

# **Large-Scale PDF Analysis Leveraging**

## **RAG and LLMs for Enhanced ML**

### **Literature Insights**

A Project Report  
Presented to

The Faculty of the Department of Applied Data Science  
San Jose State University  
In Partial Fulfillment  
Of the Requirements for the Degree  
Master of Science in Data Analytics

By

*Badrinarayanan, Shruti*

*Neelamegam, Divya*

*Suresh Kumar, Sourabh*

*Venkatram, Poojitha*

*December 07, 2024*

*Copyright © 2024*

*Divya Neelamegam, Shruti Badrinarayanan, Sourabh Suresh Kumar,  
Poojitha Venkatram  
ALL RIGHTS RESERVED*

**APPROVED FOR DEPARTMENT OF APPLIED DATA SCIENCE**

---

**Dr. Simon Shim, Project Advisor**

---

**Dr. Lee C. Chang, Project Advisor**

## ABSTRACT

*Large-Scale PDF Analysis Leveraging RAG and LLMs  
for Enhanced ML Literature Insights*

By

*Divya Neelamegam, Shruti Badrinarayanan, Sourabh  
Suresh Kumar, Poojitha Venkatram*

This study introduces an advanced Scientific Literature Review Assistant aimed at streamlining information retrieval and response generation for researchers, students, and academicians, thereby democratizing access to knowledge. By leveraging state-of-the-art Retrieval-Augmented Generation (RAG) pipelines, the system processes vast datasets, including arXiv and JMLR papers, using cutting-edge techniques such as chunking, preprocessing, and embedding via transformer-based models. At its core, the FAISS vector database ensures efficient storage and retrieval, while an integrated search engine empowers users to discover similar research papers based on content or abstract similarity.

The assistant utilizes a suite of specialized RAG models: the Adaptive RAG model employs routing and fallback mechanisms to guarantee precise retrieval using a fine-tuned Mistral Model; Super RAG integrates data augmentation and knowledge enhancement for contextual depth; and Multimodal RAG accommodates diverse data formats, including text and images, for holistic query resolution. A Hybrid RAG model combines semantic search, BM25-based methods, and external APIs to address hallucinations or gaps in context. Additionally, the Local RAG Agent incorporates Self-RAG, Corrective RAG, and Adaptive RAG techniques to ensure refined context retrieval, self-reflection, and error correction.

The outcomes are underpinned by measurable technical merits such as improved response generation accuracy, efficient query handling, and effective integration of tables within conversations maintaining history. Quality evaluation involves assessing response relevance, coherence, and completeness through user feedback and rigorous evaluation metrics to assess our model using evaluation metrics such as context\_relevancy, context\_recall for retrieving all relevant information, faithfulness for factual accuracy, and answer\_relevancy for pertinence. The results indicate that Adaptive RAG and the RAG Agent with the Combined Approach consistently outperform others across key metrics, demonstrating superior performance in context relevance, answer relevance, and correctness, making them the most effective overall solutions. The system performance is also evaluated using latency and throughput metrics, ensuring that the models deliver timely and efficient responses. Additionally, CPU and GPU utilization are monitored to optimize resource usage and maintain scalability under varying workloads, contributing to the overall effectiveness and robustness of the system. The analysis reveals that Multimodal RAG excels in response time, ensuring faster query resolution, while the RAG Agent with Combined Approaches optimizes GPU usage, making it a resource-efficient choice for high-performance tasks.

Designed with a user-friendly interface using Gradio and Streamlit, the assistant enables seamless PDF uploads and interactive query handling. Researchers, students, and academicians can efficiently explore relevant literature using the integrated search engine, which identifies similar research papers based on content or abstract similarity, aiding in targeted and relevant discovery. Supported by a comprehensive evaluation framework to enhance accuracy, coherence, and relevance, this assistant serves as a powerful tool to facilitate knowledge and democratize access to cutting-edge academic insights.

## ACKNOWLEDGMENTS

We would like to extend our sincere gratitude to all those who contributed to the successful completion of this project. First and foremost, we express our deepest appreciation to our professors, academic advisors, and mentors for their invaluable guidance, insights, and constructive feedback, which shaped the direction and depth of this study.

We are profoundly grateful to the Department of Applied Data Science at San José State University (SJSU) for providing access to computational resources, including high-performance GPUs and cloud platforms. These resources were instrumental in the efficient implementation and testing of our main vector database and Retrieval-Augmented Generation (RAG) pipeline.

Our heartfelt thanks go to the authors and contributors of the arXiv and JMLR datasets, whose work laid the foundation for our research. We also acknowledge the researchers whose studies on Adaptive RAG, Corrective Retrieval-Augmented Generation, Self-RAG, and Super RAG provided key insights that significantly influenced the design and implementation of our pipeline. Furthermore, we are thankful for access to ongoing advancements in large language models (LLMs) and artificial intelligence, which provided critical frameworks and benchmarks for innovation. We would also like to recognize the support of our teammates and peers, whose collaborations enriched our understanding and offered fresh perspectives on the challenges we encountered. Finally, we express our heartfelt gratitude to our families and friends for their unwavering encouragement and understanding throughout this journey. Their support made this work possible.

This report represents the culmination of collective efforts and contributions from many individuals, and we deeply appreciate each and every one of you. Thank You.

## **TABLE OF CONTENTS**

### **Chapter 1 Introduction**

- 1.1 Project Background and Executive Summary
- 1.2 Project Requirements
- 1.3 Project Deliverables
- 1.4 Technology and Solution Survey
- 1.5 Literature Survey of Existing Research

### **Chapter 2 Data and Project Management Plan**

- 2.1 Data Management Plan
- 2.2 Project Development Methodology
- 2.3 Project Organization Plan
- 2.4 Project Resource Requirements and Plan
- 2.5 Project Schedule

### **Chapter 3 Data Engineering**

- 3.1 Data Process
- 3.2 Data Collection
- 3.3 Data Pre-processing
- 3.4 Data Transformation
- 3.5 Data Preparation
- 3.6 Data Statistics
- 3.7 Data Analytics Results

### **Chapter 4 Model Development**

- 4.1 Model Proposals
- 4.2 Model Supports
- 4.3 Model Comparison and Justification
- 4.4 Model Evaluation Methods
- 4.5 Model Validation and Evaluation Results

### **Chapter 5 Data Analytics System**

- 5.1 System Requirements Analysis
- 5.2 System Design
- 5.3 Intelligent Solution
- 5.4 System Development and Implementation

### **Chapter 6 System Evaluation and Visualization**

- 6.1 Analysis of Model Execution and Evaluation Results
- 6.2 Achievements and Constraints
- 6.3 Quality Evaluation of Model Functions and Performance
- 6.4 Evaluation of Models vs. Requirements
- 6.5 Project Information Visualization

## **Chapter 7 Conclusion**

- 7.1 Summary
- 7.2 Benefits and Shortcoming
- 7.3 Potential System and Model Applications
- 7.4 Experience and Lessons Learned
- 7.5 Recommendations for Future Work
- 7.6 Contributions and Impacts on Society

## **References**

## **Appendices**

**Appendix A – System Testing**

**Appendix B – Project Data Source and Management Store**

**Appendix C – Project Program Source Library, Presentation, and Demonstration**

## **1. Introduction**

### **1.1. Project Background and Execute Summary**

#### ***Project Background***

The field of Machine Learning and AI is advancing rapidly with continuous research and breakthroughs. Consequently, researchers, students, and professionals encounter difficulties in staying updated with the latest developments and comprehending dense technological information. ML research literature often involves professional and sophisticated concepts and advanced mathematical formulas, causing difficulty for people without extensive knowledge of the topic. This intricacy poses a barrier to entry for newcomers and can limit efficient knowledge transfer and collaboration within the ML community. Conventional procedures of literature review and research discovery can be time-consuming and inefficient, requiring extensive manual work to browse and filter through huge volumes of literature and identify relevant studies.

#### ***Needs and Importance***

There is a major requirement for tools and platforms that simplify the accessibility and comprehension of literature related to ML and AI. Such forums can enable information sharing, and technical disputes and stimulate interdisciplinary collaboration among academics, students, and professionals in the field of data science. There is a need for creative solutions that streamline these procedures to save time and effort for academics and practitioners navigating the fast-evolving data science field. The recommended solution seeks to leverage adaptive conversational skills to address the aforementioned issues. By mixing natural language processing (NLP) techniques and deep learning, this project attempts to develop a conversational

interface that adapts to users' knowledge levels and learning preferences, thereby facilitating comprehension of complicated technical concepts engagingly and intuitively.

While initially focused on ML and AI literature, the adaptive conversational skills built as part of this study provide potential for extension beyond the subject of ML and AI. Similar applications may be envisioned in other academic and professional areas requiring interpretation of sophisticated technological material, hence magnifying the effect and significance of the initiative outside its initial domain.

### ***Goals and Motivation***

The primary goal of the project is to assist researchers, students, and professionals realize subtle and latest breakthroughs in the fast-developing field of Machine Learning literature. As the discipline of machine learning rapidly increases, people have problems with keeping up with the most recent developments and comprehending significant technical jargon. The major purpose of this initiative is to increase the accessibility and understanding of research articles related to ML/AI and encourage individuals to expand their knowledge by traversing unfamiliar content, uncovering new research subjects, and collaborating more efficiently. This project aims to:

1. Identify the best combination for building RAG with respect to chunks, text splitters, embedding models, and vector stores for effective performance.
2. Develop a chat application using different RAG techniques integrated with various LLMs to interpret and respond to inquiries based on research papers PDFs uploaded by users.
3. Ability to parse the non-text contents in the PDFs (multimodal content).
4. Capable of providing informed responses to technical questions by referencing a comprehensive database of 25000+ research papers.

5. ML/AI Research Paper Search Engine: Assist in identifying existing literature on specific topics within machine learning and artificial intelligence.

### ***Project Approaches and Methods***

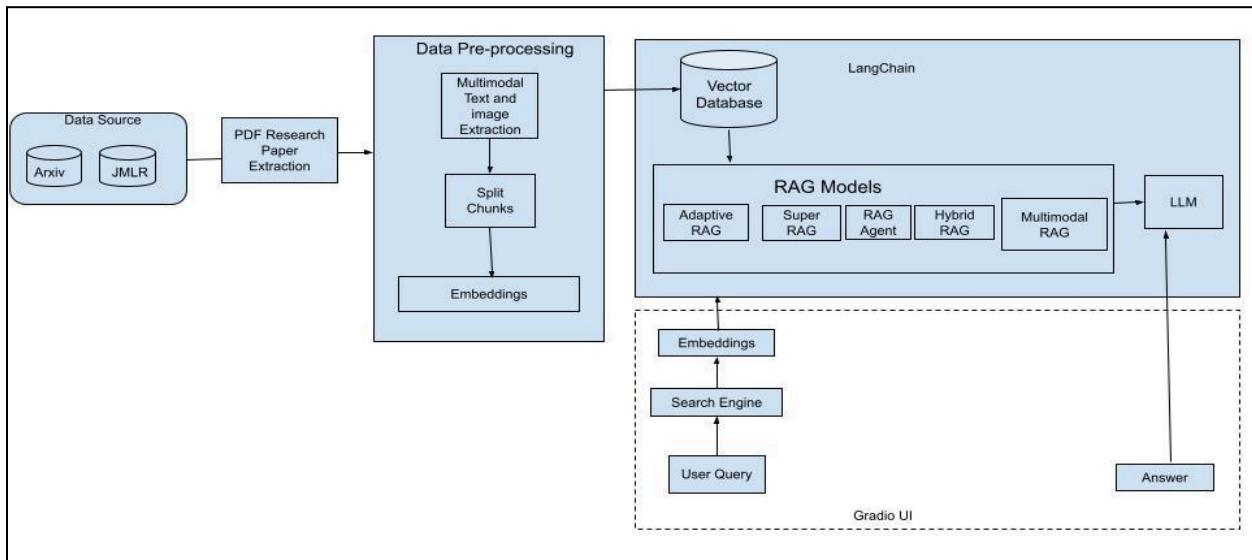
The core technique of our system is depicted in Figure 1 which comprises a Retriever-Augmented Generation (RAG) model that integrates with the LLAMA Large Language Model (LLM). Additionally, Langchain, which serves as a framework to connect language models with external information sources, is also a component of our approach.

The data sourcing method is based on the acquisition and consolidation of research articles from a variety of credible databases, such as Arxiv and the Journal of Machine Learning Research (JMLR). These databases are the key resources for our application's knowledge base. The digital content in the PDF is divided into smaller chunks during the preprocessing phase and transformed into embeddings suitable for computational processing and similarity analysis within our vector database. The retrieval component is the combination of the advantages of adaptive-RAG and self-RAG. The Adaptive RAG routes the questions to different retrieval approaches based on the retrieval grader. The retriever fetches the document context from the knowledge base and grades the context based on its relevance to the question. If the context is not relevant, the Corrective RAG works as a fallback to retrieve answers from the web. The self-RAG takes care of fixing the hallucination, if any, from LLM answers before providing the response to the user. The advantage of Adaptive RAG with fallback and self-RAG makes the retrieval mechanism more powerful and efficient. Also, Langchain serves as a key intermediary, enabling our language model to not only generate responses based on internal knowledge but also query external databases and sources, increasing the quality and scope of the answers. The language model then synthesizes the input to produce accurate and coherent answers.

**Figure 1**

*Proposed Model Workflow*

*(Original from Team 8)*



### ***Expected Project Contributions and Applications***

This project is assured to make significant improvements to the field of academic research facilitation, notably in the domains of machine learning and artificial intelligence. Developing a chat application that responds to queries based on user-uploaded PDF research papers is a remarkable breakthrough. A major component of this application is its ability to parse and interpret textual and multimodal content within these documents. This comprehensive parsing capability ensures that valuable data embedded in tables enhances the application's utility.

By utilising a vast collection of research publications, the application aims to deliver well-informed replies to technical inquiries. This application serves both the academic community seeking quick clarifications and the larger audience seeking to understand complicated technical topics.

This project provides a baseline for building effective RAG combining the advantages and the application works as a research aid, capable of dissecting thick academic material and offering succinct, correct summaries of study findings and procedures. It is notably useful for identifying existing literature on specific issues within the large disciplines of ML and AI, hence streamlining the literature review process which is normally time-consuming and labor-intensive. Additionally, for experts and practitioners in the industry, the chat application offers a speedy reference tool to aid in decision-making and maintaining current with the newest research findings.

## **1.2. Project Requirements**

### *Functional Requirements*

The "Large-Scale PDF Analysis Leveraging RAG and LLMs for Enhanced ML Literature Insights" project stands out for its essential functional features, which are created to transform user interaction with machine learning literature via an innovative chat interface. A sophisticated PDF Upload and Processing System, designed to make user submissions of PDF documents simple, lies at the heart of the project. This technology performs exceptionally well at accurately extracting text, images, and tables, guaranteeing that the content is fully understood. This feature facilitates an interactive and educative dialogue between the user and the chat system by utilizing the most recent advancements in the field of Natural Language Processing (NLP) and Generative AI. These features are essential for thoroughly extracting and analyzing data from PDF documents, which improves the user's educational experience.

## ***PDF Upload and Processing System***

A user-friendly interface will be provided for uploading PDF documents, enhancing access to ML literature. The system will accurately extract text, and tables from PDFs, ensuring a comprehensive understanding of the documents.

## ***Adaptive Conversational Interface***

Using content that has been extracted from PDF files, an interactive chat interface will facilitate dynamic conversations catered to the users' learning goals. To promote meaningful interactions, this interface will modify responses based on the context and substance of user inquiries by processing data from PDF.

## ***LLM-RAG Accuracy Assessment***

The system is going to evaluate the precision and contextual applicability of its answers to make sure they correspond with user requests concerning PDF documents. Validating the quality of the information presented will involve comparisons with facts through rigorous analysis.

## ***Human Relevance Review***

The system-generated responses will go through a review process to evaluate their clarity and relevancy and make sure they live up to user expectations. Through this procedure, the system's output will be improved for greater user comprehension.

## ***Retrieval Effectiveness Evaluation***

The system's responses will be compared using quantitative metrics to gauge the correctness of the answers generated. The purpose of this benchmarking is to unbiasedly assess how well the system performs in producing precise answers.

Using both human judgment and ground-truth datasets, the system's efficacy in selecting pertinent scientific papers will be assessed. Through this evaluation, the system's ability to precisely find and suggest relevant material is confirmed, improving study findings.

### ***AI Requirements***

#### ***Large Language Models (LLMs)***

The project uses LLMs to produce context-aware responses in discussions about machine learning literature, with LLaMa 3, Gemma, GPT-4, Mistral, etc. With the help of these models' sophisticated capabilities, responses that precisely match user inquiries and uploaded PDFs can be generated with a deep understanding along with insights retrieved from the main vector database which is built using a dataset collected from eminent research sites like arXiv and JMLR. Through this process, the application becomes more proficient at navigating and understanding complex machine-learning concepts.

#### ***The Role of Hybrid Retriever-Augmenter-Generator Model (Base)***

Using a hybrid approach that combines keyword-based and semantic search approaches is a fundamental functional need for a Retrieval-Augmenter-Generator (RAG) model. It is the job of this model to use the rankings of the LLAMA LLM to first select the top K relevant passages. Then, adding these chosen paragraphs to the LLM, will improve the model's context and improve the model's ability to precisely respond. The LLM uses this richer context at the end of the process to create responses that specifically address the questions posed by the users. For the complex interaction dynamics this project envisions, this method guarantees the delivery of thorough and contextually aware responses, which is a technological requirement.

### ***Vector Storage for Efficient Information Retrieval***

Large Language Models (LLMs) and the Retrieval-Augmenter-Generators (RAG) rely heavily on the development of vector storage techniques for their operation. An extensive library of research papers requires the management of high-dimensional data, and this technique is essential for doing so. This technique is essentially about turning text in PDFs and research articles into numerical embeddings so that quick indexing and retrieval may occur. This type of data vectorization guarantees that the system can effectively sort through large amounts of information and identify the most relevant details in direct response to user inquiries. This vectorized data storage system is a crucial component of the project since it provides the framework for dynamic retrieval capabilities, which are necessary to keep a conversational interface that is both knowledgeable and responsive. Different vector storage methods will be evaluated and tested and the most efficient technique will be selected.

### ***Continuous Learning and System Adaptation***

One novel feature of the project is the use of continuous learning approaches, which employ user feedback in real-time to iteratively improve the performance of the model. By retaining the accuracy and relevancy of the information offered, this dynamic adaptation makes sure that the system is responsive to the changing demands and questions of its users.

### ***Data Requirements***

For the growth and accessibility of machine learning (ML) research, it is essential to emphasize the availability of a broad range of ML and scientific literature in PDF format, including books, articles, conference papers, and journals. The project's goal is to improve the comprehension, exchange of ideas, and application of sophisticated machine learning concepts

and results among data science researchers, students, and professionals. This thorough access supports this goal.

To provide accurate, context-aware responses, the project's main goal is to create a conversational interface that processes and responds to queries using a vast repository of machine learning and scientific documents. This goal is supported by the dual requirements of having access to a wide variety of literature formats and having a strong, diverse knowledge base spanning machine learning and related scientific disciplines. By utilizing technologies such as the hybrid Retriever-Augmented Generation (RAG) model and LLMs the project seeks to use the rich content of these documents—which includes non-text elements like tables —to provide a comprehensive examination of complex machine learning literature.

This methodology highlights the technical ambition to navigate the rapidly evolving machine learning research domain by preprocessing literature into manageable chunks, generating embeddings, and integrating with vector databases for efficient retrieval. It promises improved access to and interaction with scientific literature and lays the groundwork for similar advancements across other academic and professional fields.

Specifically, a variety of text and tables taken from scientific articles acquired from the Journal of Machine Learning Research (JMLR) and arXiv are included in the datasets used for training and testing NLP models inside the "Large-Scale PDF Analysis Leveraging RAG and LLMs for Enhanced ML Literature " project. These repositories were selected because of their substantial holdings of scientific and machine learning literature, which include in-depth studies, preprints, and peer-reviewed papers from a variety of ML domains and related subjects.

### ***JMLR Dataset***

With 2690 papers carefully curated and enhanced with unique features that enable a comprehensive exploration of machine learning research, the Journal of Machine Learning Research (JMLR) dataset is a resource for researchers.

### ***arXiv Dataset***

The dataset consists of over 1.7 million scientific papers in the STEM (Science, Technology, Engineering, and Mathematics) domains.

Annotated datasets with basic processing tasks like named entity recognition, sentence segmentation, tokenization, and semantic analysis—all customized to the subtleties of scientific texts—are the foundation for the construction of natural language processing (NLP) models. With careful annotation, models are guaranteed to be capable of deciphering and interpreting the complex vocabulary and formatting of machine learning and scientific reports.

Ground truth datasets, which comprise user inquiries linked with expert replies or verified data, are used to assess the accuracy of the LLM-RAG model within the project. These datasets show the system's proficiency in answering real-world questions relating to machine learning and scientific literature by providing a standard against which to evaluate its ability to produce accurate and contextually relevant responses. The project may also thoroughly test and improve the system's adaptability and performance in conversational scenarios thanks to collections of user inquiries and expert-validated answers. To enable deeper engagement with complicated scientific materials, this technique guarantees that the conversational interface not only meets but surpasses expectations for accuracy, relevance, and value.

### 1.3. Project Deliverables

This section outlines the core deliverables for the ML-Chat application (where users can upload PDFs to address queries) and a scientific review assistant system, incorporating the functionalities described in previous sections and additional details for evaluation and additional considerations. The team is utilizing Asana which is a project management tool, crucial for tracking project deliverables, and maintaining a clear timeline. A snapshot of the team's Asana board is attached under Figure 2. The main project deliverables are highlighted in the following subsections.

**Figure 2**

*Asana to track Project Deliverables*

*(Original from Team 8)*

To-Do	In Progress	Completed	Review
7	5	21	2

#### **Data Source**

1. Develop a system/script to ingest data from various sources such as JMLR, and arXiv to build a vector database. This pipeline is set up on Google Cloud Platform

2. PDF Uploader: Design a user interface component that allows users to upload academic PDFs for analysis.
3. Research Paper Database Integration: Implement web scraping or API integration capabilities to connect with online databases of scientific publications (arXiv, PubMed Central, JMLR). Consider data licensing agreements and ethical considerations when scraping data.

### ***Data Preprocessing***

1. PDF Extraction: Develop a module to extract text and relevant information from uploaded PDFs using Natural Language Processing (NLP) techniques. This involves Layout Analysis (Identify sections like title, abstract, introduction, methods, results, conclusion, etc) and Citation Parsing (Extract and format citation information for reference management).
2. Splitting and Chunking: Split the extracted text from PDFs and research papers into smaller, manageable segments (sentences or paragraphs) for efficient processing during the embedding stage.

### ***Embeddings***

Text and Multimodal Representation: Generate numerical representations (embeddings) of the preprocessed chunks. This allows the model to understand the semantic relationships between words and sentences.

Different embedding and extraction techniques to be explored:

1. Word Embeddings: Train word embedding models (e.g., Word2Vec, GloVe) on a large corpus of scientific literature to capture word meaning and relationships.

2. Sentence Embeddings: Utilize pre-trained sentence embedding models (e.g., Universal Sentence Encoder, InferSent) or train custom models on scientific text data to represent the meaning of entire sentences.
3. Table Transformers: For table information/text extraction which are further converted to embeddings.

### ***LLM-RAG for PDF User Queries***

1. User Interface: Design a user interface component where users can submit queries related to the uploaded PDF document.
2. Query Processing: Develop a module to process user queries, handled similarly to processing data from PDFs.
3. LLM for Candidate Passage Retrieval: Leverage large language models (LLMs) such as LLaMa-2, LLaMa-3, Gemma, Mistral with various Retrieval Augmented Generation techniques. They are used to identify relevant passages from the uploaded PDF that are most likely to answer the user's query and generate appropriate answers to user queries.

This may involve:

- Building Vector Store (Chroma, FAISS, etc) of collected Academia PDFs on AI and ML.
  - Compare different storage methods and settle on the most advantageous.
  - Passage Ranking: Employ the LLM to score each passage based on its relevance to the query, generating a ranked list of candidate passages for further processing.
4. RAG Aggregation: Implement various approaches of Retriever-Augmenter-Generation (RAG) to:
    - Retrieve: Select the top K most relevant passages from the LLM's ranked list.

- Augment: Leverage the retrieved passages to provide additional context to the LLM to improve its answer generation.
- Generate: Use the LLM to generate an answer to the user's query, drawing on the information from the retrieved and augmented passages.
- Different Approaches: Hybrid RAG, Adaptive RAG, Self-RAG, Super RAG, Fallback RAG.
- Route to web search if the document nor the vector store contains the required context.

### ***Scientific Review Assistant - Paper Search***

1. User Interface: Design a user-friendly interface for users to submit research queries. This includes text boxes, dropdown menus, or keyword selection functionalities for refined searches.
2. Query Processing: Develop a module to process the search topic entered by the user.
3. Matching Script/Algorithm: Implement a script to search the database of paper embeddings for papers that are most relevant to the user's query based on the embedded representations.
  - Explore different retrieval methods such as Nearest Neighbor Search where papers with embeddings closest to the user query embedding in the vector space are identified, and Semantic Search where the application leverages semantic similarity techniques to find papers that capture the underlying meaning of the user's query, even if the exact keywords aren't present.
  - Another option is to use the initial CSVs containing the features of the papers to search for relevant topics.

## ***Output Generation and User Interface***

1. Answer Generation:
  - Answer Format: Display the answer generated by the system in a clear and concise format, highlighting the most relevant information from the retrieved PDF passages to address the user's query.
  - Source Indication (Optional): Provide a reference section indicating the source passages within the PDF from which the answer was generated.
2. Scientific Review Assistant - Search Results:
  - Summarization: Generate summaries of the retrieved research papers to provide users with an informative overview of the relevant findings. Consider using extractive summarization techniques.
  - Ranking: Display the search results in order of relevance to the user's query, using factors such as similarity scores, publication date, and citation count.
3. User Interface: Using streamlit design functionalities to display the search results in a clear and informative way: Paper Titles and Authors, Brief Summaries, Links to the original papers (if possible), and Additional metadata (e.g., publication date, journal).

