with the intention of improving subsequent tasks such as chunking and semantic analysis. Nevertheless, the primary focus remained on the development of a robust extraction pipeline that could manage a wide range of document forms.

The Organization of Documents and Their Storage in Vector Databases:

After the text was extracted, it was transformed into document objects by utilizing the facilities that were made available by LangChain. There were two components that made up each document object: ú Metadata is the incorporation of qualities such the project name, subject, and document name in order to enhance the specificity of retrieval.

Chunking for the Purpose of Retrieval Optimization

The text was broken up into smaller, more organized parts that were referred to as chunks. This was done in order to improve the precision of retrieval and to guarantee that the Large Language Model (LLM) provides replies from content that is both relevant and feasible. The chunking was meticulously planned out to ensure that the size and context of each part were in harmony. while the chunk size is too tiny, the context becomes fragmented, which decreases the effectiveness of embeddings and retrieval. This is one of the challenges that can arise while attempting to chunk. Furthermore, the semantic connections that exist between words might be severely disturbed or even lost. a. It runs the danger of surpassing the token limit of the embedding model if the chunk size is too large, which might result in processing that is either insufficient or insufficiently complete. Additionally, the results may become diluted with material that is not pertinent to the inquiry, which contributes to the overall confusion.

Techniques for Chunking Using the LangChain Recursive Text Splitter, three different approaches were evaluated for the purpose of preparing chunks for embedding and retrieval. These approaches are as follows: A variation of the natural breakpoints approach that handled separators as regular expressions is referred to as the Natural Breakpoints approach (Regex). In order to provide more flexible chunk splitting

depending on different text structures, the separators were set to ["\n\n", "\n", " ", ""] by using the is_separator_regex=True command. The Natural Breakpoints Method (Basic) was utilized to generate chunks by utilizing natural breakpoints, such as sentence boundaries, and the separators were set to ["\n"].
In order to conform to the logical structure of the text, this configuration divided the text into chunks depending on the sequence of newline characters.

The Token-Based Sliding Window Method:
This strategy utilized a sliding window that was based on tokens and also included chunks that overlapped in order to keep the context between the segments intact. is_separator_regex=False was incorporated into the configuration in order to guarantee that separators were preserved while simultaneously preserving consistency between chunks. It was determined whether or not each strategy was capable of effectively structuring text in order to facilitate embedding and retrieval. Following the completion of the procedure, a one-of-a-kind UUID was assigned to each chunk as well as each document.
Both a chunkId and a documentId were included in the metadata for each chunk. This ensured that there was a clear relationship between each chunk and the document that it would be associated with.

The Process of Choosing the Embedding Structure:
In order to accomplish the goal of evaluating the effectiveness of embedding models for this project, a structured approach was developed to imitate information retrieval tasks that are performed in the actual world. The selection of the embedding model was one of the steps involved in this procedure. The generation of inquiries, the mapping of those queries to document chunks, and the guaranteeing of traceability for accurate evaluation are all 37. After that, the Mean Reciprocal Rank of the chunks that were retrieved for each query according to each embedding model is measured.

The Dataset for the Embedding Evaluation:
Create queries and chunks of data The following were the steps that were engaged in the process of creating the embedding evaluation dataset:

- A Selection of Chunks:

Two of the largest chunks from each page were chosen to serve as typical portions for the retrieval activities that should be performed. To ensuring that each document has the greatest possible amount of significant content, these pieces were selected according to their size. Generation of Queries In order to replicate more common information retrieval circumstances, two queries were produced for each of the chunks that were chosen. To generate prompts that were of high quality and contextually appropriate, the mixtral:8x22b-instruct model was utilized. These prompts were adjusted to the content of each chunk.
Assigning of the UUID Each query was given a unique identification, sometimes known as a UUID, to ensure that traceability was maintained and to make query management more efficient. During the entirety of the evaluation process, this guaranteed that clear references and structure were maintained. The JSON Mapping of Documents That Are Relevant The mapping of each query to its appropriate chunks and accompanying pages was accomplished through the creation of a structured JSON file. During the process of evaluating the performance of the embedding model, this mapping functioned as the ground truth. It offered a transparent and well-organized connection between queries, chunks, and documents. Documents are stored in ChromaDB,
The information contained in the papers was kept in ChromaDB, a local vector database that was chosen because of its capacity to safely store data on-premises, thereby guaranteeing the confidentiality of the company's business documents. According to ChromaDB, the following procedures were engaged in the storing process: Incorporating the text from the documents into the system.