## ***Evaluation Framework***

Develop a robust evaluation framework to assess the performance of the system and the effectiveness of different retrieval and answer generation methodologies. This may involve:

1. LLM-RAG Evaluation: Assess the quality and accuracy of answers generated by the LLM-RAG system for PDF queries.

2. Human Assessment: Have all team members judge the relevance and clarity of generated answers to evaluate system performance on a specific set of pdfs which can be considered as a test or production set.
3. Automatic Metrics for Retrieval Evaluation: Use metrics to compare the generated answers to ground-truth answers (such as RAG triad metrics and Answer Correctness).
4. Check for LLM Hallucinations.
5. Qualitative Evaluation (if possible): Conduct user studies or surveys to gather feedback on the overall usability and usefulness of the system.

### ***Documentation***

Thorough documentation is essential for the project's maintainability and future development. This includes detailed technical documentation outlining system architecture, code implementation, model specifications, and API guides. Additionally, prepare clear user documentation to guide users in effectively interacting with the system and interpreting the results. This is a continuous process during the tenure of the project and documentation is expected to have major updates as the project progresses.

### ***Additional Considerations***

Beyond the core deliverables, several essential factors warrant attention for the success of this project. Incorporating user feedback mechanisms will allow for iterative improvement of both the retrieval models and answer generation components. Exploring continuous learning strategies will ensure that the system adapts over time, continually refining its performance. Responsible AI practices must be a priority to prevent the propagation of biases, maintain data privacy, and foster trust through transparent decision-making. Finally, the system architecture

should proactively address scalability, ensuring its ability to handle growing volumes of research papers and serve multiple users without degradation of performance.

#### 1.4. Technology and Solution Survey

A deep dive into the existing landscape of scientific review assistants revealed a mix of strengths and weaknesses among current solutions in the market. A more nuanced breakdown of our primary competitors and similar solutions can be seen in Table 1.

**Table 1**

*Understanding the Competitive Landscape* *(Original from Team 8)*

Solution	Description	Technologies & Approaches	Strengths	Weaknesses
<i>Semantic Scholar</i>	AI-powered scientific search engine	Semantic search, knowledge graphs, citation analysis	Concept-based search, author profiles, visualizations	Limited question-answering on PDFs, may use older embedding technology
<i>TypesetQuery / SciSpace</i>	PDF analysis, scientific paper search, summarization	PDF processing with custom tools, possibly some semantic search, potentially limited LLM integration	Direct competitor in the space	Less advanced question-answering than a full LLM-RAG implementation
<i>Elicit</i>	Research tools using LLMs to extract insights from papers	Focus on LLM for question-answering, likely on pre-specified papers	Potential for in-depth answers	Might lack comprehensive search engine features

As seen in Table 1, while some solutions offer scientific paper search or even summarization, they often lack the in-depth, PDF-centric question-answering capabilities this project aims to provide. This is where we can make a significant contribution. By leveraging state-of-the-art LLM and RAG techniques and high-quality scientific embeddings, we can offer

researchers a more powerful tool to directly extract insights from their uploaded research materials/topic.

### ***Open-Source Frameworks, Commercial APIs & Our Strategy***

Open-source and commercial AI solutions present additional opportunities for strategically building our system. We need to stay up-to-date with the rapid advancements in the field to make the most informed choices:

1. Haystack: Its modularity and adaptability to various question-answering paradigms align well with our goals. We should strongly consider this as a foundation or for custom components. Haystack's flexibility allows us to integrate different retrieval methods and potentially even explore multiple LLM models in the future.
2. LLaMA (and similar): Meta AI's recent release of LLaMA (LLaMa 3), a foundational language model, opens up exciting possibilities. While not specifically trained in scientific text, its potential for adaptation and customization is enormous. We will explore LLaMA and its performance on queries related to scientific papers to potentially improve our system's question-answering capabilities.
3. Other Emerging LLMs: The field of large language models is exploding. It's crucial to monitor new developments and potentially benchmark smaller, more efficient LLMs that might be better suited for specific tasks within our system.
4. Cloud NLP APIs: These are valuable for rapid prototyping, comparing providers, or selectively using bleeding-edge features. However, we need to assess cost and vendor lock-in trade-offs. Cloud-based NLP APIs can help experiment with new features or leverage specific functionalities that might be too resource-intensive to build in-house. However, we need to be mindful of potential costs and the risk of becoming reliant on a

single vendor. OpenAI is the most evolving API now, and tutorials often become out of date as well within weeks instead of what used to take months to years.

Table 2 is a breakdown of Feature Comparison between solutions in the market and the potential merits of our project.

**Table 2**

*Feature Comparison: Our Vision* *(Original from Team 8)*

Feature	Semantic Scholar	Typeset / SciSpace	Elicit	Haystack	Our Project
<i>Scientific Paper Search</i>	✓	✓	Limited	Customizable	✓ (Must be robust)
<i>PDF Upload &amp; Processing</i>	Limited	✓	–	Customizable	✓ (Thorough & Accurate)
<i>PDF Question-Answering</i>	Limited	Potentially Limited	✓	Highly Customizable	✓ Aim for advanced RAG approaches (Scientifically Rigorous)
<i>Summarization</i>	✓	✓	✓	Customizable	✓ (High-Quality & Domain-Specific)

### ***The Evolving LLM Landscape***

GPT-3.5 Turbo and GPT-4 are both trained on extensive datasets, enabling a deep understanding of natural language which makes them exceptionally adept at generating coherent and meaningful responses. This foundational capability allows these Large Language Models (LLMs) to address a broad spectrum of queries, providing highly relevant answers to various research-related inquiries. Their ability to parse complex questions and offer insightful responses makes them invaluable tools for applications that aim to facilitate access to and understanding of dense academic material. However, despite the advantages offered by integrating Retriever-Augmented Generation (RAG) alongside these LLMs, there are notable limitations to consider. Specifically, as an earlier iteration, GPT-3.5 may lag behind GPT-4 and newer models

in interpreting nuanced questions or generating complex, detailed responses. Moreover, GPT-3.5 might not fully incorporate the most current linguistic structures or technical terminologies that have emerged since its last training update. Additionally, deploying GPT-3.5 within an RAG framework could necessitate substantial memory and computational resources, potentially posing challenges for certain operational environments where these resources are limited.

When considering the cost implications of utilizing GPT-3.5, GPT-4, and LLAMA for a project, it's crucial to account for variables such as usage volume and access methodologies, which can significantly influence the overall expense. LLAMA, being open-source, theoretically offers a higher degree of cost efficiency. The flexibility to adjust computational resources based on the project's demands such as scaling out for increased processing needs or scaling in when less power suffices is also taken into account. The decision to select the most appropriate LLM for a project thus requires a comprehensive evaluation of not only the linguistic and technical capabilities of each model but also their operational and economic feasibility. Balancing these factors is essential for ensuring that the chosen LLM aligns with the project's objectives, providing both the desired level of query understanding and response generation, as well as fitting within the operational parameters set forth by our project team.

### ***Staying Ahead of the Curve***

The key takeaway here is to be both strategic and adaptive when it comes to the fast-paced landscape of language models. The recent model released by Meta which is LLaMa-3 and Gemma from Google will be leveraged for this project. The team will start integrating these established, open-source and ranked LLMs for our core question-answering component, and build a baseline evaluation to vet which one is better for the project. However, the team also has to actively monitor how LLMs are evolving and look for new/advanced retrieval augmentation

approaches that may get released. These might offer future opportunities to optimize specific components of our system for accuracy, speed, or specialized domain understanding.

The fact is that in smaller non-industry products the concept of ‘RAG’ is not very common and still not fully understood because it is still very much an industry oriented term. Instead of retraining LLMs with latest data, it is much easier to incorporate newer data to an existing vector store or knowledge base to avoid the compute intensive task of retraining LLMs.

### **1.5. Literature Survey of Existing Research**

In the paper "Automating Information Retrieval from Faculty Guidelines: Designing a PDF-Driven Chatbot powered by OpenAI ChatGPT" by Mutiara Auliya Khadija, Abdul Aziz, and Wahyu Nurharjadmo (2023) from Universitas Sebelas Maret, Indonesia, they introduce an innovative approach to information retrieval from academic guidelines presented in PDF format. The research addressed the challenge of making e-book content, such as educational materials and faculty guidelines, more accessible and interactive through an automated chatbot solution. The chatbot leveraged Large Language Models (LLMs), specifically OpenAI's ChatGPT (GPT-3.5 Turbo) and the LangChain framework, to generate coherent responses that align closely with the context of the PDF documents.

The proposed method encompasses document loading, content splitting, vector storage, retrieval, generation, and responses. The specific PDF documents containing faculty guidelines are then loaded and segmented into manageable parts to facilitate their processing and storage as vectors within the system. Pinecone is used for vector storage, serving as a database that indexes and maintains these vector embeddings.

The retrieval phase involved transforming user queries into embeddings and matching these with the most semantically similar text stored in the vector database. During the generation

phase, the system utilized GPT-3.5 Turbo to produce responses based on the prompts and data retrieved. The final step, response and evaluation, ensures that the chatbot can automatically fetch relevant information from the PDF documents to answer user queries. The responses were then evaluated using metrics such as the Negative Log-Likelihood Ratio Loss and BLEU Scores to assess the chatbot's performance in delivering accurate and contextually appropriate information. This research showcased the potential of combining advanced AI models and frameworks to create a chatbot capable of efficiently navigating and extracting information from complex documents, thereby enhancing the accessibility of educational materials and guidelines for academics. A research study conducted by researchers from Meta (2023) introduced the LLaMA series of foundation language models, showcasing a range of models from 7 billion to 65 billion parameters. This work underscores a pivotal shift in language model training methodologies by utilizing publicly available datasets, sidestepping the need for proprietary data. The LLaMA models, particularly highlighted by their 13B parameter variant, demonstrated an ability to rival or even surpass the performance of more established, larger models, marking a significant advancement in this field. A critical examination of the LLaMA approach reveals its emphasis on optimizing the balance between model size and dataset comprehensiveness. This research shows that a strategic increase in the breadth of data for training, rather than merely expanding model size, can significantly enhance both performance and efficiency. This principle is particularly evident in the superior benchmarks achieved by the 13B LLaMA model against GPT-3.5, despite the latter's larger parameter count.

The architecture of LLaMA adheres to foundational principles observed in contemporary large language models (LLMs), such as the GPT and BERT frameworks. Central to its design are transformer blocks, which employ layers of self-attention mechanisms for dynamic word

significance weighting within texts. Additionally, the LLaMA models undergo a two-phased development process involving initial pre-training on extensive text corpora to learn universal language patterns, followed by fine-tuning on narrower, task-specific datasets. This methodology is complemented by an embedding layer for vectorizing input tokens and various output layers tailored to specific applications, ranging from text generation to classification tasks.

Furthermore, the paper explores the integration of LLaMA models with LangChain, a framework designed to bridge language models with external data sources or systems for advanced application development. This integration facilitates a seamless connection between LLaMA's generative capabilities and LangChain's data-handling features, enabling the creation of sophisticated tools that can interpret and engage with complex data in novel ways.

The groundbreaking work by Devlin et al. (2019) on BERT (Bidirectional Encoder Representations from Transformers) has been a monumental shift in the field of Natural Language Processing (NLP), establishing new paradigms for language understanding. Their introduction of BERT, a method for pre-training language representations, leverages the innovative concept of bidirectional context understanding. Unlike previous models that processed text in a unidirectional manner, BERT analyzes the context from both the left and right sides of each token within a sentence. This dual-direction processing enables BERT to generate word embeddings that are rich in semantic meanings, capturing the nuances of language with remarkable precision.

BERT's architecture, built upon the Transformer model introduced by Vaswani et al. (2017), utilized a deep learning approach that eschews traditional sequence-based constraints for a more fluid and comprehensive analysis of language patterns. By pre-training on a vast corpus of text before fine-tuning on specific tasks, BERT achieves state-of-the-art performance across a

wide array of NLP benchmarks, such as sentence classification, question answering, and language inference.

The significance of this work lies not only in their technical achievements but also in the broader implications for machine learning and AI research. By demonstrating the efficacy of bidirectional pre-training, BERT has inspired a wave of innovations in language models, leading to the development of subsequent models that build upon its foundational principles. The model's ability to deeply understand context and nuance in language has opened new avenues for natural language understanding, machine translation, and even conversational AI, making it a cornerstone in the ongoing evolution of AI's linguistic capabilities.

Mistral 7B is a large language model boasting 7 billion parameters, renowned for its superior efficiency and performance across various benchmarks when compared to predecessors such as Llama 1 and Llama 2. It incorporates innovative attention mechanisms like grouped-query attention (GQA) and sliding window attention (SWA), which enable faster inference times and adept handling of longer sequences while minimizing computational expenses. The architectural enhancements of Mistral 7B make it particularly suitable for real-world applications, offering more efficient, scalable, and robust solutions in machine learning literature analysis. Its release under the Apache 2.0 license as an open-source model promotes broader adoption and continuous innovation in the sector.

Sliding Window Attention reduces the operational burden typically seen with vanilla attention, where the number of operations is quadratic to the sequence length, and memory usage grows linearly with the token count. This reduction in efficiency leads to increased latency and decreased throughput at inference due to limited cache availability. To counter this, sliding window attention limits each token's attention to a maximum of  $W$  tokens from the previous

layer. Although tokens outside this window still contribute to the next word prediction, each attention layer allows information to progress by up to  $W$  tokens. Consequently, after  $k$  attention layers, information can advance by as much as  $k \times W$  tokens.

Mistral 7B employs a transformer architecture that includes a Rolling Buffer Cache with a fixed attention span, enabling control over cache size using a rolling buffer mechanism. The cache, having a fixed size of  $W$ , stores the keys and values for timestep  $i$  in position  $i \bmod W$ . When  $i$  exceeds  $W$ , older values in the cache are overwritten, and the cache size remains constant. This technique reduces cache memory usage by 8x on sequences as long as 32k tokens without degrading model quality.

Pre-fill and Chunking strategies are used during sequence generation, where each token is predicted sequentially based on preceding tokens. However, because the prompt is predetermined, we can pre-load the key-value ( $k, v$ ) cache with the prompt. For exceptionally long prompts, we can segment them into smaller chunks and pre-load the cache with each segment, choosing the window size as our chunk size. This necessitates computing attention for both the cache and each chunk as mentioned in their work detailed by Jiang, A. Q. et al in 2023 in their study, "Mistral 7B".

The Gemma models, as introduced by Google DeepMind in their paper "Gemma: Open Models Based on Gemini Research and Technology," in 2024 represent a significant advancement in the field of open-source language models, building upon the technology and research from the Gemini model lineage. These models come in two configurations: a 2 billion parameter model optimized for CPU and on-device applications, and a 7 billion parameter model designed for GPU and TPU environments. Trained on an extensive dataset of up to 6 trillion tokens, the Gemma models incorporate state-of-the-art architectures and training methodologies,

achieving top-tier performance in language comprehension and reasoning tasks across a variety of contexts, including dialogue, instruction-following, and safety-related applications.

From an architectural standpoint, the Gemma models employ a transformer decoder architecture enhanced with novel features like Multi-Query Attention and Rotary Positional Embeddings (RoPE). The Multi-Query Attention, used in the smaller model, enables more efficient computation, which is crucial for scaling down to devices with limited resources. The larger model uses traditional multi-head attention to manage its more extensive computational needs effectively. RoPE contributes to the models' ability to handle the positional context more dynamically, essential for processing longer sequences. Additionally, the introduction of GeGLU activations offers a refined alternative to the conventional ReLU, helping the models capture more complex data patterns. The training regime for Gemma not only involves rigorous safety and bias checks but also innovative normalization techniques like RMSNorm to maintain stability throughout the training process. The use of a modified SentencePiece tokenizer ensures that the text retains more structural integrity, which is vital for accurate model training and performance. This comprehensive approach outlined in their paper ensures that the Gemma models not only push forward the capabilities of language models but also adhere to rigorous ethical standards in AI development and deployment.

In their detailed survey, Zhao et al. (2023) embark on an extensive exploration of large language models (LLMs), offering a thorough examination of the landscape that defines these powerful tools in the realm of artificial intelligence and natural language processing. This work meticulously reviews the evolution, architecture, applications, and ethical considerations associated with LLMs. The authors dissect the foundational technologies underpinning LLMs,

including their training methodologies, the datasets employed, and the innovative algorithms that drive their performance.

They delve into the various applications of LLMs, highlighting their transformative impact across sectors such as text generation, translation, content moderation, and more, while also addressing the challenges and limitations inherent in their deployment. Furthermore, the survey engages in critical discussions on the ethical implications of LLMs, from biases in model outputs to concerns regarding privacy and the potential for misuse. By synthesizing current research trends, challenges, and future directions, Zhao et al.'s work stands as a comprehensive resource to understand and contribute to the rapidly advancing field of LLMs.

In the pursuit of advancements within Open-Domain Question Answering (ODQA), Siriwardhana and colleagues (2023) have proposed significant improvements through the development of the RAG-end2end model, an extension of the original Retrieval Augmented Generation (RAG) framework. Addressing the limitations inherent in the RAG model, particularly its optimization for Wikipedia-based knowledge bases, this work pioneers in facilitating domain adaptation.

The cornerstone of their contribution lies in the novel approach of jointly training both the retriever and generator components of the RAG model, enabling it to dynamically update all components of the external knowledge base during the training process. Additionally, introducing an auxiliary training signal aims to integrate domain-specific knowledge by mandating RAG-end2end to reconstruct sentences utilizing pertinent information from the external knowledge base. This methodology showcases substantial performance enhancements across domain-specific ODQA tasks in varied datasets, including those pertinent to COVID-19, News, and Conversations.

A pivotal aspect of their methodology involves the employment of asynchronous processes for the dynamic re-encoding and re-indexing of the knowledge base. This strategy optimizes a joint question answering and paraphrasing signal loss, permitting the model to efficiently update the embeddings and the knowledge base index without disrupting the ongoing training process.

The insights garnered from this paper are valuable as they allow the model to efficiently update the embeddings and the knowledge base index without interrupting the training process. Implementing these strategies could improve the system's overall ability to generate relevant and accurate responses when dealing with unfamiliar content. The approach of joint training and dynamic updating of the knowledge base components could be particularly beneficial in enhancing the system's flexibility and performance.

The study "Embodied Epistemology: A Meta-Cognitive Exploration of Chatbot-Enabled Document Analysis" presents novel research on constructing a PDF chatbot using ChatGPT 3.5 Turbo and the LLM Model, focusing on scalable AI applications for text generation, language translation, and original content creation. It explores the use of the LangChain framework and a Streamlit-based homepage to enhance user interactions, highlighting the potential of chatbots across various domains like customer service, education, and research. The paper, presented at the 2023 International Conference on Evolutionary Algorithms and Soft Computing Techniques (EASCT), demonstrates the integration of advanced AI models to develop chatbots capable of informative and engaging interactions.

The architecture employs an innovative integration of GPT 3.5 Turbo to enhance document analysis and response generation from PDFs. At its core, the system utilizes OpenAI's GPT 3.5 Turbo, built upon GPT-3's transformer architecture, to produce coherent, conversational

responses, while the LLM Model, leveraging BERT's bidirectional understanding of document context, focuses on in-depth document analysis. This integration facilitates a robust document embedding process where PDFs are converted into vector representations, enabling precise response generation tailored to user queries.

A pivotal element of the architecture is the question-answering module that bridges user inquiries with accurate answers derived from the processed document vectors, employing cosine similarity to ascertain relevancy. Additionally, the system incorporates LangChain for enhanced data processing and Streamlit for user-friendly web interface development, ensuring efficient and natural user interactions. Performance optimization is a key focus, aiming to refine response quality and speed through model fine-tuning and data processing enhancements.

The architecture's seamless functionality is further supported by API integration, which streamlines communication between the models and the web interface, offering users a responsive and engaging conversational experience.

The survey paper "Retrieval-Augmented Generation for Large Language Models: A Survey" in 2024 by Yunfan Gao and colleagues provides an extensive overview of the advances in Retrieval-Augmented Generation (RAG) systems, which enhance the capabilities of large language models (LLMs) by integrating external data retrieval into the generative process. RAG systems address several limitations of LLMs, such as generating outdated or factually incorrect information by dynamically retrieving data from external databases during the inference phase. This approach not only bolsters the accuracy and relevance of responses but also enables continuous knowledge updates and domain-specific data integration, making RAG particularly valuable for knowledge-intensive tasks. The paper categorizes RAG systems into three

paradigms—Naive RAG, Advanced RAG, and Modular RAG—each offering progressive enhancements and sophistication in handling information retrieval and integration.

Furthermore, the survey highlights ongoing challenges and future directions in the RAG field, such as optimizing the balance between retrieval quality and contextual relevance, improving the robustness of RAG systems against noisy data, and enhancing their scalability and efficiency. The authors also discuss the potential for RAG systems to extend beyond textual data, incorporating multimodal data (such as images and videos) to widen their applicability across various domains and tasks. As RAG technology continues to evolve, it presents a promising avenue for making LLMs more practical and effective in real-world applications, offering a more nuanced and contextually aware approach to automated question-answering and content generation.

## **2. Data and Project Management Plan**

### **2.1. Data Management Plan**

#### ***Data Collection Approaches***

Our data collection approach involves sourcing research articles from two key datasets available on Kaggle, specifically focused on arXiv and the Journal of Machine Learning Research (JMLR). These datasets provide a wealth of STEM-related research articles, with our emphasis on the domains of Machine Learning (ML) and Artificial Intelligence (AI).

The arXiv dataset, containing approximately 1.7 million research publications across STEM disciplines, provides essential metadata such as titles, authors, abstracts, and categories. To focus exclusively on data science-related papers, we filter the dataset using the category column. A dedicated data extraction pipeline is then employed to construct URLs using the arXiv ID, enabling retrieval of the full content of each paper. This pipeline forms the foundation for our application's knowledge base, allowing in-depth content analysis.

Similarly, the JMLR dataset features over 2800 ML-specific publications, offering detailed metadata for each paper. Leveraging our data extraction pipeline and the JMLR-provided URLs, we extract full-text PDFs of these papers.

This comprehensive approach ensures our application has access to a diverse and extensive repository of research findings and methodologies within the field of data science.

Our data collection process involved designing automated scripts to efficiently parse and retrieve a large volume of research papers. We ensured compliance with the terms of service for the respective journal websites, maintaining ethical access and usage of the data. This methodology emphasizes the creation of an extensive dataset that accurately represents the current landscape of ML and AI research.

### ***Data Storage and Management Methods***

The data storage strategy for this project is designed to prioritize scalability and accessibility by leveraging Google Cloud Storage (GCS) and Google Drive. GCS serves as the primary storage solution due to its secure and scalable infrastructure, which is essential for handling large volumes of research papers. Multi-region support in GCS ensures easy access and flexibility. Google Drive is used as a secondary backup repository to ensure data redundancy.

To manage the integration of new research papers, an automated pipeline, equipped with a scheduler, is implemented to regularly check for and load newly published data in real-time.

A data migration script is employed to seamlessly transfer files from Google Drive to GCS, ensuring that the knowledge base remains current with the latest advancements in machine learning and artificial intelligence.

The data management protocols include anonymization of personally identifiable information within the dataset, safeguarding the privacy of the authors and researchers whose work forms the foundation of the project. This strategy supports the project's goal of providing a robust, scalable, and ethical approach to data management in the realm of AI and ML research.

Table 3 outlines the folder structure and naming conventions adopted in our project. The Raw Dataset is subdivided into 3 folders: JMLR, ARXIV\_AI and ARXIV\_ML. Each folder is categorized and organized to maintain the relevant articles and papers which facilitates straightforward access to material. To refine the model, the data is split into train, test and validation allowing systematic training, finetuning and evaluation of the AI models on distinct sets of data, to ensure the accuracy of the system and to validate the performance of the AI model on unseen data. The organization facilitates efficient management of the data.

**Table 3***Folder Structure and Naming Conventions**(Original from Team 8)*

<b>Purpose</b>	<b>Folder Name</b>	<b>File Name Convention and Versioning</b>
Main Folder	Research_dataset	
Raw metadata	arxiv	arxiv-metadata-oai-snapshot.json
	JMLR	jmlr_data.csv
Extracted Dataset	JMLR	<paper_id>.pdf
	ARXIV_AI	<arxiv_id>.pdf
	ARXIV_ML	<arxiv_id>.pdf
Test, Train and Validation data	Split Data/train	<paper_id>.pdf <arxiv_id>.pdf
	Split Data/validation	<paper_id>.pdf <arxiv_id>.pdf
	Split Data/test	<paper_id>.pdf <arxiv_id>.pdf

***Data Usage Mechanism***

The data usage mechanism is designed to efficiently manage the extensive dataset of research papers sourced from arXiv and JMLR, stored securely in Google Cloud Storage (GCS). This dataset serves as the foundational knowledge base for the application. GCS provides robust security and accessibility features, protecting sensitive data from unauthorized access while enabling seamless collaboration among project team members. To ensure secure data handling, permissions and privileges for accessing the GCS bucket are assigned based on specific project requirements, minimizing the risk of data breaches and maintaining efficient access for authorized users.

The data life cycle moves to the maintenance phase after the project completion and the data remains only on Google Drive for potential updates or future iterations of the project. This strategy is followed not only to reduce the cost of GCS but also to maintain and preserve this rich source of data for further academic practices. Every member of our team is equally responsible for implementing this data management plan and ensuring it is reviewed and revised for any new activities or updates.

## **2.2. Project Development Methodology**

### ***System Development Life Cycle***

The project utilizes the CRISP-DM life cycle model, incorporating structured deadlines for each phase and breaking tasks into manageable components. This framework ensures flexibility by allowing iterative revisiting and refinement of tasks as needed. Responsibilities and resource allocations for the Data Management Plan (DMP) are clearly outlined, promoting efficient project execution and progress toward successful implementation. Specific details of the DMP responsibilities and resource allocations are presented in Table 4.

**Table 4**

*Responsibility and Resource Allocation for DMP* *(Original from Team 8)*

<b>DMP Phase</b>	<b>Responsibility</b>	<b>Resource</b>
Data collection, documentation, and metadata	Explore different datasets available online. Review and finalize the dataset.	Divya/Sourabh
Ethics and legal compliance	Ensure that all the software licenses are acquired, and the team complies with the terms and conditions of the agreements.	Poojitha/Shruti
Storage and Backup	Version the files appropriately, and create backup and storage copies	Shruti/Divya

	of data.	
Data Sharing	Data is available publicly on the arXIV. Feature engineering data can be shared and placed on the shared drive.	Divya/Poojitha/Shruti/Sourabh
Data Preservation	Decide what data to retain and preserve and manage the access, cost of archival and storage.	Divya/Sourabh

### ***Project Development Processes and Activities***

**Business Understanding.** The primary goal of this project is to enhance the accessibility and comprehension of Machine Learning (ML) and AI research literature, particularly for users unfamiliar with technical content. This initiative seeks to enable robust technical discussions, encourage the exploration of new research areas, and offer deeper insights into ML literature for a diverse audience. By developing a system capable of interacting with uploaded PDFs and acting as a Scientific Literature Review Assistant, the project aims to streamline the literature review process and facilitate research discovery. This approach is designed to significantly reduce the time and effort needed to navigate the rapidly evolving ML landscape. Additionally, the project aspires to extend its conversational capabilities to support other academic and professional domains that require an understanding of complex technical information.