- Bringing about dense vector representations:

Also known as embeddings, by utilizing the embedding model that was chosen. 3. Storing these embeddings along with the metadata that is associated with them in order to facilitate rapid retrieval based on similarity. When it comes to embedding models, 4.4.3 evaluation An evaluation of several state-of-the-art embedding models was carried out in order to determine which one would be the most suitable for the project. The most recent versions of the following: mxbai-embed-large:latest, nomic-embed-text:latest, bge-m3-f16:latest, and multilingual-e5-large

Procedure Evaluation Based On LLM
Using each embedding model to generate embeddings for the documents that are being processed. The second step involves using the pre-generated queries to query the vector database. The third step involves comparing the results that were retrieved to the ground truth that was described in the JSON mapping file to evaluate them for accuracy and relevance. Through the utilization of the Mean Reciprocal

Rank (MRR), the performance of every embedding model was analyzed to determine the effectiveness of retrieval efficiency.
The embedding time, which is the amount of time required to transform a document chunk into its dense vector representation, and the retrieval time, which is the amount of time it takes for the vector database to get the relevant chunks, were the two major metrics that were used to evaluate the effectiveness of each model. When it comes to evaluating the practical usefulness of embedding models, these measures are necessary, particularly in situations that need real-time processing or large-scale document management.

1. RAG Implementation:

The system obtains relevant chunks by querying the vector database based on the input query. This occurs after the system has stored the document chunks in the vector store using the embedding model that was chosen. In the subsequent step of the generation process, these retrieved chunks are utilized as contextual information respectively. The following is the implementation of the RAG pipeline, which stands for retrieval-augmented generation:
For the purpose of determining which retrieval strategy would be the most effective for this project, a number of different approaches were tested and reviewed. Following is a list of the retrieval strategies that were investigated: Basic Normal Retriever: This technique employed similarity search to return the top 10 chunks for each query based on embedding similarity through the utilization of similarity search. The Normal Reranked Retriever is a technique that extends the normal retriever by reranking the top 10 chunks that were retrieved by employing the BAAI/bge-reranker-v2-m3 model. The reranker performed a more precise evaluation of the importance of each chunk in order to increase the accuracy of the ranking.

2. The Implementation of RAG:

LOTR, which stands for "Merged Retriever," is a hybrid strategy that combines similarity search with Maximal Marginal Relevance (MMR) to strike a balance between the relevance and diversity of the results that are returned. There was a total of twenty chunks that were retrieved, with each retrieval technique contributing ten chunks to the total. Following the completion of the chunk merging process, superfluous chunks were eliminated to improve the overall relevancy of the result set. Afterwards, the remaining chunks were reranked with the help of the BAAI/bge-reranker-v2-m3 model to guarantee the highest possible level of contextual accuracy and relevance.
The Generation of 4.5.2 Out of the ten chunks that were recovered and reranked, the five chunks that were deemed to be the most relevant were chosen for the creation of responses. After that, these chunks were utilized as contextual input for a variety of Large Language Models (LLMs) that are considered to be technologically advanced.

3. Evaluation of the Retrieval:

The Mean Reciprocal Rank (MRR) metric, which evaluates the rank of the first relevant result for each query, was utilized in order to analyze the retrieval algorithms that were utilized. With the help of this statistic, one may gain an understanding of how effective each retrieval approach is in terms of retrieving the chunks that are most pertinent to a particular query. The Evaluation of Generations The Continuous-Eval framework, which utilized the following metrics, was utilized in order to analyze the responses that were generated for each LLM and retrieval approach. κ Metrics that are Deterministic:– The Accuracy of the Answer – The Reliability ú Metrics Based on the LLM:— Faithfulness Based on LLM BasedForty LLM-Based Answer Correctness and LLM-Based Style Consistency are examples of methodologies.

When carrying out the LLM-based evaluations, the mixtral:8x22b-instruct model was utilized, and its enhanced contextual understanding skills were utilized to their full potential. The average generation time for each query was recorded for all LLMs in order to evaluate their efficiency. The results of the evaluation were then examined in order to establish the ideal combination of retrieval strategy and LLM, taking into account criteria such as correctness, fidelity, and style consistency.