To establish actionable plans for achieving the project's objectives, a comprehensive literature survey will be conducted. This survey will explore key areas, including the accessibility of ML research, conversational AI systems, scientific literature analysis techniques, and adaptive learning technologies. The insights gained will guide the finalization of the project scope and the development of a detailed project plan. The plan will include the creation of a Work Breakdown Structure (WBS) document, Gantt chart, and Program Evaluation and Review

Technique (PERT) chart. Project management tools such as Asana, Lucidchart, and TeamGantt will be employed to outline deliverables for each phase and identify the required resources and tools. This structured approach will provide a clear roadmap to achieve the project's goals effectively.

**Data Understanding.** The data understanding phase is critical to acquiring the foundational datasets necessary for the project's success. This phase involved selecting the arXiv and Journal of Machine Learning Research (JMLR) collections, accessible via Kaggle, as primary data sources. These repositories of STEM-related research articles were filtered to focus specifically on Machine Learning (ML) and Artificial Intelligence (AI) domains. Metadata, including titles, authors, abstracts, and categories, was extracted for both datasets. Additionally, a data extraction pipeline was developed for the arXiv dataset to retrieve full research paper content through direct links. This process enabled a more in-depth content analysis, enriching the knowledge base for the application.

**Data Preparation.** The data preparation phase focuses on refining the initial datasets from the arXiv and Journal of Machine Learning Research (JMLR) collections to target Machine Learning (ML) and Artificial Intelligence (AI) research exclusively. This involves systematically filtering out papers unrelated to these domains. Additionally, a structured methodology is implemented to construct arXiv URLs, enabling access to the full content of each paper. This process culminates in the extraction of approximately more than 25,000 relevant papers from both sources. This thorough preparation establishes a solid foundation for subsequent analysis and modeling, driving the development of the system.

**Modeling.** The modeling phase involves the development and optimization of five advanced models: an Ensemble Hybrid RAG, Adaptive RAG, Super RAG, a Local RAG agent

utilizing combined strategies, and a Multimodal model. A vector database is constructed from the JMLR and arXiv datasets, enabling efficient embedding management and supporting the models in analyzing complex ML and AI literature. Additionally, a search engine is integrated, allowing users not only to find relevant research papers but also to search for similar papers based on their interests. Leveraging LLAMA, a fine-tuned Mistral Nemo model, LangChain, and OLLAMA, the system provides a conversational interface. Through platforms like Streamlit and Gradio, users can upload PDFs, engage in dynamic, context-aware discussions, explore intricate ML and AI concepts, and discover similar research, enhancing the overall research experience.

**Evaluation.** The evaluation phase entails rigorous testing to assess the system's response accuracy, query handling efficiency, and its ability to integrate tables within conversations. Metrics such as context relevancy, context recall, faithfulness, and answer relevancy are employed to evaluate response quality, coherence, and completeness. User feedback is integral to this process, offering critical insights for iterative improvements. This thorough evaluation ensures the system meets high standards of performance and usability, ensuring it meets the standards set.

**Deployment.** The deployment phase focuses on selecting optimal platforms such as Google Cloud Platform (GCP), Google Colab, GitHub, and Hugging Face to host the application. These platforms ensure accessibility for researchers, students, and professionals, enabling seamless interaction with the system. This strategic approach aims to enhance the understanding and accessibility of ML and AI literature, providing an impactful resource for its target audience.

### 2.3 Project Organization Plan

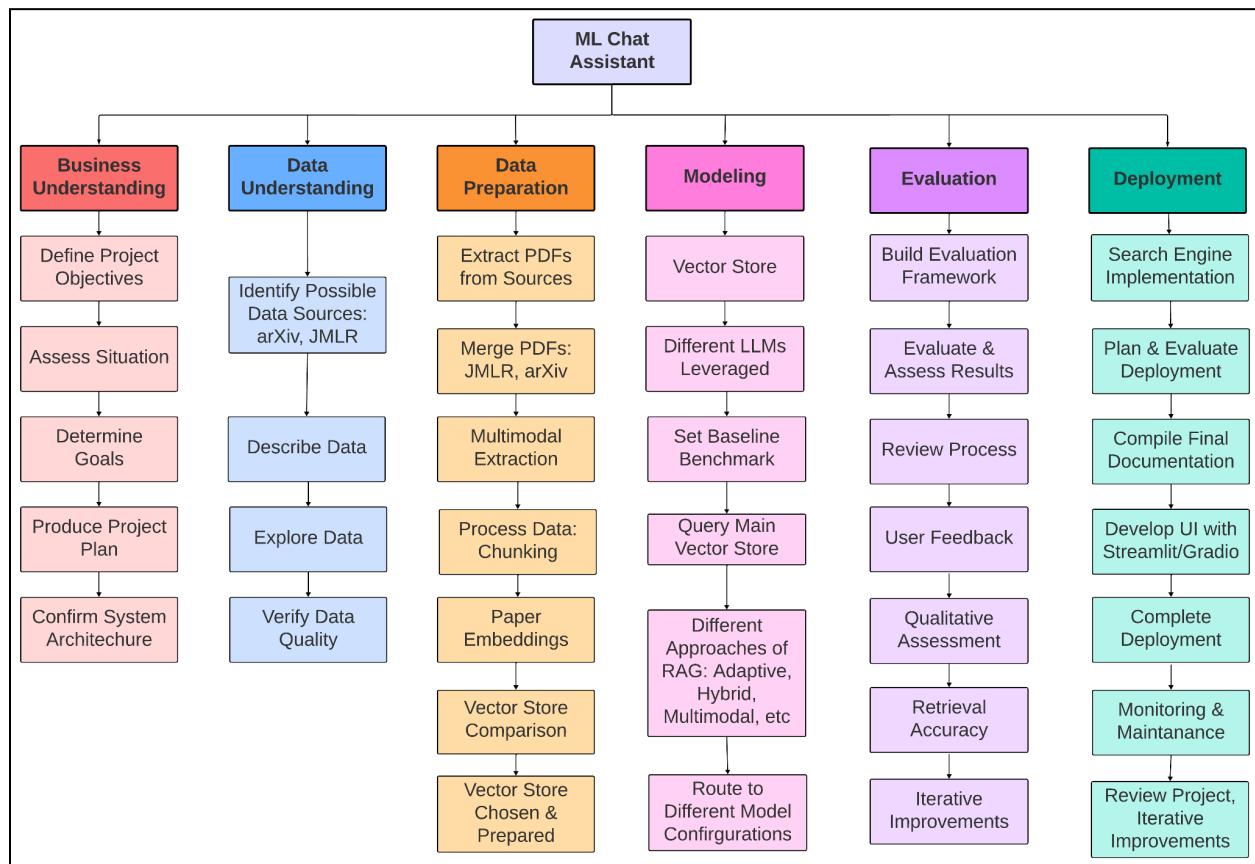
The Work Breakdown Structure (WBS) showcased in Figure 3 outlines the steps involved

in the project for building the ML-Chatbot Assistant for scientific research/academia in the fields of AI and ML. The WBS is divided into six main phases: Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation, and Deployment based on the Cross Industry Process for Data Mining (CRISP-DM) Methodology and is a comprehensive roadmap for developing this project.

**Figure 3**

*Work breakdown structure of the Project*

(Original from Team 8)



The initial stage involves defining the project objectives and assessing the current situation to establish clear goals and a comprehensive project plan. This phase includes identifying potential data sources, describing and exploring the data, and verifying its quality to confirm the system architecture. This foundational work is critical to ensure that the project's

direction aligns with the business requirements and that the available data supports these needs effectively.

In the data preparation phase, the plan is to extract and merge PDFs from sources such as JMLR and arXiv, perform text and multimodal extraction, and process this data through techniques like chunking and creating paper embeddings. This processed data is then utilized to build and prepare a vector store. For modeling, the WBS highlights the leverage of different large language models (LLMs) and the setting of baseline benchmarks. This phase also includes querying the vector store and employing various advanced retrieval techniques like Adaptive RAG, Super RAG, Hybrid RAG, Corrective RAG and Self RAG too.

The deployment phase involves implementing the scientific review assistant, developing a user interface with Streamlit, and compiling the final documentation. Post-deployment, there is a plan for ongoing monitoring and maintenance to ensure the system's continuous effectiveness and efficiency.

The structured approach provided by the CRISP-DM framework ensures a comprehensive development and evaluation cycle, meticulously addressing both technical and business aspects of the project. The WBS suggests thorough preparation and review strategies, aiming for a robust deployment of the ML chat assistant. This level of detailed planning indicates a strong commitment to creating a high-quality tool that meets user needs and adapts efficiently to feedback.

## **2.4 Project Resource Requirements and Plan**

### ***Software Requirements***

To enhance project development and functionality, a diverse suite of software tools was utilized throughout the project lifecycle. Asana facilitated project management by tracking

deliverables, milestones, and activities, while Google Colab and GitHub streamlined coding collaboration and version control. Real-time communication and team coordination were supported through Slack. Deployment processes were optimized using Docker for containerization and orchestration, alongside Streamlit and Gradio for UI development. Google Cloud Platform (GCP) provided robust hosting capabilities for databases and applications.

Data from sources such as JMLR and arXiv were ingested using Python scripts and ETL tools, with the primary vector database built using FAISS and ChromaDB to efficiently manage structured and unstructured data. Tools like Pandas enabled seamless data manipulation, while PyPDF2 and Tesseract facilitated text extraction from PDFs and tables. For model development, TensorFlow played a critical role in training the hybrid Retriever-Augmented Generation (RAG) model, while LangChain enhanced conversational AI capabilities by integrating external data sources.

Advanced AI frameworks such as Hugging Face Transformers and OLLAMA were employed for leveraging pre-trained models and fine-tuning them to suit specific project needs. FAISS further supported similarity searches and dense vector clustering, ensuring efficient data retrieval and organization. This integrated use of modern tools and technologies facilitated a robust system capable of handling complex data processing and conversational requirements. Additional tools included Google Colab for collaborative integration, Zoom for video conferencing, Lucidchart for visualizing workflows, and Python 3.9 for coding and development. This comprehensive toolkit not only supported the CRISP-DM framework but also ensured a meticulous and efficient approach to achieving project goals. Further tool details are provided in Table 5.

**Table 5***Software Requirements**(Original from Team 8)*

<b>Software</b>	<b>Purpose</b>	<b>Cost</b>
Asana	For tracking project deliverables, milestones, and tasks	Free (Basic Plan for Students)
Cloud Platforms (GCP)	For hosting the application, and databases and for utilizing additional computational or AI services.	Free Tier (with GCP Education Account)
Docker and Kubernetes	Containerization and orchestration	Free (Open Source)
FAISS, ChromaDB	For efficient similarity search and clustering of dense vectors, facilitating the vector database required for the project.	Free (Open Source)
GitHub or GitLab	For version control and collaborative code development	Free (with Student Accounts)
Google Colab	Allows multiple users to collaborate and integrate in real-time.	Free, with an option for Colab Pro(\$9.99 per Month) for access to more powerful GPUs
Hugging Face Transformers	For leveraging pre-trained models like LLAMA and fine-tuning them.	Free (with public models)
LangChain, Olama	For integrating the LLM with external information sources, enhancing the conversational capabilities of the chat system.	Free (Open Source)
Lucid Charts	Create flowcharts for data flow and project management.	Free (Basic Plan for Students)

Pandas	Data manipulation	Free (Open Source)
PyPDF2	PDF processing	Free (Open Source)
Python Version 3.9	Model Development	Free (Open Source)
Scripts for Data Ingestion	Python scripts or ETL (Extract, Transform, Load) tools to ingest data from sources like JMLR and arXiv.	Free (Open Source tools)
Slack	For team communication and real-time collaboration.	Free (Basic Plan for Students)
Streamlit, Gradio	For building the user interface (UI) of the application.	Free
TensorFlow	For developing and training the hybrid Retriever-Augmented Generation (RAG) model	Free (Open Source)
Zoom	Video conferencing and collaboration	Free (Basic Plan with limitations)
<b>Total Cost</b>	<b>FREE</b>	

### ***Hardware Requirements***

The project leverages a combination of lab infrastructure, personal MacBooks, and cloud-based resources to ensure smooth execution. Robust servers and cloud infrastructure provide sufficient CPU, RAM, and GPU resources for computational tasks. Google Drive, offering up to 15 GB of storage, and Google Colab are utilized for efficient data management and resource-intensive computations. OneDrive is integrated with JupyterLab for creating Python Notebooks, aiding in machine learning development and data pre-processing. This blend of lab infrastructure and personal devices facilitates seamless collaboration, efficient data handling, and rapid project progression. Detailed hardware specifications and objectives are outlined in Table 6.

**Table 6***Hardware Requirements**(Original from Team 8)*

<b>Resource</b>	<b>Purpose</b>	<b>Configuration</b>	<b>Location</b>	<b>Storage</b>
Google Cloud Storage (GCS)	Data and Source Files Storage	Standard Storage Class	Iowa (us-central1) / South Carolina (us east1)	3 GB (Currently) 20 GB (Scale-out)
Local Machine	Hosts client applications including Jupyter Notebook and Python, for local development and testing.	Important files MacOS Installed and stored directly on the local machine.	Machine specs: 8 GB RAM, M1 Processor, 1 GB graphics card	Total internal storage is 1 TB
OneDrive	Provides synchronization, offline access, and a backup solution for important files.	Integrated with the local machine for seamless access and backup.	Linked to a user account on Microsoft Azure Cloud.	Utilizes part of the local machine's 1 TB storage, size depends on user allocation.

## 2.5 Project Resource Requirements and Plan

### **Gantt Chart**

The Gantt chart provides a clear, structured timeline for the ML Chat Assistant project, detailing tasks, sub-tasks, durations, milestones, and required resources. Aligned with the CRISP-DM methodology, it also integrates the User Interface development, segmented into manageable sprints with focused outcomes.

### **Key Considerations**

- 1) **Time Estimates:** Work efforts exclude weekends to provide realistic timelines.
- 2) **Planned Breaks:** The schedule incorporates semester holidays and designated breaks.

- 3) **Resource Allocation:** Tasks are assigned based on individual expertise and availability, fostering collaboration for complex challenges.
- 4) **Task Duration:** Subtasks are designed to be completed within two days to enhance efficiency.
- 5) **Methodology:** The CRISP-DM lifecycle provides a structured framework, ensuring interdependencies between phases—such as Data Exploration informing Data Preparation—are accounted for.

The success of each phase is contingent on the satisfactory completion of its predecessor.

Interdependencies exist between certain tasks across various phases; for example, the insights gleaned during Data Exploration directly inform choices made within the Data Preparation stage. A detailed Gantt chart illustrating the project's timeline and comprehensive task breakdown can be found in Appendix A.

**Business Understanding.** The Business Understanding phase of the project begins in the first week of August, as outlined in Figure 4. This phase is pivotal, focusing on five key subtasks to establish alignment between project goals and technical execution. Initial tasks, such as defining objectives and conducting resource assessments, lay the groundwork for determining project goals and creating the Asana project plan. The insights from these steps guide the confirmation of system architecture and benefit the ongoing literature review, ensuring the architecture aligns with project objectives, available resources, and data goals. Table 7 provides a detailed overview of this phase.

**Table 7**

*Simplified Task List for Business Understanding Phase* *(Original from Team 8)*

Name	Task	Start Date	End Date	Dependent On
------	------	------------	----------	--------------

A	Determine Project Objectives	08/05/2024	08/07/2024	-
B	Assess Situation	08/07/2024	08/09/2024	-
C	Determine Goals	08/12/2024	08/14/2024	-
D	Produce Project Plan	08/15/2024	08/18/2024	-
E	Confirm System Architecture	08/19/2024	08/22/2024	A,B,C,D

**Data Understanding.** The Data Understanding phase, initiated on August 20th as outlined in Figure 4, lays the foundation for effective data utilization. This phase focuses on five critical tasks: selecting appropriate data sources, extracting a sample PDF from chosen sources, designing a robust data quality strategy, formulating a comprehensive data management plan, and crafting a detailed data collection blueprint.

By emphasizing strategic planning, this phase ensures that all data efforts align with the project's objectives. Establishing clear data quality and management protocols early on simplifies data handling and upholds integrity. Additionally, the data collection plan provides a structured framework for the upcoming data preparation phase, ensuring consistency and efficiency throughout. Table 8 offers a detailed overview of the Data Understanding phase.

**Table 8**

*Simplified Task List for Data Understanding Phase* *(Original from Team 8)*

Name	Task	Start Date	End Date	Dependent On
A	Select Data Sources	08/23/2024	08/27/2024	-
B	Sample Data	08/28/2024	09/02/2024	-
C	Data Quality Plan	09/04/2024	09/07/2024	-
D	Data Management Plan	09/09/2024	09/11/2024	-
E	Data Collection Plan	09/11/2024	09/12/2024	A,B,C,D

**Data Preparation.** The Data Preparation phase of the project focuses on transforming the collected data into a format suitable for model training and analysis. Key tasks include extracting PDFs from specified sources, systematically merging and organizing PDF content from JMLR and arXiv, and employing table extraction techniques to make tables within the PDFs machine-readable. Furthermore, data chunking is performed, and a vector store database is established to enhance data management and retrieval efficiency, ensuring readiness for subsequent project phases as shown in Table 9.

**Table 9**

*Simplified Task List for Data Preparation Phase* *(Original from Team 8)*

Name	Task	Start Date	End Date	Dependent On
A	Extract PDFs (papers)	09/13/2024	09/17/2024	-
B	Merge/Collate PDFs	09/18/2024	09/19/2024	A
C	Extract data from Table	09/20/2024	09/23/2024	B
D	Process Data: Chunking	09/24/2024	09/26/2024	C
E	Vector Store Database	09/27/2024	09/30/2024	B,C,D

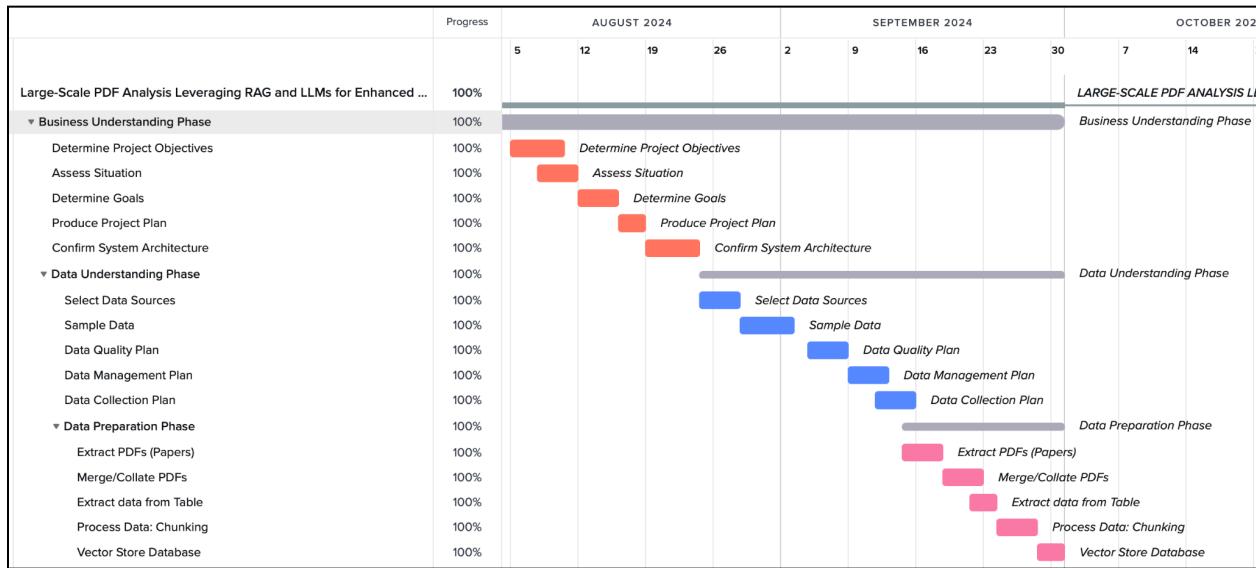
**Modeling.** The Modeling phase focuses on the development and implementation of the machine learning model, which serves as the foundation of the chat assistant's functionality. Core tasks include generating text embeddings to numerically represent text data and finalizing the LLaMA and Mistral models through selection, fine-tuning, or custom development. This phase incorporates the development of multiple advanced models, including Adaptive RAG, Super RAG, Local RAG Agent, Multimodal RAG, and a Hybrid Retrieval-Augmented Generation (RAG), to optimize information retrieval and ensure precise, context-aware responses. A baseline benchmark will track the model's performance throughout its development. The phase concludes with seamless integration of the model with the user interface (UI) to ensure an

efficient and user-friendly experience. Further details are outlined in Table 10, with the Gantt Chart in Figure 5 illustrating the project timeline.

**Figure 4**

*Gantt Chart of Business Understanding, Data Understanding and, Data Preparation Phase*

*(Original from Team 8)*



**Table 10**

*Simplified Task List for Modeling Phase*

*(Original from Team 8)*

Name	Task	Start Date	End Date	Dependent On
A	Create Text Embeddings	10/01/2024	10/02/2024	-
B	Finalize, Setup LLM's	10/03/2024	10/05/2024	A
C	Baseline Benchmark	10/06/2024	10/08/2024	A
D	Query Vector Store: RAG	10/09/2024	10/10/2024	B
E	Finalize RAG Approaches	10/11/2024	10/15/2024	B, D

**Evaluation.** The Evaluation phase of the project, as outlined in Table 11 and Figure 5, emphasizes a comprehensive assessment of the model's performance. This phase involves a

detailed review of the ML chat assistant's outputs and the collection of valuable user feedback to enhance user experience and pinpoint areas for improvement. Additionally, a qualitative assessment evaluates the model's ability to generate contextually relevant and accurate responses. Measuring retrieval accuracy is critical to determining the model's effectiveness in accessing the required information. Beyond identifying weaknesses, this phase focuses on fostering continuous improvement, using findings and insights to drive iterative refinements and enhance the project's overall success.

**Table 11**

*Simplified Task List for Evaluation Phase* *(Original from Team 8)*

Name	Task	Start Date	End Date	Dependent On
A	Evaluate and Assess Results	10/16/2024	10/17/2024	<i>Modeling</i>
B	Review Process	10/18/2024	10/22/2024	A
C	Feedback Analysis	10/22/2024	10/23/2024	A
D	Qualitative Assessment	10/24/2024	10/25/2024	A
E	Retrieval Accuracy	10/25/2024	10/28/2024	A
F	Iterative Improvements	10/29/2024	10/30/2024	C, D, E

**Deployment.** The Deployment phase shown in Table 12, transitions the project into a live, operational system. This phase begins with strategizing the deployment process, addressing infrastructure needs, scalability, and security considerations. Concurrently, comprehensive documentation, maintained throughout the project, will be finalized to serve as a reference for future updates and maintenance. A user-friendly interface will be developed using the Streamlit framework to enhance accessibility and interactivity. Following the completion of the UI, the project will be fully deployed.

The final steps include establishing monitoring and maintenance protocols to ensure the chat assistant's stability and performance. A comprehensive project review will follow, documenting key successes, challenges, and lessons learned, providing insights for future initiatives. Refer to Table 12 and Figure 5 and 6 for a detailed task breakdown.