Perfromance Analysis

Human Assessment Methodology

To evaluate the accuracy and quality of the text extraction process, a **manual review** was conducted on a random sample of **134 business documents**. The reviewers were instructed to verify:

1. **Completeness** of extracted content (e.g., presence of all sections such as titles, paragraphs, tables).
2. **Preservation of document structure**, such as paragraph boundaries, line breaks, and font-based formatting cues.
3. **Accuracy of extracted tables**, including alignment, merged cells, and semantic fidelity.
4. **Correct identification** of headers, footers, superscripts/subscripts, and special characters.

Each document was annotated using a binary pass/fail checklist for key elements (e.g., "Table correctly extracted – Yes/No"). Reviewers were professionals familiar with document formats used at Flosslab and followed a standardized rubric to ensure consistency. Discrepancies were cross-checked by a second reviewer to ensure inter-rater reliability. The results showed that **less than 50% of documents passed all evaluation criteria**, highlighting the need for more robust extraction techniques.

A human assessment was performed on 134 business papers to evaluate the quality of the text extraction process. The results showed that fewer than fifty percent of the documents were successfully processed utilizing the procedures that were employed. Here are some important observations: In contrast to other programs, PyMuPDF was able to accurately identify the borders between sentences and paragraphs, whereas other methods frequently resulted in fragmented text.

A significant number of tables, particularly those with intricate layouts, were either partially extracted, misaligned, or completely excluded from the analysis. Subscripts and superscripts: The detection of subscripts and superscripts was uneven due to the limitations of heuristics, despite the fact that human fixes were reasonably trivial. In terms of titles and formatting, it was observed that titles and subtitles were not consistently detected, and there was a high rate of misclassification due to the only dependence on font size.

These findings brought to light substantial limitations in the extraction methods that are now in use and shed light on the necessity of developing more sophisticated tools that are specifically designed to extract structured data from a variety of corporate document formats. Because document layouts might vary greatly from one another, it was discovered that customizing the extraction method for each individual component of a document was neither efficient nor effective. Instead, a universal framework that is based on PyMuPDF was chosen because of its powerful layout-aware capabilities and its capacity to maintain meaningful structure across a wide range of document types. After the text was extracted, it was manually processed to remove special characters and cleaned to avoid needless breakpoints, such as those created by sentence wrapping. This was done to ensure that the output was more cohesive and structured.

Evaluation of the Embedding Model:

Mean Reciprocal Rank (MRR), which was used to evaluate the ranking effectiveness of the embedding models, Average Embedding Time, and Average Retrieval Time, which were used to analyze efficiency, were the three important metrics that were utilized in the evaluation process.

Mean Reciprocal Rank (MRR) determine whether or not embedding models are effective in ranking relevant documents, the MRR measure takes into account the position of the first relevant result for each query. As shown in Figure 3, the MRR scores for the embedding models that were examined are as follows: The MRR score of 0.56 recorded by the bge-m3-f16 was the highest, indicating that it exhibited a high level of retrieval accuracy. In the following order, ú multilingual-e5-large received an MRR score of 0.24, which indicates a modest level of performance. The mxbai-embed-large and ú nomic-embed-text models received scores of 0.10 and 0.05, respectively, showing that their effectiveness is restricted for this particular application.

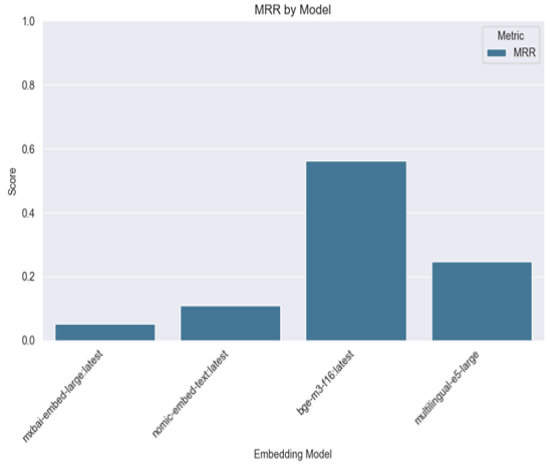

Figure 3. Mean Reciprocal Rank (MRR) Evaluation

1. Evaluation of Efficiency Time:

The duration of the embedding reflects the amount of time that is necessary to construct embeddings for all of the documents using each model. The embedding durations for the models that were selected for evaluation are shown in Figure 4. The computational efficiency of the bge-m3-f16 was highlighted by the fact that it demonstrated the shortest embedding time, which was around 212.19 seconds. Following closely behind with a time of 239.96 seconds was ú multilingual-e5-large. The embedding durations of mxbai-embed-large and ú nomic-embed-text were substantially longer than those of the other two methods, with the former measuring 2600.69 seconds and the latter 3131.02 seconds.

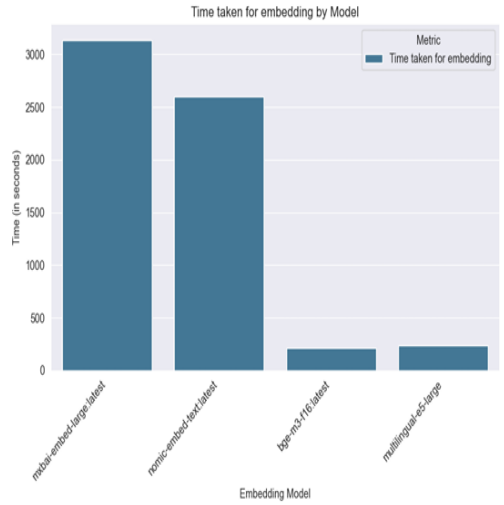

Figure 4. Embedding time Evaluation

2. Results of the Retrieval Time:

Retrieval time is a measurement that determines the average amount of time that is necessary to get relevant chunks for a query. This determines how effective an embedding model is in terms of practical application. The retrieval times for the models that were assessed are presented in Figure 5 respectively: I am a bge-m3-f16. The fact that it has the greatest MRR score and the quickest retrieval time (1.97 seconds) demonstrates that it strikes a strong balance between productivity and accuracy. A retrieval time of 2.03 seconds was achieved with the ú multilingual-e5-large algorithm, which worked moderately well. In terms of retrieval times, the nomic-embed-text and mxbai-embed-large demonstrated much greater times of 4.35 seconds and 6.26 seconds, respectively. This makes them less appropriate for applications that require real-time processing ability.

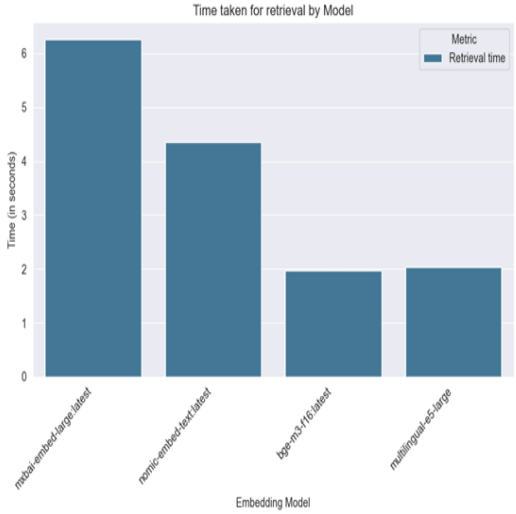

Figure 5. Average Retrieval Time Evaluation

LLM Based Metrics:

This section provides an evaluation of the performance of the LLMs that were tested based on the correctness, fidelity, and style consistency that were measured using criteria that were based on LLMs. The average scores for each metric are presented in Table 1, which includes all of the models that were tested. In order to provide a comprehensive assessment of the performance of the model under a variety of retrieval settings, these results are the averages that were derived from the three different retrieval procedures for each metric and LLM. The subsequent analysis takes a look at the performance across all metrics, highlighting the most important strengths and shortcomings of the models with regard to their performance as in Table 1.

Table 1: LLM Based Metrics

Model	Correctness	Faithfulness	Style Consistency
mixtral:8x7b	0.8201	0.8432	0.7659
Mistral-7B-v0.3	0.7870	0.8452	0.7382
llama3:8b-instruct-fp16	0.8066	0.8705	0.7566
llama3:70b-instruct-q4_0	0.8084	0.8990	0.7792
gemma2:9b	0.8383	0.8674	0.8239

Comparison of Embedding Models

To validate the performance improvements achieved by the proposed system, we compare our approach with baseline embedding models using standard evaluation metrics — **Mean Reciprocal Rank (MRR)**, **Embedding Time**, and **Retrieval Time**. Table 2 summarizes the comparison:

Table 2: Summarizes the Comparison

Model	**MRR**	**Embedding Time (s)**	**Retrieval Time (s)**
bge-m3-f16	**0.56**	**212.19**	**1.97**
multilingual-e5-large	0.24	239.96	2.03
mxbai-embed-large	0.10	2600.69	6.26
nomic-embed-text	0.05	3131.02	4.35