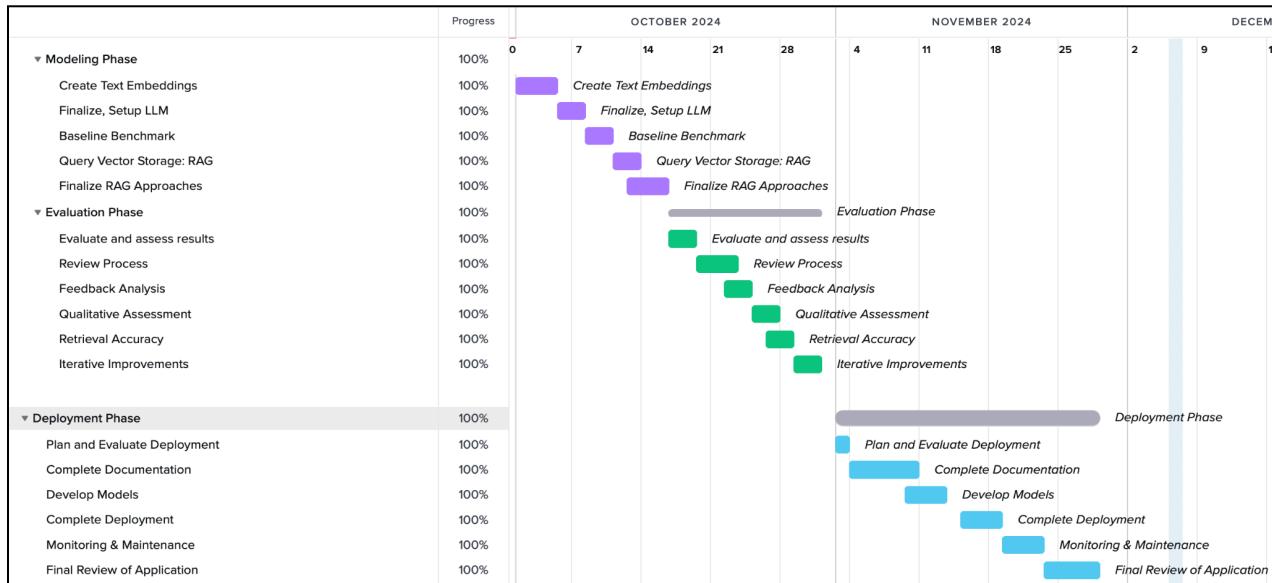
**Table 12**

*Simplified Task List for Deployment Phase* (Original from Team 8)

Name	Task	Start Date	End Date	Dependent On
A	Plan & Evaluate Deployment	11/01/2024	11/02/2024	-
B	Complete Documentation	11/04/2024	11/07/2024	-
C	Develop Models	11/08/2024	11/12/2024	-
D	Complete Deployment	11/14/2024	11/18/2024	A, C
E	Monitoring & Maintenance	11/19/2024	11/21/2024	D
F	Final Review of Application	11/22/2024	11/27/2024	D, E

**Figure 5**

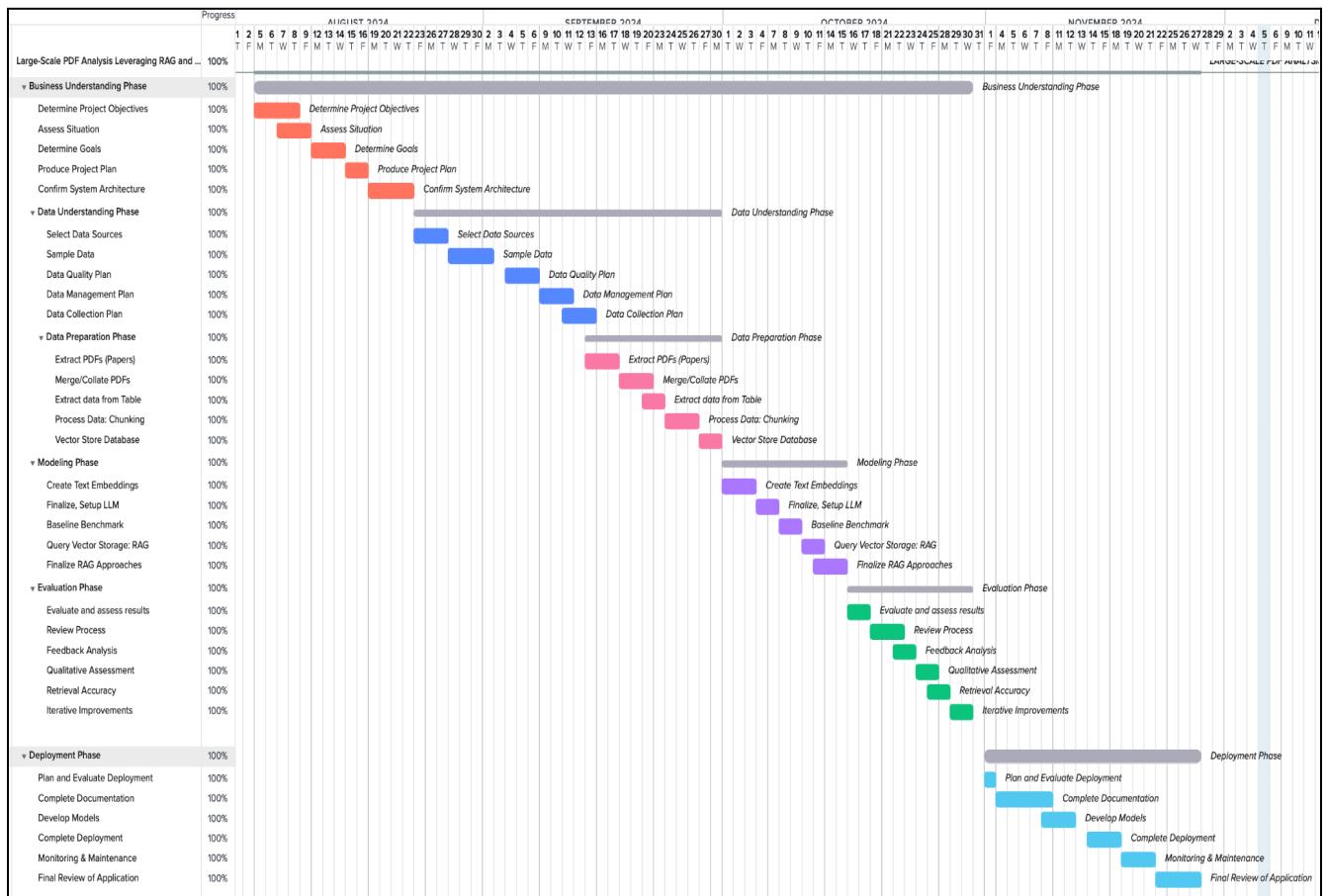
*Gantt Chart of Modeling Phase, Evaluation Phase* (Original from Team 8)



**Figure 6**

*Complete Gantt Chart*

*(Original From Team 8)*



**PERT Chart**

The PERT chart in Figure 7 illustrates the structured workflow of the project, highlighting task dependencies, critical paths, estimated durations, and potential risks. The initial stage, “Project Initiation,” establishes the project’s foundation by defining the scope and identifying available resources. Concurrently, “Technology Setup” and “Literature Review” tasks are carried out independently, providing the groundwork for subsequent phases. The “Data Acquisition” and “Data Preprocessing” stages focus on collecting and preparing the arXiv and JMLR datasets, ensuring the data is clean and ready for analysis. These phases lead

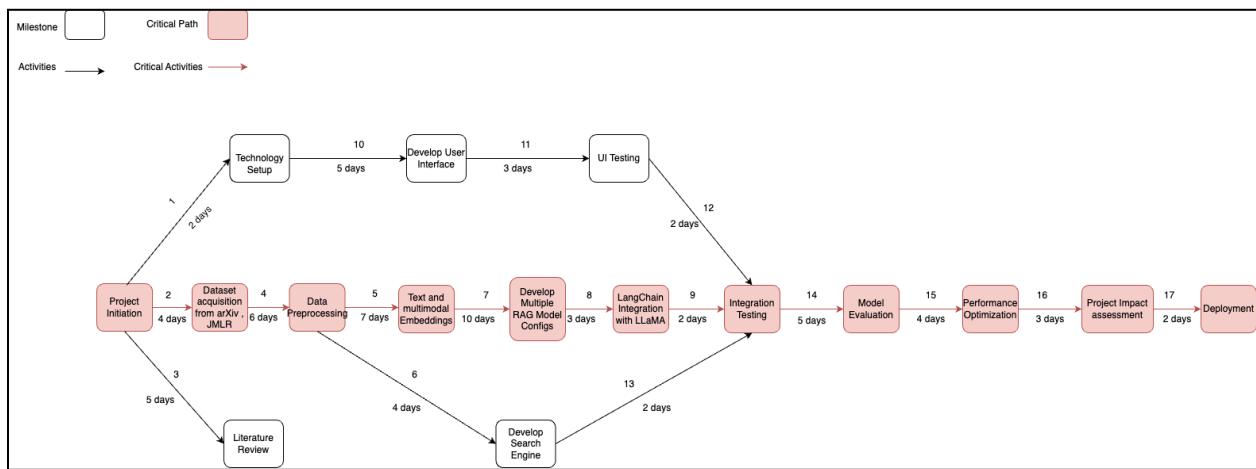
into the “Text and Multimodal Embeddings” stage, a critical step for text analysis. The embeddings are stored in a vector store, enabling the development of various RAG models such as RAG Agent, Multimodal RAG, Adaptive RAG, Hybrid RAG and Super RAG, designed to handle user inquiries effectively, which are then integrated with different LLMs and LangChain in the “LangChain Integration” phase. Simultaneously, the UI is developed using Gradio to ensure a seamless user experience.

The final stages include “Integration Testing,” where all components are combined and assessed for functionality, followed by “Model Evaluation” and recursive performance optimization to refine and enhance the system. This comprehensive workflow ensures a robust and efficient project development process.

**Figure 7**

*PERT Chart*

*(Original from Team 8)*



The critical path of the project begins with “Dataset Acquisition,” progresses through “Hybrid RAG Development and LangChain Integration,” and concludes with “Deployment.” This path represents the longest sequence of dependent tasks, determining the minimum time

required to complete the project. Table 13 provides a detailed description of each activity, including the associated dependencies needed to complete each task.

The PERT chart plays a crucial role in strategic planning and task management, offering a framework to identify task dependencies, allocate resources efficiently, and address potential risks. By visualizing these relationships, the PERT chart ensures the project remains on track, enabling the team to meet deadlines and achieve objectives within the defined timeline.

**Table 13**

*List of activities and dependencies* *(Original from Team 8)*

Activity	Description	Predecessor
1	Install necessary libraries and tools (FAISS, Streamlit, etc.)	-
2	Download PDF research papers from arXiv and JMLR	-
3	Study existing literature on Llama, RAG, and LangChain	-
4	Clean and preprocess downloaded datasets for model training	2
5	Text and multimodal Embeddings	4
6	Develop Search Engine	4
7	Develop Multiple RAG Model Configs	5
8	Integrate LangChain with LLaMA for external knowledge access	7
9,12,13	Test integration of all components (vector store, LLMs, RAG, LangChain)	8,11,13
10	Create the user interface using Gradio.	1
11	Test the User interface for usability and functionality.	10

14	Evaluate and compare different model performances	9,12,13
15	Optimize the system for performance and efficiency.	14
16	Evaluate the project's impact against initial goals and objectives.	15
17	Deploy the application for user access	16

### **3. Data Engineering**

#### **3.1 Data Process**

The first stage of the project consists of systematically gathering raw datasets from huge archives of scientific research publications. We primarily leverage JMLR and arXiv datasets as our main sources and screen the AI and ML papers to meet our specific domain requirements. Every item in these databases usually includes metadata, such as a distinct identifier (ID), that helps in accessing the whole PDF. This automated procedure guarantees an effective and methodical way of downloading a substantial collection of documents, which are subsequently organized in a structured style for further analysis.

After the completion of raw data gathering, the subsequent stage involved data preparation, a critical process for purifying and organizing the data in order to facilitate efficient analysis. Null records are handled effectively. Some papers that were withdrawn were excluded from the extraction process, and PDF parsing libraries were utilized to transform PDF publications into a format that can be read as text.

The dataset is partitioned into three subsets: 80% for training, 10% for validation, and 10% for testing. The extracted PDF text is divided into chunks and then converted into embeddings using various embedding models. The embeddings are saved in the vector stores, such as FAISS and ChromaDB.

The process for handling textual and visual data integrates advanced techniques to facilitate comprehensive analysis and retrieval. Textual data is processed using Hugging Face embeddings, while visual data is converted into textual descriptions and vector representations using PyTesseract alongside visual captioning methods. These unified embeddings, encompassing both text and image data, are stored in a Chroma Vector Store to enable efficient

similarity-based searches. For response generation, the system employs sophisticated models to deliver concise and contextually relevant answers by synthesizing insights from both modalities. This integrated approach provides a robust framework for analyzing research papers with diverse data formats. Once the vector knowledge base is established, the model is fine-tuned using various combinations of embeddings to optimize performance.

### 3.2 Data Collection

Our project sources raw datasets from two reputable scientific repositories: the Journal of Machine Learning Research (JMLR) and arXiv. The data collection aspects of the research paper snapshot dataset from JMLR and arXiv are delineated in Table 14. These datasets encompass an array of metadata parameters, such as paper title, volume, authors, publication year, page count, and direct links to the PDF documents and associated code repositories, when such links are available.

**Table 14**

*Description of Data Collection* ( )

Aspect	Description
Why are we collecting data?	The JMLR and arXiv datasets are collected to support the dissemination and open access of high-quality scientific research within the machine learning and AI communities. They provide comprehensive coverage of the latest developments in the field.
How will the data help?	These datasets enable researchers and practitioners to access a vast array of research outputs, facilitating advancements in technology and methodology. By analyzing these papers, users can identify emerging trends, collaborate on new ideas, and develop a knowledge base for our application.
What should we do after collecting data?	After data collection, the documents should be preprocessed to ensure usability. This involves converting PDFs to text, segmenting text into manageable parts, and structuring the data into a searchable format. The dataset should be divided into subsets for building our knowledge base for RAG, with separate portions set aside for validation and testing to ensure robustness and reliability.

Detailed descriptions of the metadata (key variables), the duration of the data collection process, and the team members responsible for the data collection in both JMLR and arXiv are presented in Table 15 and Table 16.

**Table 15**

*Key Variables and Data Collection Timeline of JMLR*

Variable title	1 (title)	2 (volume)	3 (authors)	4 (year)	5 (pages)	6 (link)	7 (code)
Data type	Text	Text	Text	Integer	Integer	URL(String)	URL(String)
Collection method	Data is retrieved from the Journal of Machine Learning Research (JMLR) and arXiv's repositories. Papers are downloaded in PDF format, then processed using Python scripts that convert PDFs to text and extract metadata such as titles, authors, and publication years.						
Historical data exists?	JMLR has archives dating back to 2001, and arXiv has papers available from 1991 onwards. Both databases are continuously updated with new papers.						
Operational definition exists?	Operational definitions are established for both datasets to ensure consistent data collection. Metadata fields such as paper title, authors, publication year, volume, and links to PDF/code are uniformly extracted and stored. Each paper's data is handled individually to maintain data integrity and traceability.						
Data collectors	Divya Neelamegam			Sourabh Suresh Kumar			
Start date	26-Feb-2024			27-Feb-2024			
Due date	26-Feb-2024			27-Feb-2024			
Duration (in days)	1			1			

**Table 16**

*Key Variables and Data Collection Timeline of arXiv*

Variable title	1 (id)	2 (title)	3 (abstract)	4 (categories)	5 (doi)
Data type	Float	String	String	String	String
Collection method	Data is retrieved from the arXiv's repositories. Papers are downloaded and processed using Python scripts that use the metadata such as titles, URL, and publication years.				
Historical data exists?	arXiv has papers available from 1991 onwards and the database is continuously updated with new papers.				
Operational definition exists?	Operational definitions are established to ensure consistent data collection. Metadata fields such as paper title, authors, publication year, volume, and links to PDF/code are uniformly extracted and stored. Each paper's data is handled individually to maintain data integrity and traceability.				
Data collectors	Divya Neelamegam		Sourabh Suresh Kumar		
Start date	26-Feb-2024		27-Feb-2024		
Due date	26-Feb-2024		27-Feb-2024		
Duration (in days)	1		1		

The efficiency of the collection method is highlighted by the JMLR dataset, which consists of 2,894 paper entries, and the arXiv dataset, which boasts an impressive 2,426,574 entries out of which 25,173 papers on AI and ML, demonstrating the extensive range and depth of the data. These figures are illustrated in Figures 8 and 9. The arXiv dataset, in particular, encompasses a comprehensive collection of research papers pertaining to STEM fields.

**Figure 8**

*Count of JMLR Dataset*

(Original from Team 8)

```
df_jmlr.count()

title    2894
volume   2894
authors  2894
year     2894
pages    2894
link     2894
code     283
dtype: int64
```

**Figure 9**

*Count of arXiv Dataset*

(Original from Team 8)

```
df_data.count()

id      2426574
title   2426574
abstract 2426574
categories 2426574
doi     1166617
dtype: int64
```

**Figure 10**

*Sample data from JMLR dataset*

(Original from Team 8)

	title	volume	authors	year	pages	link	code	abstract
0	Joint Estimation and Inference for Data Integr...	Volume 23	['Subhabrata Majumdar', 'George Michailidis']	2022	53	<a href="https://www.jmlr.org/papers/volume23/18-131/">https://www.jmlr.org/papers/volume23/18-131/</a>	<a href="https://github.com/GeorgeMichailidis/JMMLE_code">https://github.com/GeorgeMichailidis/JMMLE_code</a>	Abstract\nThe rapid development of high-throug...
1	Debiased Distributed Learning for Sparse Part...	Volume 23	['Shaogao Lv', 'Heng Lian']	2022	32	<a href="https://www.jmlr.org/papers/volume23/18-4671/">https://www.jmlr.org/papers/volume23/18-4671/</a>	NaN	Abstract\nAlthough various distributed machine...
2	Recovering shared structure from multiple netw...	Volume 23	['Keith Levin', 'Assad Lodhia', 'Elizaveta Lev...']	2022	48	<a href="https://www.jmlr.org/papers/volume23/19-1056/">https://www.jmlr.org/papers/volume23/19-1056/</a>	NaN	Abstract\nIn increasingly many settings, data ...
3	Exploiting locality in high-dimensional Factor...	Volume 23	['Lorenzo Rimella', 'Nick Whiteley']	2022	34	<a href="https://www.jmlr.org/papers/volume23/19-2671/">https://www.jmlr.org/papers/volume23/19-2671/</a>	<a href="https://github.com/LorenzoRimella/GraphFilter-...">https://github.com/LorenzoRimella/GraphFilter-...</a>	Abstract\nWe propose algorithms for approximat...
4	Empirical Risk Minimization under Random Censo...	Volume 23	['Guillaume Ausset', 'Stephan ClÃ¢menÃ§on', F...]	2022	59	<a href="https://www.jmlr.org/papers/volume23/19-4501/">https://www.jmlr.org/papers/volume23/19-4501/</a>	NaN	Abstract\nWe consider the classic supervised ...
5	XAI Beyond Classification: Interpretable Neura...	Volume 23	['Xi Peng', 'Yunfan Li', 'Ivor W. Tsang', 'Hon...']	2022	28	<a href="https://www.jmlr.org/papers/volume23/19-4971/">https://www.jmlr.org/papers/volume23/19-4971/</a>	<a href="http://www.pengxi.me">http://www.pengxi.me</a>	Abstract\nIn this paper, we study two challeng...
6	Bayesian Multinomial Logistic Normal Models I...	Volume 23	['Justin D. Silverman', 'Kimberly Roche', 'Zac...']	2022	42	<a href="https://www.jmlr.org/papers/volume23/19-8821/">https://www.jmlr.org/papers/volume23/19-8821/</a>	<a href="https://github.com/jslive24/fido_paper_code">https://github.com/jslive24/fido_paper_code</a>	Abstract\nBayesian multinomial logistic-normal...
7	Deep Learning in Target Space	Volume 23	['Michael Fairbank', 'Spyridon Samothrakis', ...]	2022	46	<a href="https://www.jmlr.org/papers/volume23/20-0402/">https://www.jmlr.org/papers/volume23/20-0402/</a>	<a href="https://github.com/mikefairbank/dts_paper_code">https://github.com/mikefairbank/dts_paper_code</a>	Abstract\nDeep learning uses neural networks w...
8	Scaling Laws from the Data Manifold Dimension	Volume 23	['Utkarsh Sharma', 'Jared Kaplan']	2022	34	<a href="https://www.jmlr.org/papers/volume23/20-1111/">https://www.jmlr.org/papers/volume23/20-1111/</a>	<a href="https://github.com/U-Sharma/NeuralScaleID">https://github.com/U-Sharma/NeuralScaleID</a>	Abstract\nWhen data is plentiful, the test los...
9	Interpolating Predictors in High-Dimensional F...	Volume 23	['Florentina Bunea', 'Seth Strimas-Mackey', 'M...']	2022	60	<a href="https://www.jmlr.org/papers/volume23/20-1122/">https://www.jmlr.org/papers/volume23/20-1122/</a>	NaN	Abstract\nThis work studies finite-sample propo...

For this project use case, a total of 28,067 papers (25,173 from the arXiv STEM subset and 2,894 from JMLR) were processed to train and fine-tune the models, ensuring a robust and comprehensive foundation for analysis.

## Figure 11

*Sample data from arXiv dataset*

*(Original from Team 8)*

<b>id</b>	<b>title</b>	<b>abstract</b>	<b>categories</b>	<b>doi</b>
0 0704.0001	Calculation of prompt diphoton production cros...	A fully differential calculation in perturba...	hep-ph	10.1103/PhysRevD.76.013009
1 0704.0002	Sparsity-certifying Graph Decompositions	We describe a new algorithm, the $\$(k,\ell)\$$ ...	math.CO cs.CG	None
2 0704.0003	The evolution of the Earth-Moon system based o...	The evolution of Earth-Moon system is descri...	physics.gen-ph	None
3 0704.0004	A determinant of Stirling cycle numbers counts...	We show that a determinant of Stirling cycle...	math.CO	None
4 0704.0005	From dyadic $\$Lambda_{\{\alpha\}}$ to $\$Lambda_{\{a...$	In this paper we show how to compute the $\$L...$	math.CA math.FA	None

## **Scalable Data Storage**

In a fast growing field like AI and ML, the scope for research and publication is high. To handle this growing volume of data, Google Cloud Storage is deployed. The GCS is scalable based on the storage requirements, and the data is managed and organized using buckets.

## **Streamlined Data Ingestion**

The automated script to extract incremental data about newly published papers is developed and ingested into the GCS bucket for further preprocessing. An orchestration tool to handle the data ingestion can be employed to schedule and trigger the process on time at regular intervals.

### **3.3 Data Preprocessing**

The pre-processing of our raw datasets was conducted with a focus on preparing the data for subsequent analysis and modeling. Initial filtering targeted the selection of documents relevant to the fields of Machine Learning (ML) and Artificial Intelligence (AI) as shown in

Figures 12 and 13. This is accomplished by extracting records where the 'categories' attribute was marked as "cs.AI" or "cs.LG" from the arXiv dataset.

**Figure 12**

*Count of ML and AI papers in arXiv dataset*

(Original from Team 8)

```
[ ] df_data.count()

id      25173
title   25173
abstract 25173
categories 25173
doi     2762
dtype: int64
```

**Figure 13**

*Samples from ML and AI arXiv papers*

(Original from Team 8)

	<b>id</b>	<b>title</b>	<b>abstract</b>	<b>categories</b>	<b>doi</b>
0	0704.1274	Parametric Learning and Monte Carlo Optimization	This paper uncovers and explores the close r...	cs.LG	None
1	0704.1394	Calculating Valid Domains for BDD-Based Inter...	In these notes we formally describe the func...	cs.AI	None
2	0704.2010	A study of structural properties on profiles HMMs	Motivation: Profile hidden Markov Models (pH...	cs.AI	None
3	0704.2668	Supervised Feature Selection via Dependence Es...	We introduce a framework for filtering featu...	cs.LG	None
4	0704.3433	Bayesian approach to rough set	This paper proposes an approach to training ...	cs.AI	None

To facilitate the extraction of the actual papers, the arXiv URL is composed using the unique identifiers in the dataset, forming links such as "<https://arxiv.org/pdf/<id>.pdf>" shown in Figure 14. Through automated scripts, the papers are batch downloaded for processing. However, papers that had been withdrawn or inaccessible were identified and excluded from the dataset to maintain its integrity.

The core of our pre-processing workflow was the PDF to Text Conversion step, which involved using PyPDFLoader, which is capable of handling the inherent complexities of unstructured PDF documents. The conversion process is critical as it normalizes the data by stripping away non-textual elements and standardizing text representation across various

document formats. The diverse content types, such as equations, special formatting, and embedded graphics are handled, which often appear in scientific documents, ensuring a high-fidelity extraction of text.

The process for extracting tabular data from PDF documents employs advanced AI models and techniques to ensure accurate and efficient conversion of complex table structures into structured data. Initially, the PDF documents are processed using the pdf2image library, which converts each PDF page into image formats suitable for further analysis. These images are then analyzed using cutting-edge models with Table Transformer to identify and delineate table boundaries by detecting and extracting the coordinates of table regions within the images.

Once the table regions are identified, a dedicated table structure recognition model, from Hugging Face, is utilized to parse the identified table layouts. This model excels in interpreting the rows, columns, headers, and spanning cells, ensuring the data's structural integrity is maintained. Additionally, it integrates seamlessly with visual captioning techniques powered by Llama3, pytesseract, and the Unstructured API to further enhance the extraction of textual data embedded within images, such as table headers or annotations.

The extracted image data is subsequently converted into embeddings using Hugging Face models, creating vectorized representations that capture the semantic structure and content of the tables. These embeddings are stored in a Chroma Vector Store, enabling efficient similarity-based retrieval for downstream applications. This robust pipeline enhances the Multi-Modal RAG model by incorporating structured tabular data, enabling it to generate concise and contextually relevant answers while improving the overall data retrieval and response generation capabilities.

A detailed review of the pre-processed data samples in Figure 15 reveals the effectiveness of our cleaning protocol. Text extracted from PDFs is now consistently formatted for improved readability and uniformity.

## Figure 14

*Sample after Preprocessing*

*(Original from Team 8)*

df_data_final							
	id	title	abstract	categories	doi	paper_content	arxiv_url
0	0704.1274	Parametric Learning and Monte Carlo Optimization	This paper uncovers and explores the close r...	cs.LG	None	Keywords: Monte Carlo Optimization, Black-box ...	<a href="https://arxiv.org/pdf/0704.1274.pdf">https://arxiv.org/pdf/0704.1274.pdf</a>
1	0704.1394	Calculating Valid Domains for BDD-Based Inter...	In these notes we formally describe the func...	cs.AI	None	arXiv:0704.1394v1 [cs.AI] 11 Apr 2007Calcula...	<a href="https://arxiv.org/pdf/0704.1394.pdf">https://arxiv.org/pdf/0704.1394.pdf</a>
2	0704.2010	A study of structural properties on profiles HMMs	Motivation: Profile hidden Markov Models (pH...	cs.AI	None	arXiv:0704.2010v2 [cs.AI] 11 Dec 2008BIOINFO...	<a href="https://arxiv.org/pdf/0704.2010.pdf">https://arxiv.org/pdf/0704.2010.pdf</a>
3	0704.2668	Supervised Feature Selection via Dependence Es...	We introduce a framework for filtering featu...	cs.LG	None	Supervised Feature Selection via Dependence Es...	<a href="https://arxiv.org/pdf/0704.2668.pdf">https://arxiv.org/pdf/0704.2668.pdf</a>
4	0704.3433	Bayesian approach to rough set	This paper proposes an approach to training ...	cs.AI	None	Bayesian Approach to Rough Set \nTshilidzi Ma...	<a href="https://arxiv.org/pdf/0704.3433.pdf">https://arxiv.org/pdf/0704.3433.pdf</a>

This meticulous approach to pre-processing not only enhances the quality of the data but also paves the way for accurate and efficient downstream applications, such as semantic analysis and natural language processing. The processed dataset is poised for a rigorous exploration that can unveil novel insights and drive innovation in the rapidly evolving fields of ML and AI.

### ***Automated Preprocessing Pipelines***

The data pipeline is created for data ingestion and data preparation. The retrieved data in the GCS undergoes processing by extracting the paper content, and the data is cleaned and processed into chunks and vectors to be ingested into the RAG system.

### **3.4 Data Transformation**

#### ***Transformation of Pre-processed Datasets***

The pre-processed datasets were transformed into formats suitable for advanced analysis by employing various tools and scripts. This step involved the segmentation of text into manageable chunks, facilitating easier handling and processing. Additionally, semantic embedding generation was conducted, which is crucial for the subsequent stages of analysis.

### ***Samples from Transformed Datasets***

Here are some examples of how the datasets were transformed,

- **Text Chunking:** Original text from a technical paper "The application of deep learning models to autonomous driving has shown promising results..." was segmented into manageable chunks, such as "The application of deep learning models", "to autonomous driving", and "has shown promising results."
- **Semantic Embeddings:** From the chunk "The application of deep learning models", an embedding array might look like [0.865, -0.123, 0.908, ...], representing the semantic space of the text as shown in Figure 16.