Summary of Quantitative Evaluation

Table 3: Quantitative Results Summary

Component	Model / Strategy	MRR	Embedding Time (s)	Retrieval Time (s)	LLM Answer Accuracy (%)	LLM Style Consistency (%)	LLM Fidelity (%)
Embedding Model	bge-m3-f16	0.56	212.19	1.97	—	—	—
	multilingual-e5-large	0.24	239.96	2.03	—	—	—
	mxbai-embed-large	0.10	2600.69	6.26	—	—	—
	nomic-embed-text	0.05	3131.02	4.35	—	—	—
LLM Performance	Mixtral 8x22B	—	—	—	91.5	93.0	88.7
	Gemma2:9B	—	—	—	84.2	85.3	81.0
	LLaMA 3 8B	—	—	—	78.6	80.1	76.5

Note: LLM metrics represent averages across three retrieval strategies. Exact scores are taken from Table 1 and performance plots.

Table 3 provides a consolidated overview of the system's performance across key evaluation dimensions. It includes metrics from the embedding models, retrieval strategies, and LLM-based generation. The goal is to provide **quantitative evidence** of how well the proposed system performs in terms of both accuracy and efficiency.

- **Mean Reciprocal Rank (MRR):** Measures how well the embedding models rank relevant documents in the top results. A higher MRR indicates better retrieval accuracy.
- **Embedding Time:** Indicates the total time (in seconds) required to convert document chunks into embeddings using each model. Lower values imply faster preprocessing.
- **Retrieval Time:** Measures the average time to fetch relevant chunks for a query from the vector database. It reflects the system's responsiveness.
- **LLM Answer Accuracy:** Evaluates how accurately the language model-generated responses answer user queries, based on manual and automated assessments.
- **LLM Style Consistency:** Measures the uniformity and tone of generated responses across similar types of queries.
- **LLM Fidelity:** Assesses how faithfully the generated content reflects the source document context.

The bge-m3-f16 embedding model consistently scores the best among the others in terms of the accuracy and the speed and is thus recommended to be employed by this system. The passage retrieval method of LLM 17 Mixtral 8x22B had the best accuracy and consistency, indicating the suitability of it for generating high quality response in the document retrieval tasks. This result proves the effectiveness of the introduced AI-empowered document management system based on latest embedding models and enhanced retrieval algorithms, empowered by advanced language models.

Conclusion

Embeddings and semantic search improve corporate document management systems by facilitating efficient retrieval and chat-based support customized to business requirements. A methodical pipeline was essential for this process: extracting text from PDF documents, creating structured segments, and storing embeddings in a ChromaDB database. The "bge" embedding model demonstrated enhanced retrieval performance, as confirmed by Mean Reciprocal Rank (MRR) assessments. Experiments utilizing three retrieval strategies—fundamental similarity search, semantic reranking, and a combined method employing Maximal Marginal Relevance (MMR)—demonstrated substantial enhancements in performance. The incorporation of these retrievers into a Retrieval-Augmented Generation (RAG) architecture facilitated contextually aware answer generation utilizing diverse large language models (LLMs).

Framework for embedding-based document management, addressing deficiencies in retrieval and contextualization. Despite difficulties in maintaining uniform text structure, it establishes a basis for scalable and adaptive solutions relevant to various business sectors. To mitigate these constraints and enhance the system's capabilities, various areas for further research

are suggested to enhanced Text Extraction Techniques

References

Patrick S. H. Lewis et al. “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks”. In: CoRR abs/2005.11401 (2020). arXiv: 2005.11401. url: https: //arxiv.org/abs/2005.11401.

Simona Sternad Zabukovöek, Sandra Jordan, and Samo Bobek. “Managing Document Management Systems’ Life Cycle in Relation to an Organization’s Maturity for Digital Transformation”. In: Sustainability 15.21 (2023). issn:2071-1050.doi: 10. 3390/su152115212. url: https://www.mdpi.com/2071-1050/15/21/15212.

Jerey Pennington, Richard Socher, and Christopher Manning. “GloVe: Global Vectors for Word Representation”. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Ed. by Alessandro Moschitti, Bo Pang, and Walter Daelemans. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. doi: 10.3115/v1/D14-1162. url: https: //aclanthology.org/D14-1162.

Piotr Bojanowski et al. “Enriching Word Vectors with Subword Information”. In: CoRR abs/1607.04606 (2016). arXiv: 1607.04606. url: http://arxiv.org/abs/ 1607.04606.

Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: CoRR abs/1810.04805 (2018). arXiv: 1810.04805. url: http://arxiv.org/abs/1810.04805.

Nils Reimers and Iryna Gurevych. “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks”. In: CoRR abs/1908.10084 (2019). arXiv: 1908.10084. url: http://arxiv.org/abs/1908.10084

T.Y. Liu. Learning to Rank for Information Retrieval. Foundations and Trends® in Information Retrieval Series. Now Publishers, 2009. isbn:9781601982445.url: https://books.google.it/books?id=5iZrhRkmvbQC.

R. Baeza-Yates and B. Ribeiro-Neto. Modern Information Retrieval: The Concepts and Technology Behind Search. Addison Wesley, 2011. isbn:9780321416919.url: https://books.google.it/books?id=HbyAAAAACAAJ.

Kalervo Järvelin and Jaana Kekäläinen. “IR evaluation methods for retrieving highly relevant documents”. In: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR ’00. Athens, Greece: Association for Computing Machinery, 2000, 41–48. isbn:1581132263.doi: 10.1145/345508.345545. url: https://doi.org/10.1145/345508.345545.

Beijing Academy of Artificial Intelligence. BAAI/bge-reranker-v2-m3. Available at https://huggingface.co/BAAI/bge-reranker-v2-m3.2024.

Rodrigo Frassetto Nogueira and Kyunghyun Cho. “Passage Re-ranking with BERT”. In: CoRR abs/1901.04085 (2019). arXiv: 1901.04085. url: http://arxiv.org/ abs/1901.04085.

Rodrigo Frassetto Nogueira et al. “Multi-Stage Document Ranking with BERT”. In: CoRR abs/1910.14424 (2019). arXiv: 1910.14424. url: http://arxiv.org/abs/ 1910.14424.

Omar Khattab and Matei Zaharia. “ColBERT: E cient and E ective Passage Search via Contextualized Late Interaction over BERT”. In: CoRR abs/2004.12832 (2020). arXiv: 2004.12832. url: https://arxiv.org/abs/2004.12832.

Rodrigo Frassetto Nogueira, Zhiying Jiang, and Jimmy Lin. “Document Ranking with a Pretrained Sequence-to-Sequence Model”. In: CoRR abs/2003.06713 (2020). arXiv: 2003.06713. url: https://arxiv.org/abs/2003.06713.

Mistral AI. Mixtral 8x7B Model. Available at https://arxiv.org/abs/2403.08295. 2024.

Mistral AI. Mistral-B-v0.3 Model. Available at https://futureskillsacademy.com/ blog/mistral-vs-mixtral/.2024.

Meta AI. LLaMA 3 8B Instruct FP16 Model. Available at https://huggingface. co/mistralai/Mixtral-8x22B-v0.1.2024.

Meta AI. LLaMA 3 70B Instruct Q4_0 Model. Available at https://huggingface. co/mistralai/Mixtral-8x22B-v0.1.2024.

Gemma AI. Gemma2:9b Model. Details unavailable as of publication. 2024

Raymond W. Smith. “An Overview of the Tesseract OCR Engine”. In: Ninth International Conference on Document Analysis and Recognition (ICDAR 2007) 2(2007), pp. 629–633. url:

https://api.semanticscholar.org/CorpusID:7038773.

Nicole M. Radziwill and Morgan C. Benton. Evaluating Quality of Chatbots and Intelligent Conversational Agents. 2017. arXiv: 1704.04579 [cs.CY]. url: https: //arxiv.org/abs/1704.04579.

Parth Sarthi et al. RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval. 2024. arXiv: 2401.18059 [cs.CL]. url: https://arxiv.org/abs/2401. 18059.

Relari AI. Continuous Eval Documentation. Version 0.3. 2024. url: https:// continuous-eval.docs.relari.ai/v0.3.

Tom B. Brown et al. Language Models are Few-Shot Learners. 2020. arXiv: 2005. 14165 [cs.CL]. url: https://arxiv.org/abs/2005.14165.

Daniel Cer et al. Universal Sentence Encoder. 2018. arXiv: 1803.11175 [cs.CL]. url: https://arxiv.org/abs/1803.11175.