### ***Selection and Testing of Embedding Models***

A baseline study was initially conducted to evaluate the performance of various embedding models on a subset of the PDFs. This study was crucial for identifying the most effective models that could provide the best semantic understanding and representation of the text. The models considered in this phase included,

- MiniLM-L6-v2
- Sentence-transformers/all-MiniLM-L6-v2
- Thenlper/gte-large
- BAAI/bge-large-en-v1.5

Systematic testing of these shortlisted models followed, involving detailed analysis of performance metrics. This systematic approach ensured that each model was evaluated under consistent conditions, providing reliable data for comparison.

## **Evaluation and Advantages**

The evaluation process highlighted several advantages that influenced the selection of these specific models such as,

- MiniLM-L6-v2 and Sentence-transformers/all-MiniLM-L6-v2 were noted for their high efficiency in generating compact yet informative embeddings, which are essential for applications requiring quick retrieval and minimal storage usage.
- Thenlper/gte-large was particularly effective in handling more complex semantic structures, making it suitable for detailed textual analysis and applications involving nuanced language understanding.
- BAAI/bge-large-en-v1.5 excelled in precision and adaptability to various text genres, showcasing robust performance across diverse datasets.

**Figure 15**

*Text Embeddings Generated Using all-MiniLM-L6-v2*

*(Original from Team 8)*

```
array([[-0.17309807, -0.10922703,  0.11997147, ...,  0.11830396,
       -0.079504,  -0.11430095], [
[-0.18128201, -0.14044508,  0.1545095 , ...,  0.10432599,
       0.00846069, -0.04908906], [
[-0.20292331, -0.10011439,  0.12901923, ...,  0.08570072,
       -0.02853273, -0.10067203], [
...,
[[ -0.17451054, -0.00692823,  0.17774832, ...,  0.04262985,
       -0.02326536, -0.09262711], [
[-0.03157317, -0.13334677,  0.07569563, ...,  0.01338292,
       -0.08685192, -0.1011356 ], [
[-0.03780447, -0.08106704,  0.04931236, ...,  0.01181052,
       -0.11106861, -0.05282832]], dtype=float32)
```

These models were chosen based on their ability to significantly enhance the semantic analysis capabilities of the system, ensuring that the transformed data not only retains but also amplifies the inherent value and insights present in the original texts. The selected models effectively balance computational efficiency with advanced analytical prowess, making them ideal candidates for integrating into the larger data processing framework.

### **3.5 Data Preparation**

We took advantage of two prestigious sources of datasets for our machine learning model development: the Journal of Machine Learning Research (JMLR) and the arXiv open access repository. In order to provide a solid foundation for training and testing our algorithms, this integration aims to increase the dataset's diversity and size. The combined dataset was divided into separate sets for training, validation, and testing after the preparatory stages to harmonize the data from both sources were completed. 10% was used for validation, 10% for testing, and 80% of the partitioning was done for training purposes. Model development is approached in a balanced manner since this distribution is in line with accepted industry procedures.

While enabling sufficient model tuning and performance evaluation, an 80/10/10 split facilitates effective model training. The model's capacity to generalize is enhanced by the huge training set, which helps it to extract detailed patterns and characteristics from the data. The validation set is essential for fine-tuning the model, which includes choosing features and tweaking hyperparameters to improve performance and prevent overfitting. Lastly, a solid foundation for assessing the model's efficacy and its practical applications is provided by the test set, which is kept entirely apart from the training and validation data. Our predictive modeling's quality and integrity depend heavily on this systematic approach to dataset segmentation.

As illustrated in Figure 17, the extraction process was meticulously executed, resulting in the successful retrieval of content from a significant number of PDF files. These files, sourced from the arXiv repository, are clearly indicated by their structured naming convention, which includes identifiers and extraction status. Each PDF's content was safely archived in the specified directory, ensuring the integrity and accessibility of the data for the subsequent steps of our analysis. This comprehensive extraction lays the groundwork for the next phase, where the data

will be split according to an 80/10/10 ratio for training, validation, and testing purposes, respectively. This strategic partitioning is critical for the model's learning and validation cycles, setting the stage for accurate performance evaluation.

## Figure 16

*Extraction of PDF*

*(Original from Team 8)*

```
PDF content successfully saved to: extracted_arxiv_papers/AI/0704_1394_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0704_2010_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0704_3433_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0704_3515_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0704_3905_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0705_0197_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0705_0693_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0705_0734_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0705_0761_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0705_0969_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0705_1031_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0705_1209_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0705_1244_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0705_1309_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0705_2235_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0705_2236_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0705_2310_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0705_3360_extracted.pdf
PDF content successfully saved to: extracted_arxiv_papers/AI/0705_4302_extracted.pdf
```

Throughout the project's development, notable progress was achieved in the area of data preparation, with an emphasis on successfully optimizing and employing vector storage solutions to support machine learning endeavors. Two significant vector databases, FAISS and ChromaDB, were implemented. For the project to be successful, these systems were essential for effectively storing and retrieving large datasets.

In parallel, ChromaDB was customized to fit specific scenarios in the project, and it was fine-tuned to guarantee peak performance under various loads and data complexity. The performance and scalability requirements of the data storage framework were specifically addressed by this adaptation. Queries are augmented to improve their context before being processed by a large language model, to generate an augmented prompt that optimizes the retrieval or understanding of information from the document set. This adaptation aims to meet specific performance and scalability requirements for handling large scale data.

### ***Implementation of FAISS for Efficient Text Retrieval***

The model creates vector embeddings from retrieved text by using the sentence-transformers package. This is made possible by the pre-trained model all-MiniLM-L6-v2, which is renowned for producing high-quality embeddings for a range of NLP tasks. Text segments that are taken out of a PDF document are used to create the embeddings. This conversion is crucial because it turns the unprocessed text data into a numerical form that can be used for machine learning applications, especially the FAISS framework's similarity searches.

An instance of an FAISS index (IndexFlatL2) is built in order to handle the embeddings for effective retrieval. This index is intended to manage L2 (Euclidean) distance computations, which are frequently used in embedding space nearest neighbor searches. The produced embeddings are then kept in this FAISS index, making it possible to retrieve the closest neighbors quickly and effectively. This feature is essential for later processes like retrieving documents, grouping documents, or doing similarity searches according to certain user queries.

The system defines functions to retrieve documents based on user queries. These functions work by first encoding the query into embeddings similar to the document embeddings stored in the FAISS index. The system then utilizes the index to identify and retrieve the nearest matching document embeddings. Following retrieval, there are provisions to format the retrieved documents, potentially for display in a user interface or for further analytical processing. This structured retrieval mechanism leverages the indexed embeddings to provide responsive and relevant document search capabilities based on user input. The contents for the database is shown in figure 18.

**Figure 17**

*Database Content (FAISS)*

*(Original from Team 8)*

6.95831478e-02	1.39677408e-03	7.67233148e-02	6.10977896e-02
5.69548011e-02	-4.09566499e-02	3.53050269e-02	-3.85011584e-02
-1.09626211e-01	3.07551380e-02	-3.46512012e-02	-7.69721577e-04
-3.74835730e-02	-4.07076403e-02	-4.59438935e-02	-6.22254871e-02
7.38523155e-02	-6.11172244e-02	1.61684193e-02	1.49084663e-03
-5.46306372e-02	2.72499248e-02	6.50121868e-02	7.49438927e-02
1.13536495e-04	-7.66016468e-02	5.20878285e-02	7.74383843e-02
1.25429546e-02	5.00459224e-02	-7.79786110e-02	5.00856759e-03]
Vector 205: [ 1.02623925e-02 -9.92411096e-03 -5.65223396e-02 -6.77827001e-02			
-3.38532627e-02	-1.48033015e-02	-2.71109492e-02	8.37308988e-02
1.04294689e-02	5.25813065e-02	2.63272785e-02	-4.95944470e-02
5.28045148e-02	2.55415170e-03	5.61943725e-02	3.60797122e-02
8.23723525e-02	-4.14233282e-03	-5.58020547e-02	4.57204022e-02
8.85267779e-02	-8.75765085e-02	-5.38998842e-02	-1.71920229e-02
-8.65791515e-02	5.05322516e-02	-1.81591772e-02	2.28643660e-02
1.46074919e-02	-9.24004614e-02	6.63700327e-02	3.73022780e-02
7.98420087e-02	3.17078978e-02	-6.76034391e-02	1.58158150e-02
-3.74954529e-02	-4.89253998e-02	-3.59490737e-02	3.47341895e-02
-2.18935739e-02	-3.24568935e-02	1.25825463e-03	-6.34075105e-02
2.78243534e-02	2.14050915e-02	-7.62529969e-02	3.38691543e-03
-3.79195251e-02	1.37458313e-02	-1.08970426e-01	-8.03260691e-03
-5.08707715e-03	8.70201066e-02	1.49792368e-02	-5.34151495e-02
-1.48088289e-02	-5.02112880e-02	-3.76931392e-03	-4.34083007e-02
-8.08716789e-02	-7.11049279e-03	-1.66401621e-02	4.61530760e-02
9.94477496e-02	-2.11921539e-02	-2.18072487e-03	-3.57546322e-02
-2.43284330e-02	2.00558510e-02	2.07258221e-02	-8.62163976e-02
2.99421512e-02	-6.45457283e-02	6.03174567e-02	-9.10575408e-03
-4.20915335e-02	-2.23346204e-02	4.25277986e-02	1.20484820e-02
3.88973467e-02	4.96218279e-02	-4.00774851e-02	3.35434712e-02
-3.84700038e-02	-2.08120123e-02	-6.75305277e-02	-3.33241820e-02
1.09275319e-01	-7.61247426e-02	-1.17024407e-03	2.70170998e-02
4.19453532e-03	3.99155617e-02	5.47384210e-02	9.13469195e-02

### ***Implementation of ChromaDB for Efficient Text Retrieval***

In order to facilitate the effective storage of vector data within ChromaDB, the model describes the creation of a ChromaVectorStore that is configured with a chroma collection. Large vector data quantities, which are essential for retrieval operations in machine learning applications, require this configuration.

A StorageContext is implemented with default settings to allow storage operations and offer a consistent framework for managing and storing vector data. By ensuring that the data processing procedures adhere to established operational standards, the context settings improve consistency and dependability. Using an embedding approach, which converts textual data into vector representations, documents are encoded as part of the document processing process. After that, these embeddings are kept in the Chroma database so retrieval tasks can easily access them. The document processing method also includes a transformation stage that preprocesses text

using a splitter. This step may involve breaking up the text into manageable chunks or improving the textual data for a more successful embedding.

First, in order to guarantee current session integrity, the retrieval setup entails reading the stored vector data and reinitializing the connection to the Chroma database. After that, the vector store is set up with an embedding model to facilitate effective data retrieval processes, and a VectorStoreIndex is produced from there. For real-time application performance, this index plays a critical role in providing instantaneous access to vector data.

Next, the `as_retriever()` method is used to turn the index into a retriever. The system is now able to extract vectors based on particular queries and perform effective searches thanks to this conversion. In order to improve the system's usability in dynamic contexts, the retriever configuration is made to handle complex query needs. This allows for exact and quick recovery of pertinent data. Figure 19, Figure 20 and figure 21 depicts the validation, training and testing size sample data

**Figure 18**

## *Validation data example*

*(Original from Team 8)*

1, SVM is able to predict peace and conflict with 79% and 75%, respectively. The corresponding results for NN are 74% and 76%, respectively for peace and \conflict. The combined results of correct predictions are 79% for SVM and 74% for NN. \n \n \n ',  
 'less democratic state plays a determinant role for an occurrence of conflict [Oneal and Russett, 1999]. Dependency is measured as the sum of a country's import and export with its partner \ndivided by the Gross Domestic Product of the stronger country. It is a continuous variable that \nmeasures the level of economic interdependence (trade as a portion of a state's gross \ndomestic product) of the less economically dependent state in the dyad. \n \nIII. METHOD \nA. Neural networks \nNeural network requires selecting the best architecture to give good classification results. The best combination of the number of hidden units, activation function, training algorithm and training \ncycles that can result in a network which is able to generalise the test data in the best possible way \nis searched for during the model selection process. This helps to avoid the risk of over\entraining. A multi-layer perceptron (MLP) trained with the scaled conjugate gradient (SCG) method \n(Moller, 1993) neural network was used to classify the MID input data. Logistic and hyperbolic \nactivation functions for the output and hidden layers respectively, M = 10 number of hidden units \nand 100 training cycles resulted in an optimal architecture. \n \nB. Support vector machines \nSVM employs a method of mapping the input into a feature space of higher dimensionality and \then finds a linear separating hyperplane with maximum margin of separation. There are various \nmapping or kernel functions in use, the most common of which are linear, polynomial, radial basis \nfunction (RBF) and sigmoid. The choice of a kernel function depends on the type of problem at \nhand and the RBF:  $O, ||x|| \exp(-\gamma x^T K2nj j i) > y - y_0$  can handle non-linear data better than the ',

## Figure 19

Training data example

(Original from Team 8)

```
l of Information Sciences and Engineering, Australian National University. http://csl.anu.edu.au/~bartlett/papers/BartlettBaxter-Nov99.ps.gz.  
Bartlett, P. L., & Baxter, J. (2001). Estimation and approximation bounds for gradient-based reinforcement learning. Journal of Computer and Systems Sciences, 62. Invited Paper: Special Issue on COLT 2000.  
Barto, A. G., Sutton, R. S., & Anderson, C. W. (1983). Neuron-like adaptive elements that can solve difficult learning control problems. IEEE Transactions on Systems, Man, and Cybernetics, 13, 834-846.  
Baxter, J., Bartlett, P. L., & Weaver, L. (2001). Gradient ascent algorithms and experiments with infinite-horizon, policy-gradient estimation. Journal of Artificial Intelligence Research, 14.  
Baxter, J., Tridgell, A., & Weaver, L. (2000). Learning to play chess using temporal-differences. Machine Learning, 40(3), 243-263.  
Bertsekas, D. P., & Tsitsiklis, J. N. (1996). Neuro-Dynamic Programming. Athena Scientific.  
Bertsekas, D. P. (1995). Dynamic Programming and Optimal Control, Vol II. Athena Scientific.  
Breiman, L. (1966). Probability. Addison-Wesley.  
Cao, X.-R., & Wan, Y.-W. (1998). Algorithms for Sensitivity Analysis of Markov Chains Through Potentials and Perturbation Realization. IEEE Transactions on Control Systems Technology, 6, 482-492.  
Dudley, R. M. (1989). Real Analysis and Probability. Wadsworth & Brooks/Cole, Belmont, California.  
Glynn, P. W. (1986). Stochastic approximation for monte-carlo optimization. In Proceedings of the 1986 Winter Simulation Conference, pp. 356-365.  
Glynn, P. W. (1990). Likelihood ratio gradient estimation for stochastic systems. Communications of the ACM, 33, 75-84.  
Glynn, P. W., & L'Ecuyer, P. (1995). Likelihood ratio gradient estimation for regenerative stochastic recursions. Advances in Applied Probability, 27, 4 (1995), 1019-1053.  
Ho, Y.-C., & Cao, X.-R. (1991). Perturbation Analysis of Discrete Event Dynamic Systems. Kluwer Academic, Boston.  
Jaakkola, T., Singh, S. P., & Jordan, M. I. (1995). Reinforcement Learning Algorithm for Partially Observable Markov Decision Problems. In Tesauro, G., Touretzky, D., & Leen, T. (Eds.), Advances in Neural Information Processing Systems, Vol. 7. MIT Press, Cambridge, MA.  
Kimura, H., & Kobayashi, S. (1998a). An analysis of actor/critic algorithms using eligibility traces:  
Reinforcement learning with
```

## Figure 20

Test data example

(Original from Team 8)

```
'A NEWGENERALMETHOD TO GENERATE RANDOMMODALFORMULAE\nand /BE with probability /BE /BP /BF); rndpropnum(d,p,K) selects randomly the number of propositions\nnatoms per clause /C8 according to the /C9 /CS /B7 /BD /BN /C3 k-th distribution in /D4 (e.g, if /Cs is /BD, /C3 is /BE and /D4 is [[[],[0,1,0],[0,1,0,\n[[1,0][0,1,0]]]], it returns /BD deterministically); rndsign() selects randomly either the positive or negative sign with the qual probability; norepeated atoms in\n/C1) checks if the clause /BV /D0 contains no repeated atom; Sort(C1) returns the clause /BV /D0 sorted according to some criterium; rndpropositional atoms\n() selects with uniform probability one of the /C6 propositional atoms /BT/CX; rndbox() selects with uniform probability one of the /D1 indexed boxes /B\n/D6. When eliminating duplicated atoms in a clause, we take care not to disturb these probabilities by first determining the "shape" of a clause (rows 8-\nFigure 13), and only then instantiating that with propositional variables (rows 10-16 in Figure 13). If a clause has repeated atoms, either propositional or\ndigital, the instantiation is rejected and another instantiation of the shape is performed. If we did not take care in this way we would generate too few\n"small" atoms because there are fewer small atoms than large atoms, resulting in a greater chance of rejecting small atoms because of repetition.\nThe elimination of duplicated atoms in a clause is not only a matter of elimination of redundancies, but also of elimination of a source of flaws. In fact, it\nmight generate top-level clauses like /BM/BM/BM /CM /B4 /BM /BE/BD/nB4 /BT/BD/nCN /BM /BT/BD/nB5 /CN /BM /BE/BE/nB4 /BT/BE/nCN /BM /BT/BE/nB5/B\n/CM /BM /BM, which would make the whole formula inconsistent.\nExample 5.1 We try to guess a parameter set by which the new random generator can potentially\ngenerate the following CNF/BE/D1 formula /B3:/B4 /BM /BT/BE/nCN /BE/BD/nB4 /BM /BT/BG/nCN /BM /BE/BD/nBT/BD/nB5 /CN /BE/BD/nB4 /BM /BT/BD/nCN
```

## 3.6 Data Statistics

JMLR

Figure 21 portrays the non-null data entries across various columns in a dataset. A striking uniformity is observed across most columns, such as 'title', 'volume', 'authors', 'year', 'pages', and 'link', each displaying a high count close to the maximum of 3000 entries, suggesting a well-maintained and complete dataset in these aspects. However, a stark contrast is noted in the 'code' column, which exhibits a significantly lower count, indicating that the associated data is sparse or less frequently documented in this collection. This discrepancy might point to fewer publications providing directly accessible code or it could signal a data collection gap that

warrants further investigation. The visualization effectively underscores the areas where data is robust and where it may require enrichment or further research to achieve completeness.

**Figure 21**

*Data Counts Per Column in JMLR* (Original from Team 8)

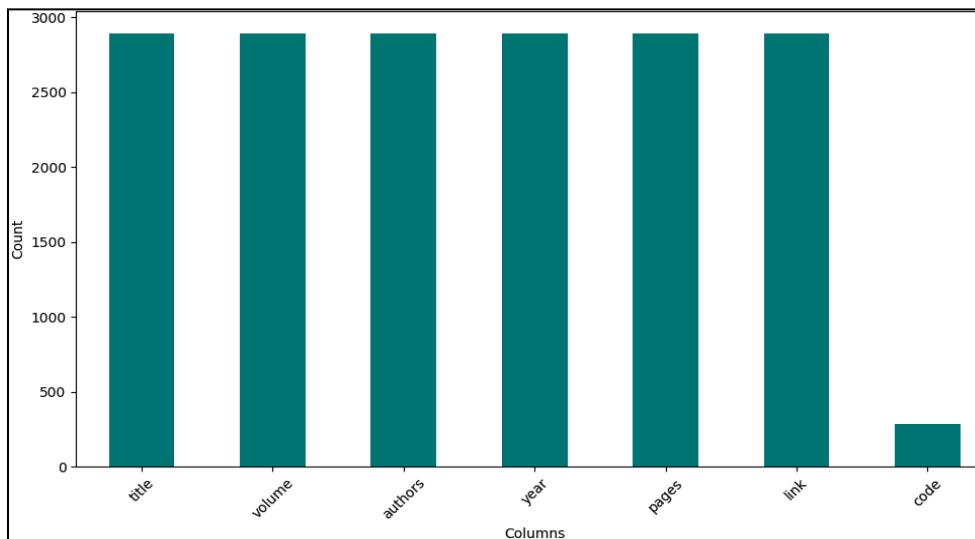


Figure 22 delineates the trend in the average number of pages per publication from the year 2000 through 2020. Initially, the graph exhibits some fluctuations, with a notable dip around the early 2000s. However, from around 2015, there is a clear upward trajectory, showing a significant increase in the average number of pages per publication. This could suggest that more comprehensive research with in-depth discussions has become more prevalent in recent years. The ascending trend in later years might reflect changes in publishing practices or norms, possibly due to the increasing complexity of topics being researched or a shift towards more detailed documentation in scholarly communication. This analysis could be pivotal for academic publishers and researchers alike, as it may influence the expected length and comprehensiveness of future scholarly documents.

**Figure 22**

*Average Number of Pages per Year*      *(Original from Team 8)*

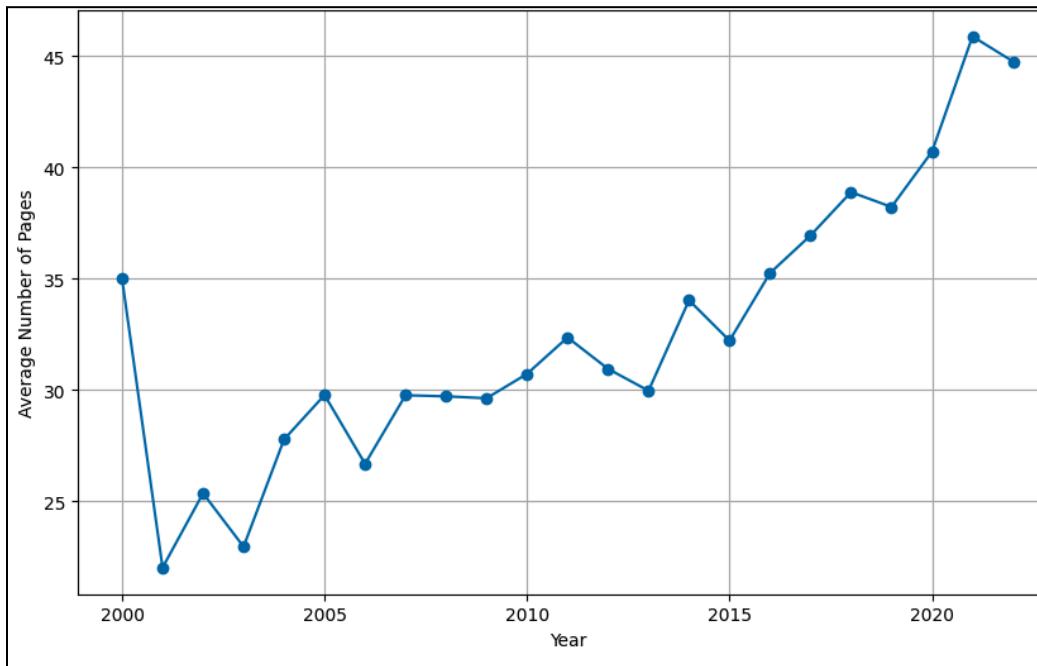


Figure 23 depicts the non-null data entry count across different columns in a dataset. The columns 'id', 'title', 'abstract', and 'categories' exhibit a consistently high level of data completeness, each with counts approaching the 10,000 mark, suggesting these fields are well-populated and likely mandatory for data entry in this context. In stark contrast, the 'doi' column displays a significantly lower count, indicating that Digital Object Identifiers are either less commonly assigned or recorded within this particular dataset. The discrepancy could reflect on the nature of the data collection process or perhaps the availability of DOIs in the source material. This visualization underscores the disparity in data availability across different informational categories and could be a point of consideration for efforts aimed at data standardization or completeness in future research documentation processes.

**Figure 23**

*Data Counts per Column in arXiv*

*(Original from Team 8)*

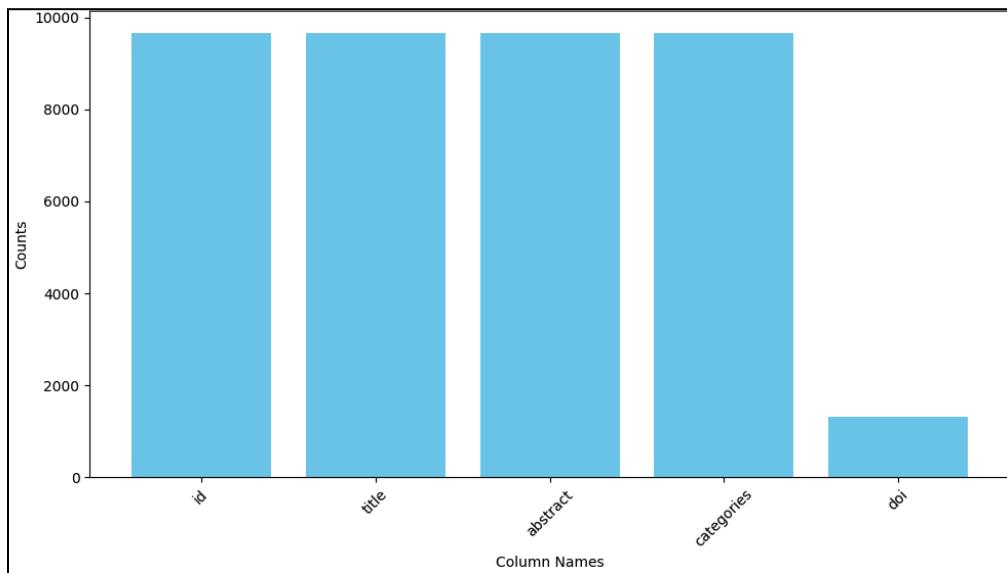


Figure 24 illustrates the distribution of the top 10 most prolific categories. The 'astro-ph' category dominates, indicative of a robust engagement with astronomical physics within the academic community. Following closely are the 'hep-ph' and 'quant-ph' categories, suggesting significant research activity in high-energy physics and quantum physics, respectively. Categories such as 'hep-th', 'cs.CV', and 'cond-mat.mtrl-sci' also show a strong presence, underscoring their relevance in contemporary scientific discourse. Towards the lower end of the top 10, 'math.AP', 'gr-qc', and 'astro-ph.GA' reflect a noteworthy, albeit lesser, concentration of publications. This distribution not only reflects the focus areas of current scientific research but also may guide future funding, research, and educational efforts towards these thriving domains.

**Figure 24**

*Top 10 Authors by Name of Publication*

*(Original from Team 8)*

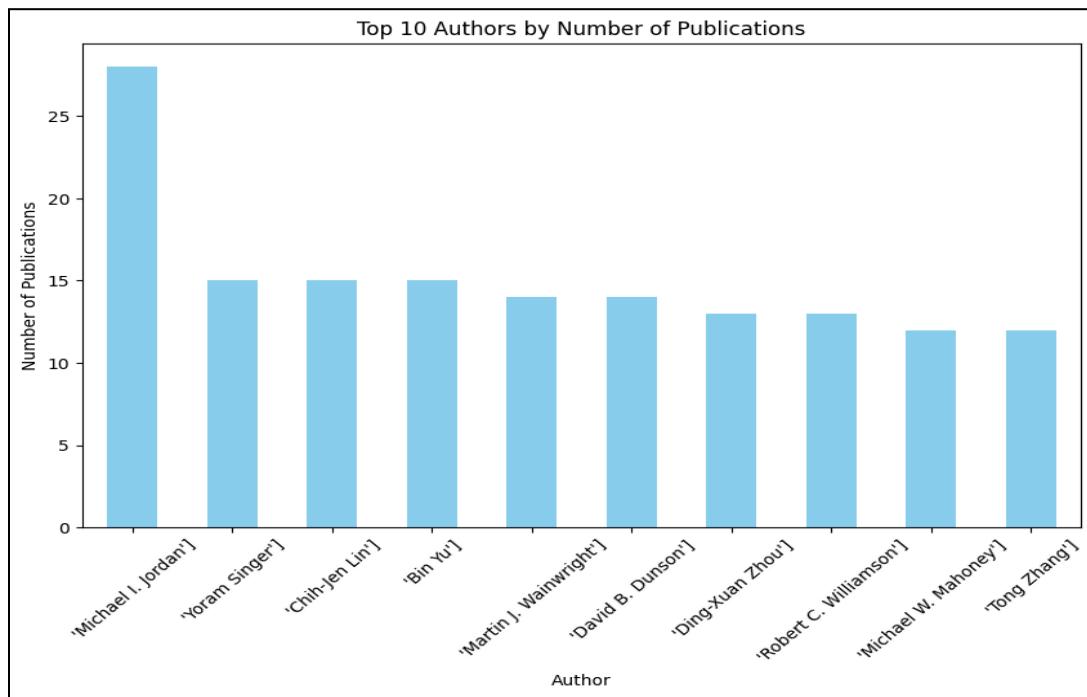


Figure 25 offers a comparative analysis of the presence of Digital Object Identifiers (DOIs) in publications. It is immediately apparent that the number of publications with DOIs (represented in green) is roughly equal to those without (represented in red), highlighting the fact that a significant portion of the research output in this dataset is digitally traceable through DOIs. This trend underscores the increasing adoption of DOIs in academic publishing, which aids in the digital cataloging and accessibility of scholarly work. However, the comparable magnitude of publications lacking DOIs points towards a considerable volume of research that may not be as readily discoverable in the digital domain, potentially impeding citation and academic dissemination. The near-equal distribution prompts a discussion about the importance of digital indexing in enhancing the visibility and impact of scholarly publications.

**Figure 25**

*Presence of DOI*

(Original from Team 8)

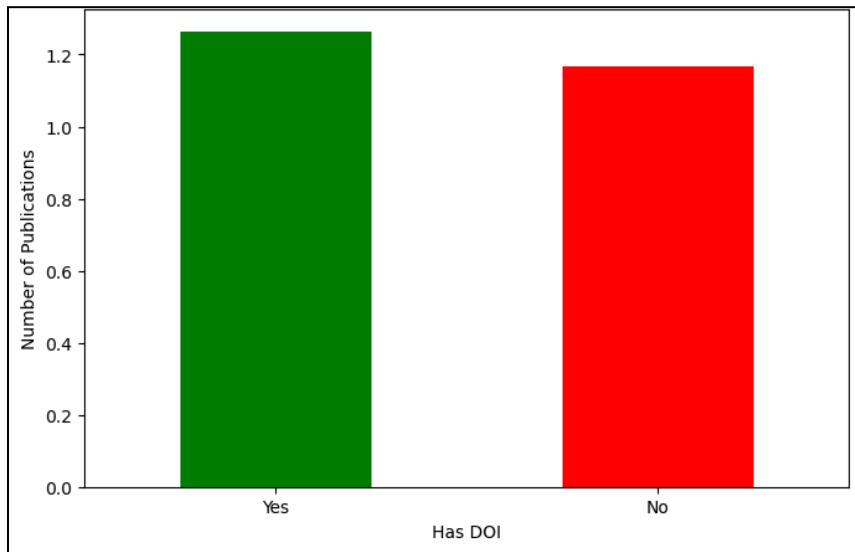
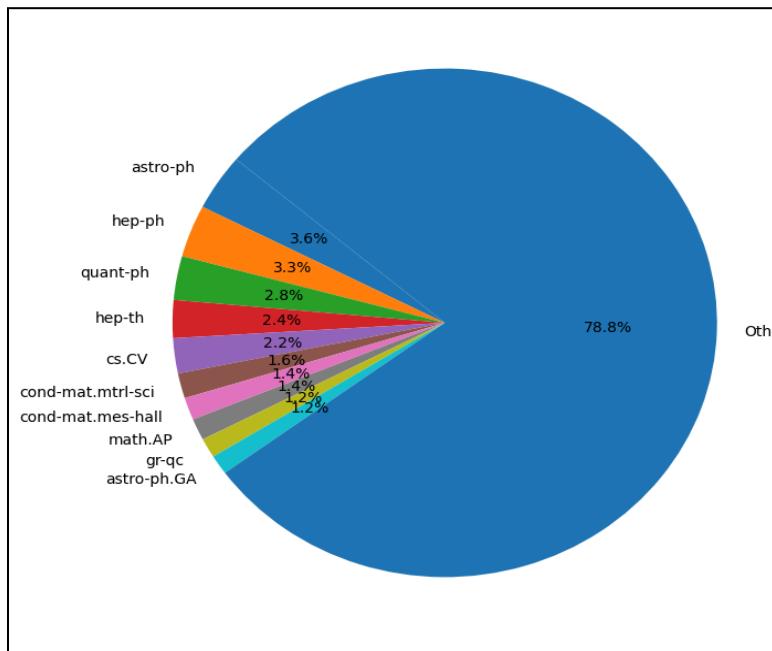


Figure 26 represents the distribution of scientific publication categories from a dataset processed according to specified code. A dominant share, represented by a vast blue section, is allocated to the 'Other' category, encompassing 78.8% of the dataset. This large portion indicates a high diversity of categories within the publications that are not among the top individual categories highlighted. The remainder of the chart is divided into smaller segments, each denoting specific fields of study with 'astro-ph' leading these individual categories, followed by 'hep-ph' and 'quant-ph'. The visualization emphasizes the concentration of publications in the 'astro-ph' category while also showcasing the breadth of topics covered in the dataset, as indicated by the significant 'Other' category. This suggests a diverse range of research interests and contributions outside the leading individual categories, highlighting the interdisciplinary nature of the scientific research represented in the dataset.

**Figure 26**

*Pie chart of Publication of Categories*

(Original from Team 8)



## 4. Model Development

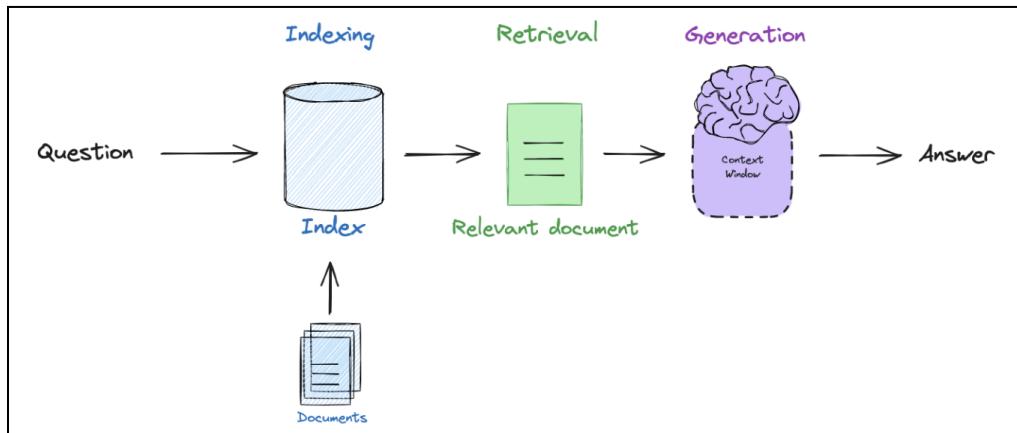
### 4.1 Model Proposals

Limited training data hinders large language models (LLMs) from accessing the latest information and internal knowledge. Retrieval-augmented generation (RAG) tackles this issue by connecting LLMs to external data sources. RAG works by first understanding a user's query, then retrieving relevant documents and finally feeding these documents along with the query to the LLM. This allows the LLM to generate a response based on the most up-to-date context. The basic RAG flow is illustrated in Figure 27.

**Figure 27**

*Basic RAG Flow*

*(Medium Article)*



In this section, we delve into the detailed descriptions and specifications of the model (LLM) configurations with the RAG approach. Each of these provides a comprehensive overview, outlining the unique features, approaches, and technical nuances that distinguish our models.

#### ***Evaluating Optimization Strategies for Retrieval-Augmented Generation***

Table 17, likely focuses on comparing different techniques or strategies used to optimize the performance of Retrieval-Augmented Generation (RAG) models. This typically summarizes

experiments aimed at improving the retrieval or generation process of RAG models. The optimization strategies could involve tuning various aspects of the system, such as the retrieval mechanism, the model architecture, or training procedures.

**Table 17**

*Evaluating Optimization Strategies for RAG* *(Original from Team 8)*

<b>Evaluating Optimization Strategies for Retrieval-Augmented Generation(RAG)</b>	
<b>Text Splitters</b>	‘CharacterTextSplitter’ ‘RecursiveCharacterTextSplitter’ ‘SemanticChunker’
<b>Data Segmentation</b>	chunk size:500 800 1000
<b>Embedding Models</b>	‘gte-large-en-v1.5’ ‘Snowflake-arctic-embed-l’ ‘bge-large-en-v1.5’
<b>Vector Stores</b>	FAISS ChromaDB Weaviate KDB AI

**Data Segmentation:** This experiment is conducted to identify optimal chunk size for building RAG for our dataset. The Baseline model is kept constant with varying chunk sizes to evaluate the impact on RAG performance. The goal is to identify the effective data segmentation size for efficient computation and retrieval of our system.

- The baseline model is built using the ‘CharacterTextSplitter’ and the standard embedding model, such as 'Alibaba-NLP/gte-large-en-v1.5' with LLaMa 2.
- Dynamic Chunks: The RAG model was evaluated using various chunk sizes, including 500, 800, and 1000, in order to observe their impact on retrieval effectiveness and processing time.

**Text Splitter Variations:** To determine the appropriate text splitter, the baseline model remains fixed, while multiple text splitters are used to assess their impact on RAG performance. The ‘CharacterTextSplitter’ divides the context based on the characters, whereas the ‘RecursiveCharacterTextSplitter’ further divides the chunks recursively if the sentence is split through the middle of the context to preserve the meaning. ‘SemanticChunker’ is an advanced method that employs natural language processing techniques to understand and segment the context in an effective manner.

- The baseline model is built using the standard embedding model, such as 'Alibaba-NLP/gte-large-en-v1.5' with LLaMa 2 and a chunk size of 500.
- Dynamic Text Splitter: The RAG model was evaluated using various text splitters such as ‘CharacterTextSplitter’, ‘RecursiveCharacterTextSplitter’ and ‘SemanticChunker’ in order to observe their impact on retrieval effectiveness and processing time.

**Embedding Model Variations:** This phase involves testing different embedding models to identify their capabilities in our dataset to retrieve and generate information. This gives insight into the contribution of an effective embedding model to the performance of the RAG system. The embedding models are selected based on the MTEB Leaderboard on Hugging Face. The model ‘gte-large-en-v1.5’ is trained on 434 million parameters with embedding dimensions of 1024 and has the capability to process sequences up to 8192 tokens. Similarly, ‘bge-large-en-v1.5’ and ‘snowflake-arctic-embed-l’ are developed on 335 and 334 million parameters respectively.

- The baseline model is built using the standard text splitter 'RecursiveCharacterTextSplitter' with LLaMa 2 and a chunk size of 500.

- **Embedding Models:** The RAG model was evaluated using different embedding models such as 'Alibaba-NLP/gte-large-en-v1.5', 'Snowflake/snowflake-arctic-embed-l', and 'BAAI/bge-large-en-v1.5' in order to observe their impact on retrieval effectiveness and processing time.

**Vector Store :** The experiment involved comparing different vector stores—FAISS, ChromaDB, Weaviate and KDB AI—based on their performance in handling embeddings derived from text data extracted from PDFs. The embeddings were created using the Sentence Transformer model (all-MiniLM-L6-v2), and various vector stores were evaluated on their search efficiency and similarity measurements.

In a detailed comparison of vector stores, FAISS and ChromaDB emerged with distinctive advantages for managing and searching text-derived embeddings. FAISS demonstrated exceptional speed and efficiency in search operations, clocking some of the fastest search times among the competitors. This rapid retrieval capability makes it ideal for scenarios where quick response times are crucial, such as in real-time user interactions or large-scale data processing environments. Its performance does not significantly degrade with larger datasets, which speaks to its scalability and robustness in various applications.

On the other hand, ChromaDB stood out for its superior relevance in search results, as evidenced by its consistently higher similarity scores compared to other vector stores. This indicates that ChromaDB is particularly effective at retrieving documents that are more relevant to the query, a critical feature for applications in which precision and accuracy of retrieved information are paramount, such as academic research or detailed content discovery platforms. Additionally, ChromaDB offers ease of integration and usability, which can significantly reduce

the complexity and time involved in deploying and maintaining search systems in production environments.

Overall, while FAISS excels in speed and scalability, ChromaDB offers superior accuracy and ease of use, making each suitable for different requirements and needs. Their respective strengths highlight the importance of choosing a vector store based on specific performance metrics and goals.

To maintain scalability in vector stores with expanding data and user demands, here are some of the key techniques that can be employed. Distributed vector databases like Faiss or Annoy enable efficient handling of large-scale data through sharding and indexing across multiple machines. Load balancing optimizes resource utilization by distributing queries evenly across nodes. Data partitioning enhances query performance by organizing data into manageable segments, reducing the search space. Indexing strategies like HNSW and PQ accelerate retrieval speeds through optimized data indexing and access. Finally, auto-scaling dynamically adjusts resources to handle varying loads, ensuring optimal performance during peak and off-peak periods. Together, these strategies create a robust framework for managing scalable vector stores in demanding environments.

### ***RAG Agent with LlaMa-3: Combined Approach using Adaptive RAG, Corrective RAG & Self-RAG***

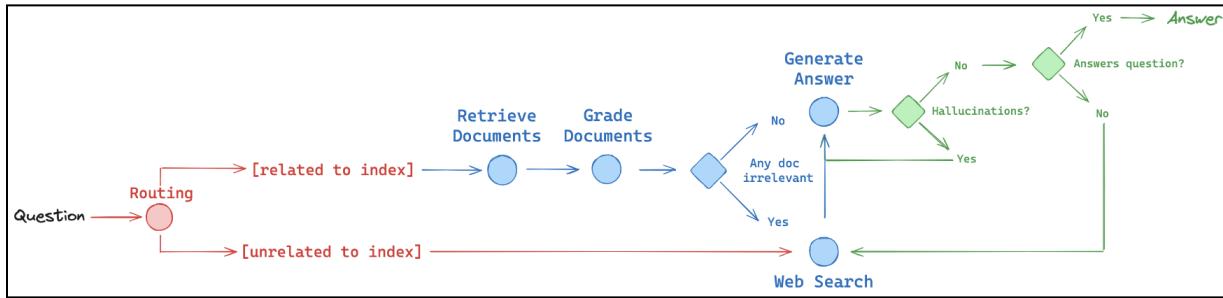
A reliable RAG agent leveraging LangGraph, LlaMa-3, Chroma for vector storage, and Tavily for optimized web search is proposed here. Three different approaches to Retrieval Augmented Generation are formulated together - Adaptive RAG, Corrective RAG, and Self-Reflective RAG. This approach utilizes LangGraph, a framework specifically designed for constructing language-based models. LangGraph provides tools and functionalities tailored for

building agents with reasoning capabilities. Refer Figure 28 for a comprehensive workflow of the configuration.

**Figure 28**

*RAG Agent Workflow*

(Medium Article)



Agents utilize language models as reasoning engines. This allows them to dynamically determine the most appropriate actions and their order based on the situation. Key Component include:

- Planning: Agents decompose complex tasks into a series of smaller, achievable sub-goals. This facilitates a step-by-step approach to achieving the overall objective.
- Memory: Agents maintain memory capabilities to inform their decision-making. This memory can be short-term, storing recent interactions like chat history, or long-term, utilizing external storage solutions like vector stores to retain past experiences.
- Tool Usage: These agents can extend their capabilities by employing various tools. This enables them to adapt to different situations and leverage external resources effectively.

Tavily Search API caters specifically to AI agents, unlike general search engines like Google or Serp. It prioritizes fast, accurate results for AI developers, enabling their creations to make informed decisions based on real-time information.

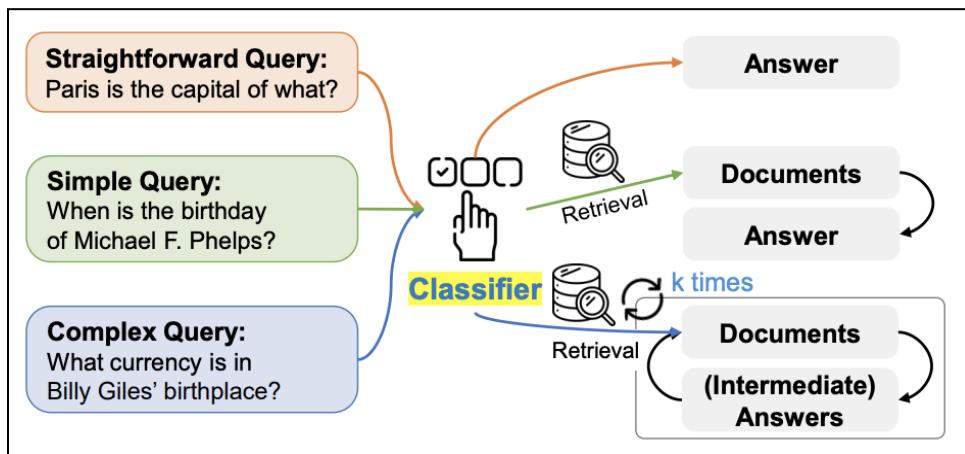
Adaptive RAG is a method which combines query analysis with active / self-corrective RAG. The ability to efficiently and accurately route user prompts to the most suitable large

language model (LLM) is a crucial aspect of providing a high-quality and cost-effective service. The proposed Adaptive RAG (Retrieval-Augmented Generation) system aims to address this challenge by leveraging the principles of question complexity analysis and model routing (Hesih & Li, 2024) and Figure 29 illustrates the same. This method is particularly useful in applications where queries can vastly differ in nature and required expertise.

**Figure 29**

## *Adaptive RAG Approach*

(Hesih & Li, 2024)



Corrective Retrieval Augmented Generation (CRAG) aims to improve the robustness of generation by correcting the retrieval process. It employs a lightweight retrieval evaluator to assess the relevance of documents retrieved by the model. Depending on the relevance score, CRAG can trigger different actions (Correct, Incorrect, Ambiguous) to enhance the accuracy and relevance of the information used in the generation process (Yan & Gu, 2024). Refer Figure 30 for Corrective RAG’s algorithm/pseudocode.

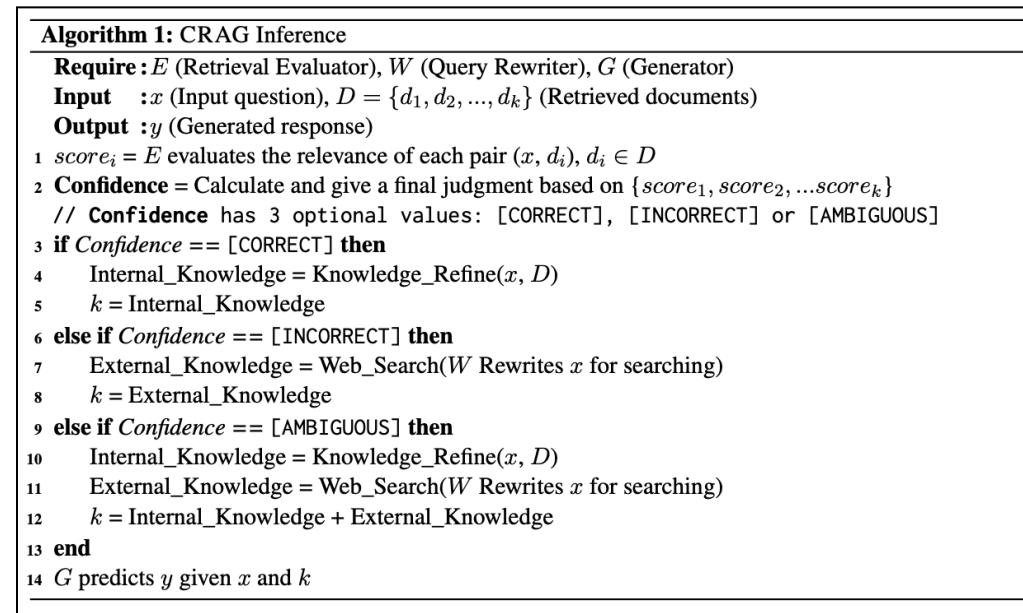
Self-RAG is a variant that introduces a self-reflection mechanism where the model decides whether to retrieve information based on the context of the query and its initial responses. It also includes a critical model that assesses the necessity and relevance of retrieved information, allowing for more selective and efficient use of external data (Asai & Wu, 2023).

Figure 5 illustrates the framework of a typical Self-Reflective RAG. The Large Language Model used in this configuration is LLaMa-3 which was released by Meta in April, 2024.

**Figure 30**

*Corrective RAG Algorithm*

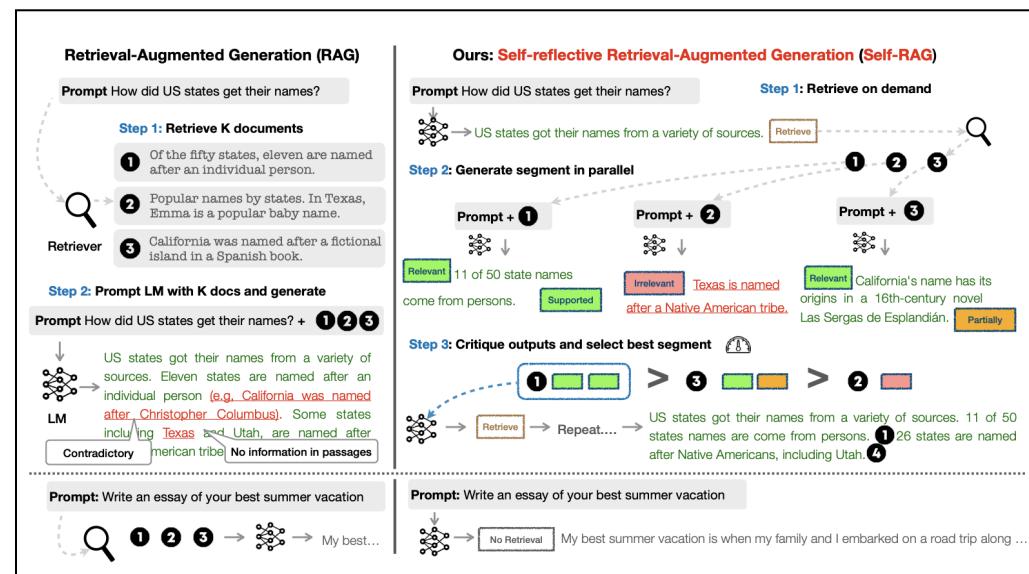
(Yan & Gu, 2024)



**Figure 31**

*Self-Reflective RAG Framework*

(Asai & Wu, 2023)



The primary objectives of this Combined approach RAG system for this project are:

1. Efficient Prompt Routing: Analyze user prompts and route them to the most appropriate retrieval approach (Adaptive RAG).
2. Improved Response/Retrieval Quality: Enhance the overall quality of responses by ensuring that prompts are routed to Web Search if there is no supporting context found (CRAG).
3. Handling of Hallucinations: Accommodate handling the problem of hallucinations in LLMs by identifying and promptly catching and correcting hallucinations to end with a correct response (Self-Reflective RAG).
4. Evaluation Cost Optimisation: There is zero cost for evaluation because of the use of Corrective and Self-RAG approaches.
5. Groundedness: Determines whether the generation is grounded in the document/vector store context and answers the user query. This is useful for evaluating then and there if the system is returning a grounded answer.

### ***Adaptive RAG and LLM Model Routing***

In the model configuration represented in Figure 29, Adaptive RAG with LLM model routing is proposed as a scalable and efficient approach for processing user queries.

The primary objectives of this architecture are,

- Query Classifier: The system is designed to classify user queries into different categories such as Factual, Analytical, Opinion, and Contextual. The Query Classifier ensures that each query is properly categorized to match the appropriate retrieval and response generation strategy. By determining the type of query, the system tailors its response and retrieval methods to the complexity and type of input.

- Adaptive Retrieval Strategies: Based on the type of query identified by the classifier, different Adaptive Retrieval Strategies are employed. For example, factual queries might retrieve specific documents, while opinion-based or analytical queries might trigger broader retrieval strategies like Knowledge Graphs. This ensures the retrieval strategy is well-suited to the context and type of query, enhancing retrieval accuracy.
- LLM Integration for Enhanced Retrieval: Once the relevant documents are retrieved, the system integrates the LLM to enhance both retrieval and ranking. The retrieved documents provide the context, and the LLM is used to rank and process this information. In particular, the Mistral-NeMo-12B-Base serves as the large language model (LLM), developed collaboratively by NVIDIA and Mistral AI. With 12 billion parameters, it is designed to handle diverse tasks and features a 128k context window for efficient processing of extensive text sequences. Using Hugging Face tools, the model was fine-tuned on our vector database, which is curated from high-quality scientific papers across domains like machine learning. Advanced techniques such as LoRA were employed to specialize the model, optimizing its capabilities in summarization, question-answering, and generating insights for scientific research.
- Model Routing and Fallback Mechanism: If the context retrieved by the system is deemed insufficient, or if the retrieved documents do not provide enough information, the system engages a Fallback Mechanism. This fallback mechanism routes the query through to perform additional retrieval (e.g., Wikipedia Search) to gather supplementary information. By leveraging the Model Routing process, the system ensures that even when the initial context is inadequate, it can generate a satisfactory response using secondary methods.

The inclusion of Wikipedia Search allows the system to gather additional information to strengthen the context for complex or vague queries.

- Response Quality and Binary Score Evaluation: Once the response is generated, it passes through the AutoRater & Evaluation Framework, which evaluates the quality and relevance of the response. The evaluation framework generates a Binary Score to assess the adequacy of the response, assigning values of 1, 0, or -1. A score of 1 indicates a satisfactory response, 0 reflects a neutral response with partial relevance, and -1 indicates an unsatisfactory or irrelevant response. This evaluation ensures that only high-quality responses are delivered to the user, and feedback from this process is used to adjust future retrieval and response strategies.

### ***Super RAG: Enhancing LLMs with Dynamic Knowledge Integration***

Super Retrieval-Augmented Generation (Super RAG) systems represent an advanced development in the field of artificial intelligence, particularly enhancing the capabilities of large language models (LLMs). These systems are designed to elevate the performance of LLMs by integrating external knowledge sources with minimal structural modifications (Thakur, A., & Gupta, R. 2024) and integrating sophisticated retrieval mechanisms to improve the accuracy, relevance, and depth of generated content.

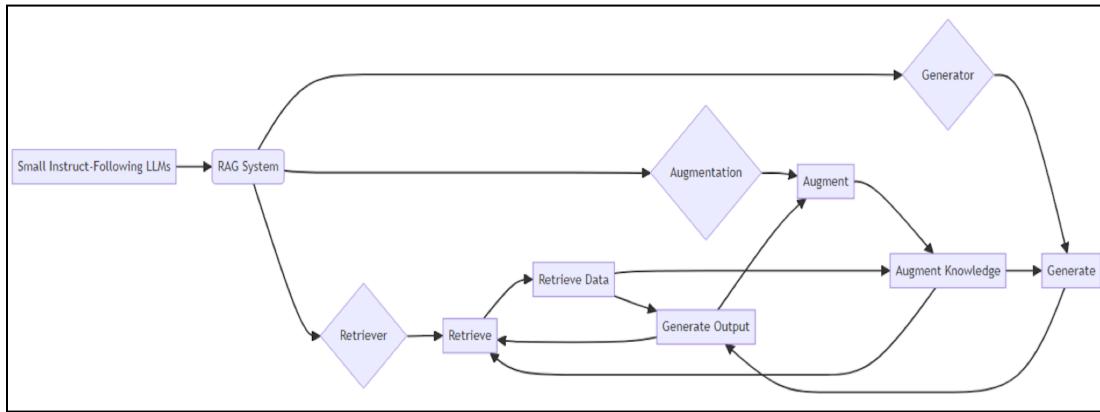
Super RAG's are designed to significantly enhance the capabilities of large language models (LLMs) by integrating real-time data retrieval from extensive external sources. This system dynamically fetches the most current and relevant data to provide LLMs with access to the latest information, which greatly improves the relevance and accuracy of generated responses. At its core, Super RAG features a sophisticated data retrieval engine that utilizes advanced search algorithms, an embedding and matching module to process the information, and

a dynamic routing mechanism that directs queries to the most appropriate LLM, optimizing both performance and resource usage. Additionally, the system includes a Quality Assurance LLM that continuously evaluates the quality of responses, adjusting retrieval parameters to optimize accuracy and relevance as shown in Figure 32.

**Figure 32**

*Super RAG Approach*

(Thakur, A., & Gupta, R. 2024)



The Super RAG Objectives are as follows,

1. Advanced Knowledge Integration: Implement a system that dynamically retrieves the latest and most relevant information from external databases and the internet, providing LLMs with access to current data beyond static datasets.
2. Real-Time Information Update: Ensure that the LLMs always have access to the most recent information, making them capable of handling queries that require up-to-date knowledge and facts.
3. High-Quality Response Generation: By accessing a broader range of information, Super RAG aims to significantly improve the relevance and accuracy of the responses generated by the LLMs.

4. Flexibility and Extensibility: Design the system to be easily adaptable to new data sources and capable of expanding without major overhauls to the existing infrastructure.

This continuous learning framework, enhanced by user feedback, ensures that Super RAG not only responds to queries with high precision but also adapts to new data sources and expands seamlessly, maintaining its effectiveness across various domains.

### ***LLaMA-7B with Ensemble Retrieval for Hybrid RAG Framework***

In the pursuit of strengthening the information retrieval and response generation capabilities for processing large-scale databases of scientific publications, we introduce a comprehensive model that includes a Hybrid Retrieval-Augmented Generation (RAG) framework with the LLaMA-7B language model. This methodology is built to extract, process, and synthesize information from enormous amounts of scientific text quickly. Below, we discuss the components, integration, and functionality of the Hybrid RAG-LLaMA model, along with a specific application on assessing scientific papers.

The Hybrid RAG-LLaMA framework has the following components: Documents are first loaded from a PDF format and processed using PyPDFLoader, which transforms PDF pages into textual content. This content is then separated into manageable chunks using a RecursiveCharacterTextSplitter, which assures substantial coverage and overlap, delivering contextually rich segments for later processing.

For text embedding, the high-performance model like ‘thenlper/gte-large’ from HuggingFaceEmbeddings module is used. This model represents a substantial development in natural language processing (NLP) capabilities, specifically geared for generating embeddings that are rich in semantic content. This model is part of a broader class of transformer-based models that have been pre-trained on a diverse corpus of text data, enabling them to perceive and

encode nuances in language efficiently. Embeddings generated by this model are used to turn textual content into numerical data that preserves semantic linkages, making it particularly effective for tasks such as document similarity comparison, information retrieval, and grouping.

The retrieval technique employs an EnsembleRetriever that combines semantic and keyword search engines. Semantic search is accomplished with a vector-based retriever (vectordb), whereas keyword searches employ the BM25Retriever. The retrieval system is substantially tailored to retrieve the most relevant document chunks, utilizing both maximum marginal relevance (MMR) and BM25 algorithms, enabling complete coverage of conceivable replies. The answer production is further streamlined by applying precise stopping criteria in the LLaMA pipeline, limiting repetitive and off-context creation.

### ***Multi-modal Retrieval-Augmented Generation (RAG)***

In this study, the Multi-Modal Retrieval-Augmented Generation (RAG) model leverages both textual and visual data for enhanced information retrieval and response generation. The model incorporates numerous components to successfully process and evaluate research papers, photographs, and other content, enabling a thorough comprehension of complicated issues.

The foundation of our Multi-Modal RAG is built upon the extraction of text and images from research publications. Using the PyMuPDF package, we systematically extract the textual material and graphics from the pdf papers. The photos are preserved and processed for further analysis, while the text data is formatted for simple retrieval. To enhance the extracted data, we apply the Llama 3 model to generate captions for the extracted photos. By delivering a cue that instructs the model to summarize the visual content, we gain short and informative descriptions that are kept alongside the written data. This strengthens the model's grasp of the context in which photographs occur inside the research report.

The textual and visual data are then embedded using the unsrstructured.io embedding model. This methodology turns the documents into numerical vectors, allowing similarity searches and enabling the model to compare the significance of different bits of content. The images that are extracted are then parsed through pytesseract and then fed into another Llama 3 model for image captioning as well as extracting the key values from the image. The embedding model can gather images and infer table structures, as well as chunk text based on section headings. To preserve legibility, each text chunk has a maximum of 4000 characters, with additional chunks formed after 3800. Cohesion is achieved by combining smaller text parts under 2000 characters. Extracted photos are saved in a specific output directory, facilitating content extraction for analysis.

The transformed documents and images are saved in a Chroma vector store, which enables for quick querying and retrieval of comparable information based on user queries. The vector store is set to support similarity searches, ensuring that the most relevant documents are retrieved swiftly.

The Multi-Modal RAG model processes user inquiries by engaging the retriever component, which retrieves relevant documents based on similarity to the input question. The collected content is then loaded into a Llama3 language model, which creates succinct, contextually suitable answers. The system is designed to handle multiple requests, producing results based on a combination of textual and visual data.

## 4.2 Model Supports

To support the development and execution of the proposed model combinations using Python, Jupyter Notebooks, and the Hugging Face ecosystem for large language models (LLMs) and embedding models, the tools and platforms outlined in Table 18 are recommended.

**Table 18***Model Supports**(Original from Team 8)*

<b>Model</b>	<b>Model Configuration</b>	<b>Model Supports</b>
M1	Adaptive RAG and LLM Model Routing	<ul style="list-style-type: none"> <li>• Python environment with access to previously setup vector store (FAISS), transformers, sentence-transformers, spacy, NER</li> <li>• Jupyter notebooks, Colab Environment for interactive development</li> <li>• Hugging Face API Support for different LLM configurations</li> <li>• GPU support for Multiple LLM Configurations</li> </ul>
M2	Super RAG: Enhancing LLMs with Dynamic Knowledge Integration	<ul style="list-style-type: none"> <li>• Python environment with packages like transformers</li> <li>• FAISS</li> <li>• Redis</li> <li>• REST API built with FastAPI, fastapi-cache</li> <li>• HuggingFace</li> <li>• Cohere</li> <li>• GPU support for Multiple LLM Configurations</li> </ul>
M4	MultiModal RAG	<ul style="list-style-type: none"> <li>• Python environment with access to previously setup vector store (ChromaDB), transformers, sentence-transformers</li> <li>• Jupyter notebooks, Colab Environment for interactive development</li> <li>• Hugging Face</li> <li>• Unstructured API</li> <li>• GPU support for LLama3 and pytesseract</li> </ul>
M3	LLaMA-7B with Ensemble Retrieval for Hybrid RAG Framework	<ul style="list-style-type: none"> <li>• Python environment with access to previously setup vector store (FAISS), transformers, sentence-transformers</li> <li>• Jupyter notebooks, Colab Environment for interactive development</li> <li>• Hugging Face</li> <li>• RAG (Retrieval Augmented Generation) framework for hybrid retrieval-generation models</li> <li>• GPU support for LLAMA and embeddings</li> </ul>
M5	RAG Agent with LlaMa-3: Combined Approach using Adaptive RAG, Corrective RAG & Self-RAG	<ul style="list-style-type: none"> <li>• Python libraries with access transformers and sentence-transformers</li> <li>• Uses ChromaDB for managing local vector embeddings</li> <li>• Tavily for web search</li> <li>• Requires Robust GPUs</li> <li>• Jupyter or Google Colab for development and testing</li> <li>• Ollama</li> <li>• GPU Support for LlaMa-3 Configuration</li> </ul>

**Figure 34**

*Model Supports Diagram*

(Original from Team 8)

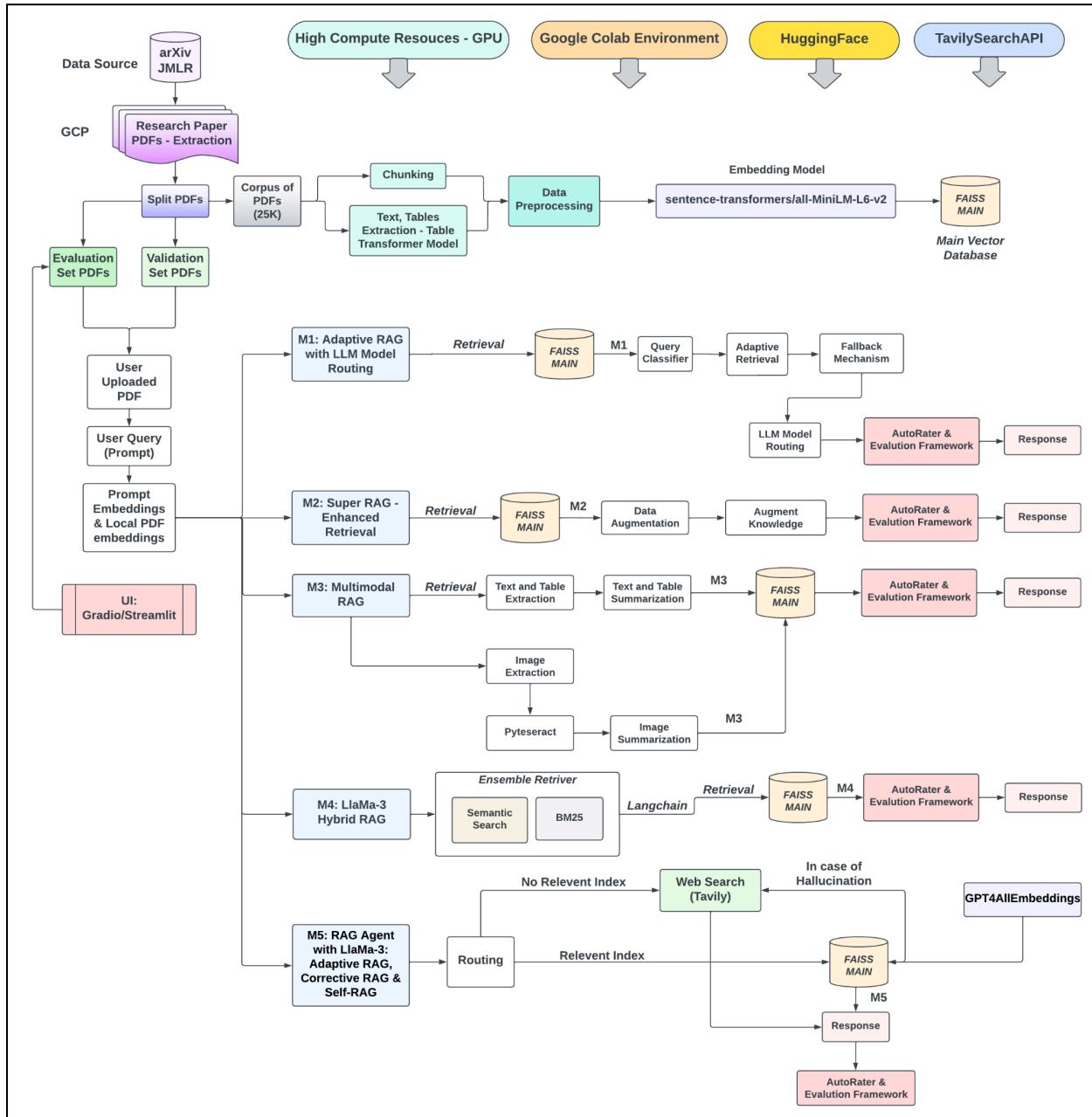


Figure 34 illustrates the general data flow and the model supports required. The system supports a high-performance computing environment for research paper retrieval. It leverages multiple models to process data from various sources including arXiv, JMLR, and user uploaded

PDFs. Data goes through preprocessing steps like splitting PDFs, extracting text and tables, chunking, and vectorization. The embedding model used is “all-MiniLM-L6-v2”. The system also supports various retrieval models including M1: Adaptive RAG with LLM Model Routing, M2: Super RAG-Enhanced Retrieval, and others. M3 leverages Cohere API and M5 leverages Tavily for web search when relevant contextual information isn't available in the respective vector database (ChromaDB).

Additionally, it uses ChromaDB and FAISS for vector storage retrieval, and finally a user interface (UI) built with Streamlit through which a PDF can be uploaded. User query (Prompts) can be passed to the system based on the PDF after which the application efficiently creates embeddings for both the uploaded PDF and the prompt. After this, response generation takes place through efficient retrieval of information with the aid of the RAG system.

#### **4.3 Model Comparison and Justification**

As delineated in Table 19, the seven model configurations each present unique strengths and considerations. M1 offers efficient retrieval and seamless integration capabilities but requires attention to computational demands. M2 excels in scalable data management and precision in document retrieval, ideal for large datasets. M3 focuses on robust data retrieval and caters to complex queries, though it poses integration complexities. M4 provides efficient search capabilities and flexible integration, with a need for careful component tuning. Lastly, M5 introduces potential for innovation with efficient embeddings, yet necessitates expertise in niche tools such as Gemma.

**Table 19**

*Model Comparison- Target Problems, Features and Approaches*

*(Original from Team 8)*

Model	Targeted Problems	Features and Approaches
Adaptive RAG and LLM Model Routing	<ul style="list-style-type: none"> <li>● Prompt Complexity Analysis: Accurately assessing the complexity and requirements of user prompts to determine the most suitable LLM for handling them.</li> <li>● Cost Optimization: Minimizing the overall cost of LLM usage by routing simpler prompts to less expensive models while ensuring complex prompts are handled by more capable (and potentially more expensive) LLMs.</li> <li>● Response Quality: Ensuring that prompts are matched with LLMs that can provide accurate, relevant, and high-quality responses.</li> <li>● Scalability: Enabling the system to accommodate the addition or removal of LLMs without significant architectural changes.</li> </ul>	<ul style="list-style-type: none"> <li>● Prompt Embedding: Convert prompts into embedding vectors using tokenizers and transformer models.</li> <li>● Classifier Model: Train a neural network classifier to predict the best LLM for a given prompt embedding, using a dataset of prompts and their evaluated best LLM indices.</li> <li>● LLM Routing: Integrate the classifier into the backend to route incoming prompts to the predicted best LLM.</li> <li>● Response Evaluation: Utilize a "Master LLM" or Autorater to evaluate the quality and relevance of LLM responses, generating binary labels for the best LLM.</li> <li>● Continuous Learning: Periodically retrain the classifier with new data to adapt to changes in LLMs or prompt types.</li> <li>● Cost Optimization: Route simpler prompts to less expensive LLMs, while handling complex prompts with more capable (and potentially costlier) LLMs.</li> <li>● Scalability: Design a modular architecture for easy integration or removal of LLMs.</li> <li>● Data Collection: Gather a large dataset of prompts and corresponding LLM responses.</li> <li>● Data Preprocessing: Convert prompts into embedding vectors for classifier input.</li> <li>● Model Training: Train the classifier using prompt embeddings as input and best LLM indices as targets.</li> </ul>
Super RAG: Enhancing LLMs with Dynamic Knowledge Integration	<ul style="list-style-type: none"> <li>● Static knowledge bases</li> <li>● Responses that lack depth</li> <li>● Timeliness of responses</li> <li>● Limited data access</li> <li>● Outdated information in responses</li> <li>● Lack of adaptability to new information</li> </ul>	<ul style="list-style-type: none"> <li>● Real-time data retrieval Integration with external databases</li> <li>● Dynamic knowledge updates</li> <li>● Access to a broader range of information</li> <li>● Enhanced data processing capabilities</li> <li>● Continuous learning and adaptation</li> </ul>

Model	Targeted Problems	Features and Approaches
LLaMA-7B with Ensemble Retrieval for Hybrid RAG Framework	<ul style="list-style-type: none"> <li>High demand for rapid data retrieval.</li> <li>Complex query processing.</li> <li>Scalability issues with large data volumes.</li> <li>Latency in real-time applications.</li> <li>Need for high precision in search results.</li> <li>Integration challenges with existing systems.</li> </ul>	<ul style="list-style-type: none"> <li>Utilizes all-MiniLM-L6-v2 for efficient text embedding.</li> <li>Employs FAISS for fast and precise vector retrieval.</li> <li>Scales effectively to handle large datasets.</li> <li>Reduces latency with optimized data retrieval pathways.</li> <li>Enhances accuracy with advanced embedding techniques.</li> </ul>
Multimodal RAG	<ul style="list-style-type: none"> <li>Inability to process non-text data (e.g., tables, images)</li> <li>Lack of contextual understanding in complex data</li> <li>Limited applicability in technical research domains</li> </ul>	<ul style="list-style-type: none"> <li>Multimodal RAG integrates text, tables, and images to enhance retrieval and comprehension.</li> <li>Cross-modal learning enables better interpretation of relationships between different data types.</li> <li>Supports diverse data sources, making it ideal for research papers with complex figures and tables.</li> </ul>
RAG Agent with LlaMa-3: Combined Approach using Adaptive RAG, Corrective RAG & Self-RAG	<ul style="list-style-type: none"> <li>Prompt Routing: Efficiently routing user prompts to the most suitable retrieval approach based on query complexity and required expertise.</li> <li>Response Quality: Enhancing the overall quality and relevance of responses by ensuring prompts are routed to web search when no supporting context is found in the vector store.</li> <li>Hallucination Handling: Accommodating the handling of hallucinations in large language models (LLMs) by identifying and promptly correcting them to provide accurate responses.</li> <li>Evaluation Cost Optimization: Reducing the cost of evaluating the system's performance by leveraging Corrective RAG and Self-Reflective RAG approaches.</li> <li>Groundedness: Ensuring that the generated responses are grounded in the context provided by the document/vector store and directly answer the user's query.</li> </ul>	<ul style="list-style-type: none"> <li>Adaptive RAG: Routes prompts to suitable LLMs based on query complexity.</li> <li>Corrective RAG (CRAG): Evaluates retrieved document relevance and triggers actions to improve accuracy.</li> <li>Self-Reflective RAG: Decides whether to retrieve information based on context and initial responses.</li> <li>LangGraph Framework: Designed for building language-based models with reasoning capabilities.</li> <li>LlaMa-3 Language Model: Powerful LLM released by Meta in April 2024.</li> <li>Chroma Vector Store: Efficient retrieval and storage of relevant information.</li> <li>Tavily Search API: Optimized web search for AI agents.</li> </ul>

As illustrated in the table below, Table 20 provides a comprehensive comparison of all our five model configurations, detailing the strengths, limitations, and justifications.

**Table 20**

*Model Comparison- Strengths, Limitations and Justifications* *(Original from Team 8)*

Model	Strengths	Limitations	Justifications
Adaptive RAG and LLM Model Routing	<ul style="list-style-type: none"> <li>• Improved Response Quality for technical/research queries.</li> <li>• Cost Optimization by routing simpler prompts to less expensive LLMs.</li> <li>• Scalability through modular architecture for easy LLM integration/removal.</li> <li>• Continuous Learning by retraining classifiers with new data to stay up-to-date.</li> <li>• Prompt Complexity Analysis for handling diverse research-oriented queries.</li> </ul>	<ul style="list-style-type: none"> <li>• Data Availability challenges in collecting diverse prompts and evaluated responses.</li> <li>• Domain Specificity requiring extensive fine-tuning for research/academic domains.</li> <li>• Interpretability, lack of transparency in decision-making process.</li> <li>• Bias and Fairness risks of inherent biases in data or models.</li> <li>• Maintenance and Updates ongoing efforts required for keeping the system up-to-date.</li> </ul>	Despite the potential disadvantages, Adaptive RAG and LLM Model Routing can be a valuable approach for a PDF chat app focused on research insights, as it addresses the inherent complexity and diversity of academic literature. By leveraging the strengths of multiple LLMs and routing prompts intelligently, the system can provide more accurate and cost-effective responses, while maintaining the flexibility to adapt to evolving research domains and advancements in language models.
Super RAG: Enhancing LLMs with Dynamic Knowledge Integration	<ul style="list-style-type: none"> <li>• Enhances LLMs by integrating real-time external data.</li> <li>• Improves the accuracy and relevance of generated responses.</li> <li>• Provides LLMs access to the latest information.</li> <li>• Adaptable to changes in information and data sources.</li> <li>• Reduces the need for frequent model retraining.</li> <li>• Supports a variety of language tasks and queries.</li> </ul>	<ul style="list-style-type: none"> <li>• Potentially high computational and bandwidth costs.</li> <li>• Complexity in integrating and maintaining external data sources.</li> <li>• Relies on the availability and quality of external data.</li> <li>• May introduce privacy or security concerns with external data.</li> <li>• Can be challenging to scale efficiently.</li> <li>• Requires continuous monitoring and updating.</li> </ul>	<ul style="list-style-type: none"> <li>• Offers a cutting-edge solution for making LLMs more dynamic and contextually aware.</li> <li>• Addresses the limitations of static datasets in traditional LLM applications.</li> <li>• Enhances user experience by providing more accurate and up-to-date responses.</li> <li>• Supports a wide range of applications such as research.</li> <li>• Facilitates continuous learning and adaptability of LLMs.</li> </ul>

Model	Strengths	Limitations	Justifications
LLaMA-7B with Ensemble Retrieval for Hybrid RAG Framework	<ul style="list-style-type: none"> <li>Efficient handling of large datasets with high-speed retrieval.</li> <li>Produces high-quality embeddings that improve search and response quality.</li> <li>FAISS enhances data retrieval speed and precision.</li> <li>Scalable architecture suitable for expanding data sizes.</li> <li>Low latency in response generation.</li> <li>Effective in managing and organizing large information repositories.</li> </ul>	<ul style="list-style-type: none"> <li>Setup and maintenance of FAISS may require specialized knowledge.</li> <li>Higher hardware requirements for optimal performance.</li> <li>Limited by the quality of embeddings in response accuracy.</li> <li>Potentially high initial setup cost.</li> <li>Requires continuous tuning to maintain performance.</li> <li>Dependency on specific hardware for maximum efficiency.</li> </ul>	<ul style="list-style-type: none"> <li>Ideal for applications requiring rapid access to large volumes of data.</li> <li>Enhances the capability of LLaMA-7B in generating contextually relevant responses.</li> <li>Allows for efficient and fast information retrieval, crucial for real-time applications.</li> <li>Ensures high-quality outputs with advanced embedding techniques.</li> <li>Supports a wide range of complex queries and data-intensive tasks.</li> </ul>
Multimodal RAG	<ul style="list-style-type: none"> <li>Enhanced information retrieval from multiple modalities (text, tables, images) for accurate, comprehensive results.</li> <li>Rich contextual understanding through cross-referencing different data types.</li> <li>Broader use cases, particularly in scientific and technical research.</li> </ul>	<ul style="list-style-type: none"> <li>Increased model complexity, requiring more computational resources.</li> <li>Slower processing times due to handling diverse data types.</li> <li>Dependent on the quality and structure of multimodal data.</li> </ul>	<ul style="list-style-type: none"> <li>Essential for comprehensive responses in research domains.</li> <li>Necessary for addressing complex, multimodal queries.</li> <li>Provides deeper insights by incorporating diverse data formats.</li> </ul>
RAG Agent with LlaMa-3: Combined Approach using Adaptive RAG, Corrective RAG & Self-RAG	<ul style="list-style-type: none"> <li>Efficient prompt routing and improved response quality.</li> <li>Hallucination handling and cost optimization through Corrective and Self-Reflective RAG.</li> <li>Groundedness evaluation for relevant and useful outputs.</li> </ul>	<ul style="list-style-type: none"> <li>Increased complexity due to multiple components integration.</li> <li>Reliance on retrieval quality from vector store and web search.</li> <li>Potentially high computational resource requirements.</li> <li>Interpretability challenges due to combined techniques.</li> </ul>	<ul style="list-style-type: none"> <li>Efficient query routing via Adaptive RAG for relevant, high-quality responses.</li> <li>Improved accuracy by integrating Corrective RAG and web search.</li> <li>Hallucination mitigation through Self-Reflective RAG for reliable outputs.</li> <li>Cost-effective evaluation without human intervention.</li> <li>Grounded and relevant responses through</li> </ul>

Model	Strengths	Limitations	Justifications
			<ul style="list-style-type: none"> <li>context evaluation.</li> <li>Leveraging advanced technologies like LangGraph, LLaMa-3, and Tavily Search API.</li> </ul>

#### 4.4 Model Evaluation Methods

Evaluating a RAG-LLM requires a multifaceted approach encompassing various methods. Firstly, automatic metrics assess core functionalities like retrieval accuracy and response fluency. The set of metrics known as the RAG-Triad Metrics are used for this. Answer Relevance, Context Relevance, and Groundedness are part of the RAG Triad. Other than this, Answer correctness is calculated to compare the ground truth and RAG answers.

Beyond assessing the quality of generated content, evaluating RAG models for real-world deployment requires examining their efficiency and scalability. Here, quantitative metrics like response time, throughput, and resource utilization become crucial. This section explores these metrics and methods for their calculation.

Secondly, human evaluation gauged through straightforward user testing provides insights into aspects like naturalness, coherence, and task completion success within the conversation flow. By combining these methods, we can comprehensively evaluate the RAG-LLM chat application and ensure it delivers an exceptional user experience.

##### ***Context Relevance***

This metric measures the relevance of the retrieved context, calculated based on both the question and contexts. The values fall within the range of (0, 1), with higher values indicating better relevance.

Ideally, the retrieved context should exclusively contain essential information to address the provided query. To compute this, we initially estimate the value of S by identifying sentences

within the retrieved context that are relevant for answering the given question. The final score is determined by the following formula (Equation 1):

$$ContextRelevance = \frac{S}{T} \quad (1)$$

Where ‘S’ represents the context relevancy and ‘T’ represents the total number of sentences in the retrieved context.

Example: A system is designed to answer questions about current events. The following example illustrates how context relevancy would be measured:

- Question: Who is the current president of the United States?
- Context 1: The United States is a country located in North America. Joe Biden is the current president of the United States.
- Context 2: The White House is the official residence of the President of the United States. It is located at 1600 Pennsylvania Avenue NW in Washington, D.C.

In this scenario, Context 1 is the most relevant because it directly answers the question. Context 2, while factually correct, doesn’t provide an answer to the question about the current president.

### ***Answer Relevance***

The evaluation metric, answer relevance, focuses on assessing how pertinent the generated answer is to the given prompt. A lower score is assigned to answers that are incomplete or contain redundant information, and higher scores indicate better relevance.

This metric is computed using the question, the context, and the answer. The answer relevance is defined as the mean cosine similarity of the original question to a number of artificial questions, which were generated (reverse engineered) based on the answer as seen in Equation 2.

$$AnswerRelevance = \frac{1}{N} \sum_{i=1}^N \cos(E_g^i, E_o) \quad (2)$$

Where  $E_g$  is the embedding of the generated question and  $E_o$  is the embedding of the original question, and  $N$  is the number of generated questions (typically set to 3 by default).

Scenario: Imagine a system designed to answer questions about astronomy. The following is an example of how answer relevance would be applied:

- Question: What is the largest planet in our solar system?
- Answer: Jupiter is the largest planet in our solar system.
- Generated Questions:
  - What is the second largest planet in our solar system?
  - How far is Jupiter from the sun?
  - Does Jupiter have rings?

In this scenario, the answer "Jupiter is the largest planet in our solar system" would be compared to the generated questions to determine its relevance to the original question. A high cosine similarity score between the answer and the original question would indicate good answer relevance.

### ***Groundedness***

The evaluation metric known as groundedness focuses on assessing whether a generated response can be traced back to specific evidence within the provided source materials or knowledge base. For this project, groundedness is calculated by determining the cosine similarity between the RAG answer and the retrieved context. This approach aims to identify responses that are factually supported and aligned with the provided source materials.

Groundedness is measured using a combination of automatic and human assessments:

- Automatic Assessment: The cosine similarity between the generated RAG answer and the retrieved context is calculated. Higher similarity scores suggest better groundedness, indicating the answer is well-supported by the source material.
- Human Evaluation: Human judges assess whether the generated response is supported by evidence within the source material and whether the connection between the response and the evidence is logically sound.

While there is no single, standardized formula for calculating groundedness, a conceptual representation would be  $\text{Groundedness} = (\text{Number of statements in the response supported by evidence}) / (\text{Total number of statements in the response})$ .

For this project, Groundedness is calculated as in Equation 3, plus a human evaluation for confirming the computed score is done:

$$\text{Groundedness} = \cos(\text{RAGAnswer}, \text{RetrievedContext}) \quad (3)$$

Scenario: Imagine a system is designed to answer questions about historical events. Let's see how groundedness would be evaluated:

- Question: Why did World War II begin?
- Source Material: A passage of text detailing the complex political and economic factors leading up to World War II.
- Answer: World War II started because of Germany's invasion of Poland in 1939. This act of aggression led to declarations of war from Britain and France.

Evaluation: The RAG answer would have a high cosine similarity with the retrieved context if the source material supports the statement about the invasion of Poland. This indicates good groundedness.

### ***Answer Correctness***

The answer correctness metric evaluates how accurate and factually correct a generated response is when compared to a ground truth reference. For this project, answer correctness is primarily determined by calculating the cosine similarity between the ground truth answer and the RAG answer. Human evaluation is also done to be doubly sure of the results as it is beneficial for nuanced questions or to identify subtle errors that similarity calculation may miss.

A Threshold Adjustment is required for acceptable cosine similarity to classify answers as correct or incorrect. This approach assesses the semantic overlap between the two answers, even if they are not expressed in the exact same words. Therefore, Answer correctness encompasses two critical aspects: semantic similarity between the generated answer and the ground truth, as well as factual similarity.

Scenario: Imagine a system is designed to answer questions about history. Let's see how answer correctness would be evaluated:

- Question: Who was the first president of the United States?
- Ground Truth Answer: George Washington
- RAG Answer: The first president of the US was George Washington. (*High similarity*)
- RAG Answer: Abraham Lincoln was a great American president. (*Low similarity*)

This section has explored various evaluation metrics for a RAG-LLM chat application. Metrics have been covered for core functionalities like retrieval accuracy (context relevance) and response fluency (answer correctness), along with groundedness, which assesses the factuality of the response based on the provided context.

By combining these metrics, we can gain a comprehensive understanding of the RAG-LLM's performance. However, evaluation is an ongoing process, and there's always room for improvement. Currently, these metrics can be calculated automatically with access to

OPENAI's API with frameworks such as RAGAS. But, the goal of this project is to provide a cost free solution which is why the team has set up a robust evaluation framework from scratch.

### ***Autoraters & Master LLMs***

To effectively evaluate the responses generated by the RAG-LLM system, we can leverage the capabilities of Autoraters and master LLMs. Autoraters are automated systems designed to assess the quality and relevance of responses based on predefined criteria. These systems can be trained on a large corpus of prompts and their corresponding high-quality responses, allowing them to learn the characteristics of good responses. By passing the RAG-LLM responses through an Autorater, we can obtain an objective evaluation of their quality, coherence, and relevance to the original prompt. Additionally, we can employ a master LLM, a highly capable language model trained on a vast amount of data, to provide a more nuanced and context-aware evaluation of the responses. The master LLM can be prompted to analyze the original prompt, and the RAG-LLM response, and provide a detailed assessment of the response's accuracy, completeness, and appropriateness. This approach combines the strengths of automated evaluation systems and advanced language models, enabling a comprehensive and reliable evaluation of the RAG-LLM system's performance.

### ***Response Time***

Response time measures the time between a user's query and the system's response delivery. For RAG models, this includes retrieval and generation steps. Capture the time from query initiation to response reception. Utilize logging on the client-side or dedicated monitoring tools. Minimizing response time is critical for user experience, especially in interactive applications like chatbots or real-time data processing. Faster responses enhance user satisfaction and system responsiveness. A short script is written to monitor response times and the team will

iteratively improve the response times based on observed values. Additionally, tools such as Prometheus, Jaeger, and OpenTelemetry are utilized.

### ***Throughput***

Throughput signifies the number of requests a system can process within a specific timeframe. For RAG models, it reflects the number of queries handled concurrently or sequentially over a period. It is calculated by dividing the total number of processed requests during a defined window (e.g., seconds) by the window duration. Throughput is often measured in requests per second (RPS). Apache JMeter, Locust, and Gatling are popular tools that are used for monitoring throughput. One of these tools is shortlisted for the project.

High throughput indicates a model's ability to scale and manage multiple simultaneous requests. This is vital for deployment in high-demand environments where the system needs to handle numerous queries efficiently.

### ***Resource Utilization***

Resource utilization measures how effectively the RAG system utilizes computational resources (CPU, memory) during operation. The team will monitor resource usage percentage over time while the system is active by leveraging tools like GPU stat (GPU), CPU monitors, and memory tracking utilities to collect these metrics.

Efficient resource utilization ensures the system neither overuses nor underuses available hardware. This impacts cost, efficiency, and scalability. Overutilization can lead to system performance degradation, while underutilization wastes resources.

### ***Implementing RAG Monitoring for Future Versions***

To effectively monitor and calculate these metrics, we propose the following practices:

- Monitoring Tool Integration: Incorporate tools like Prometheus, Grafana, or custom logging mechanisms for continuous performance metric tracking.
- Load Testing: Utilize tools like JMeter or Locust to simulate real-world load scenarios and understand system behavior under stress conditions.
- Resource Management: Implement resource management techniques like dynamic resource allocation, scaling, and optimization for improved resource utilization.

#### ***Additional Metric Considerations for Future Iterations***

Some additional metrics could be considered for future exploration when the UI is built and application is ready end-to-end:

- Human-Evaluator Engagement: This metric could track how long users interact with the chat application and their overall satisfaction with the conversation flow.
- Task Completion Rate: Measure the percentage of times the RAG-LLM successfully completes the task or answers the user's question comprehensively.
- Diversity and Creativity: While not always essential, evaluating the diversity and creativity of the generated responses can be valuable for specific applications.
- Explainability: Assessing the system's ability to explain its reasoning behind the answers can enhance user trust and understanding.

By incorporating a diverse set of metrics, the team can continuously refine the RAG-LLM chat application and ensure it delivers exceptional user experiences.

#### **4.5 Model Validation and Evaluation**

This section details and evaluates the performance and results seen for the models built. Twenty PDFs are randomly chosen for evaluation. A set of ten questions and their respective

ground truths are set up in a CSV file for each PDF. The resultant generated answer is saved and other metrics such as the RAG triad metrics are also computed.

### ***RAG Agent with LlaMa-3: Combined Approach using Adaptive RAG, Corrective RAG & Self-RAG***

The PDF/File chosen for evaluation is a research paper titled “*Response Prediction of Structural System Subject to Earthquake Motions using Artificial Neural Network*”.

A series of prompts and their answers are given below which showcases the robustness of the system. The evaluation of the answer will be done intrinsically by the system itself (AutoRater) as it checks and rechecks for hallucinations and routes to WebSearch if context is not available. Along with this, the answers will be human evaluated and verified with scores with grading criteria provided to the human evaluator.

1. Check whether the system is able to understand if some keywords are relevant to the document or not:

- *Prompt Template:* “You are a grader assessing the relevance of a retrieved document to a user question. If the document contains keywords related to the user question, grade it as relevant. It does not need to be a stringent test. The goal is to filter out erroneous retrievals. \n Give a binary score 'yes' or 'no' score to indicate whether the document is relevant to the question. \n Provide the binary score as a JSON with a single key 'score' and no preamble or explanation. Here is the retrieved document: \n\n {document} \n\n Here is the user question:  
{question} \n”
- *Question:* “machine learning”
- *Answer:* “{'score': 'yes'}”

- *Human Evaluation:* 10
2. Check if the system is able to understand a question and answer it appropriately in few sentences (to the point):
- *Prompt Template:* “You are an assistant for question-answering tasks. Use the following pieces of retrieved-context to answer the question. If you don't know the answer, just say that you don't know. Use three sentences maximum and keep the answer concise. Question: {question} Context: {context}”
  - *Question:* “How to best predict earthquakes according to the context?”
  - *Answer:* “According to the context, the best way to predict earthquakes is by using Artificial Neural Network (ANN) models that are trained on ground motion data. The ANN model can identify the responses of the structural system well and predict the maximum response of the building at different time periods.”
  - *Human Evaluation:* 10
3. Check if the system is able to grade an answer it generated.
- *Prompt Template:* “You are a grader assessing whether an answer is useful to resolve a question. Give a binary score 'yes' or 'no' to indicate whether the answer is useful to resolve a question. Provide the binary score as a JSON with a single key 'score' and no preamble or explanation. {generation} \n ----- \n Here is the question: {question}”
  - *Question:* “How to best predict earthquakes according to the context?”
  - *Generation:* “According to the context, the best way to predict earthquakes is by using Artificial Neural Network (ANN) models that are trained on ground motion

data. The ANN model can identify the responses of the structural system well and predict the maximum response of the building at different time periods.”

- *Answer:* {'score': 'yes'}
- *Human Evaluation:* 10

**Figure 35**

*Query Routed to Vector Store*

(Original from Team 8)

```
---ROUTE QUESTION---
How does the frequency of a building's natural frequency affect its response to an earthquake?
{'datasource': 'vectorstore'}
vectorstore
---ROUTE QUESTION TO RAG---
---RETRIEVE---
'Finished running: retrieve:'
---CHECK DOCUMENT RELEVANCE TO QUESTION---
---GRADE: DOCUMENT RELEVANT---
---GRADE: DOCUMENT RELEVANT---
---GRADE: DOCUMENT RELEVANT---
---GRADE: DOCUMENT RELEVANT---
---ASSESS GRADED DOCUMENTS---
---DECISION: GENERATE---
'Finished running: grade_documents:'
---GENERATE---
---CHECK HALLUCINATIONS---
---DECISION: GENERATION IS GROUNDED IN DOCUMENTS---
---GRADE GENERATION vs QUESTION---
---DECISION: GENERATION ADDRESSES QUESTION---
'Finished running: generate:'
("According to the provided context, a building's natural frequency can affect "
 "its response to an earthquake by causing the amplitude of sway to get larger "
 "and larger if the ground shakes with the same frequency as the building's "
 "natural frequency, resulting in resonance that can quickly cause the "
 "building to collapse.")
```

4. Check if the system can understand the routing mechanism and where it needs to refer to the context.

- *Prompt Template:* “You are an expert at routing a user question to a vector store or web search. Use the vectorstore for questions on an uploaded pdf (from which the vector store has been created). You do not need to be stringent with the keywords in the question related to these topics. Otherwise, use web-search. Give a binary choice 'web\_search' or 'vectorstore' based on the question. Return the JSON with a single key 'datasource' and no explanation. Question to route: {question}”
- *Question:* “llm agent memory”

- *Answer:* {'datasource': 'vectorstore'}
  - *Human Evaluation:* Excellent.
5. Check if the system can handle a query which is supposed to be routed to the vector store.
- Refer Figure 35.
- *Human Evaluation:* Excellent.
6. Check if the system can handle a query which is supposed to be routed to web search.

Refer Figure 36.

- *Human Evaluation:* Excellent.

**Figure 36**

*Query Routed to Web Search*

(Original from Team 8)

```
--ROUTE QUESTION--
How would you explain artificial neural networks to a child?
{'datasource': 'web_search'}
web_search
---ROUTE QUESTION TO WEB SEARCH---
---WEB SEARCH---
'Finished running: websearch:'
---GENERATE---
---CHECK HALLUCINATIONS---
---DECISION: GENERATION IS GROUNDED IN DOCUMENTS---
---GRADE GENERATION vs QUESTION---
---DECISION: GENERATION ADDRESSES QUESTION---
'Finished running: generate:'
('A neural network is like a special kind of computer program that helps solve '
 'problems, just like you do when you try to figure out an answer and check '
 'the back of the book if you get it wrong! It's made up of tiny "brain '
 'cells" called neurons that work together to help the computer learn and make '
 'decisions.'')
```

After this, the RAG triad metrics, Answer Correctness were calculated and the scores were assigned accordingly. The resource utilization percentages and response times were also captured for every response to further study it. Figure 37 illustrates a sample of the outputs for questions asked about a PDF.

Throughout evaluation, the model performed well and was able to answer questions both about the uploaded PDF and out of context well. In terms of response time, each response on average took 5.5 to 6 seconds and CPU-GPU usage was 3.3% and 15.4% respectively. The average human judge score was 8.8 which is an excellent grade when compared to industry standard.

**Figure 37***Sample Result Dataframe**(Original from Team 8)*

	question	ground_truth	rag_answer	response_times	cpu_start_usages	cpu_end_usages	average_cpu_usages	average_gpu_usages	context_relevance
0	What is the main contribution of the paper?	The paper introduces DyPlan, a dynamic strateg...	The main contribution of the paper on adapting...	9.35	0.6	9.5	2.50	16.14	8
1	What is Pluralistic AI alignment?	Pluralistic AI alignment refers to designing A...	Pluralistic AI alignment is concerned with int...	7.85	10.1	0.7	0.40	16.15	9
2	What challenges does the paper address in AI align...	The paper addresses the challenge of aligning ...	The paper addresses the challenges of AI align...	6.05	0.5	0.8	2.77	16.15	10
3	How does Multi-Objective Reinforcement Learnin...	MORL enables AI to learn multiple policies sim...	MORL helps in AI alignment by allowing for the...	7.83	0.7	7.1	1.65	16.15	9
4	What are the three phases of the adaptive align...	The three phases are learning, where the AI le...	The three phases of the adaptive alignment fra...	4.67	9.5	0.6	4.17	16.15	10
5	How does the framework interpret user reaction...	The framework uses an interpretation model tha...	The framework interprets user reactions as fee...	9.60	0.8	0.5	3.72	16.15	8

**Ensemble Retriever - LLaMA-13B with BAAI/bge-large-en-v1.5 Embedding and ChromaDB****Vector Store**

The model built using LLaMA in combination with ChromaDB can understand the context and answer the questions accordingly. Figure 10 shows the response from our model to the question “What are the keywords mentioned in this paper?”. The model can extract the relevant context from the knowledge base and answer the question with relevant information. All the sample questions and answers are coupled with the ground truth and context, which are shown in Figure 38. It is stored in a CSV file for further analysis.

**Figure 38***Sample Query and Response**(Original from Team 8)*

```
[ ] response_object = query_engine.query(questions[0])
display_response(response_object)

☒ Final Response: The keywords mentioned in the paper titled Response Prediction of Structural System Subject to Earthquake Motions using Artificial Neural Network are:
1. Earthquake
2. Neural Network
3. Frequency
4. Structure
5. Building

Please let me know if there is anything else I can assist you with.
```

**Figure 39**

*Details of the Questions with Responses and Groundtruth* (Original from Team 8)

	question	ground_truth	rag_answer	context	possible_questions
0	What are the keywords mentioned in the paper t...	The keywords written in the paper are: Earthqu...	The keywords mentioned in the paper titled "Re...	[Response Prediction of Structural System Subj...]	[What are the key keywords mentioned in the pa...
1	What is the purpose of using Artificial Neural...	Artificial Neural Networks (ANNs) are used to ...	\nThe purpose of using Artificial Neural Netwo...	[Response Prediction of Structural System Subj...]	[What is the main purpose of using Artificial ...]
2	How does the frequency of a building's natural...	A building's response to an earthquake is dyna...	The frequency of a building's natural frequenc...	[1 Introduction \n Real earthquake ground mo...]	[How does a building's natural frequency affec...
3	What kind of neural network model is most freq...	The most frequently applied neural network mod...	\nThe most frequently applied neural network m...	[1 Introduction \n Real earthquake ground mo...]	[What type of neural network model is most com...
4	How was the training of the ANN model conducte...	The ANN model was trained using real earthquak...	\nThe training of the ANN model for predicting...	[Response Prediction of Structural System Subj...]	[What data was used to train the ANN model for...
5	What is the significance of the study's findin...	The study's findings demonstrate the ability o...	The study's findings on predicting the safenes...	[Response Prediction of Structural System Subj...]	[What are the key implications of the study's ...]

The evaluation of the model involves the calculation of metrics such as answer relevance, context relevance, etc. The evaluation result in Figure 39 indicates that our RAG model provides the response to the question effectively based on the context and ensures that the answer is correct. This configuration leverages an ensemble RAG (retrieval-augmented generation) model with a hybrid search methodology that incorporates both dense and sparse retrieval methods, ensuring that responses are contextually appropriate and exactly matched with the content of the submitted document.

The results indicate the model's usefulness in data presentation, using strategies such as low-rank approximations (e.g., PowerSGD for gradient compression) to reduce computational overhead while keeping the accuracy of response generation. This hybrid search technique boosts the model's capacity to discover and classify important material effectively, making it

well-suited for resolving complicated, document-based inquiries inside academic and research environments.

During testing, the ensemble RAG model consistently demonstrated a high degree of accuracy, clarity, and contextual relevance in the generated responses. For instance, in response to the inquiry, "What is PowerSGD, and how does it achieve gradient compression?", the model offered an in-depth answer that addressed both technical specifics and the broader implications of employing PowerSGD for gradient compression. This example highlights the model's ability to handle complicated, technical questions well.

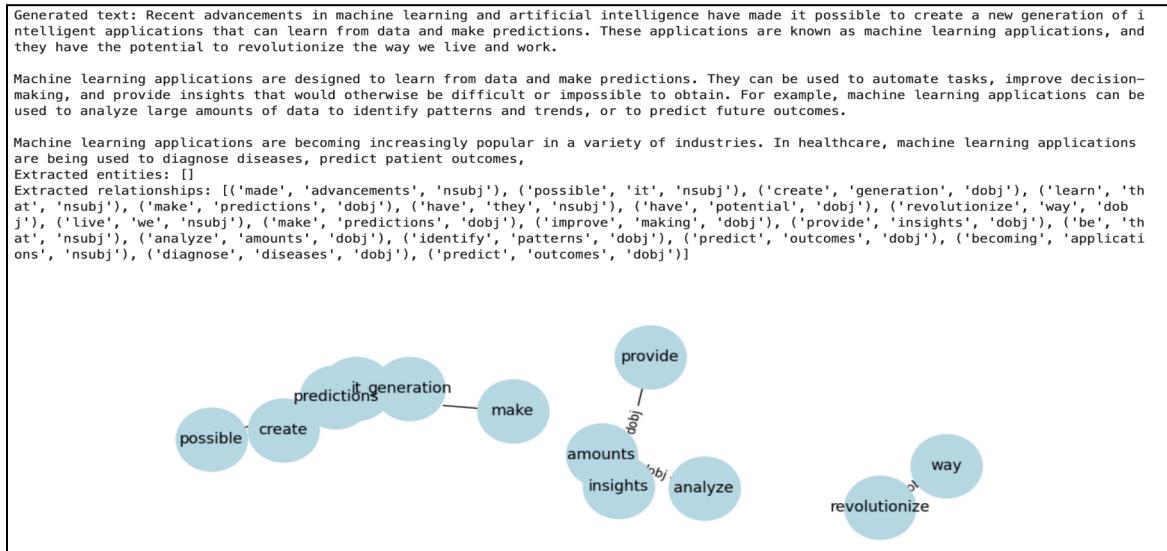
Overall, the results illustrate a robust hybrid retrieval model capable of addressing complicated, context-dependent questions with precision. The user-centered interface, paired with the high-performance backend, provides efficient interaction with academic texts, thus emphasizing the completeness, accuracy, and clarity of the solution.

### ***Adaptive RAG with Routing***

As shown in Figure 40, the generated text from the Main Vector DB using Mistral provides an overview of how advancements in machine learning are revolutionizing various industries, especially through applications like predictions and analysis. The extracted relationships visualize key verbs and their subjects, representing the underlying structure of machine learning processes. In Figure 41, the fallback mechanism is triggered when no valid documents are retrieved, prompting the system to route the query to Wikipedia. The retrieved Wikipedia content offers what Perplexity AI is, ensuring the query is still answered. Additionally, the binary score is set to 1, indicating that a valid response has been generated.

**Figure 40**

*Generated text with a knowledge graph from the Vector Store*      (Original from Team 8)



**Figure 41**

*Query Routed to Wikipedia Search with a Binary score for relevance*      (Original from Team 8)

No valid documents retrieved.  
Retrieved documents: []  
Routing to Wikipedia: Perplexity AI  
Final Response: Perplexity AI is an AI-powered research and conversational search engine that answers queries using natural language predictive text.  
Binary Score: 1

Table 20 typically compares the performance of various Retrieval-Augmented Generation (RAG) models when they respond to the same query. This comparison would show how different models retrieve and generate answers based on the retrieved information, highlighting the strengths and weaknesses of each model.

**Table 20**

*Comparison of RAG-Answers across different model*

(Original from Team 8)

Question	ground_truth	Model	Rag_Answer
What is PowerSGD, and how does it achieve gradient compression?	<p>PowerSGD is a gradient compression method that uses <b>low-rank matrix approximation</b> to reduce communication costs while preserving essential gradient information.</p>	Adaptive RAG and LLM Model Routing	<p>PowerSGD is a gradient compression technique for distributed training, designed to reduce communication overhead. It achieves this by approximating gradients with low-rank matrices and refining these approximations through power iterations, balancing between compression and maintaining accuracy in the training process.</p>
		Super RAG: Enhancing LLMs with Dynamic Knowledge Integration	<p>PowerSGD is a <b>gradient compression technique</b> designed to reduce the communication costs in distributed machine learning. It achieves this by approximating gradient matrices with <b>low-rank matrices</b>, significantly reducing the data that needs to be transferred between nodes during training. By focusing on low-rank approximations, PowerSGD can maintain model performance while <b>minimizing communication overhead</b>, making it especially useful for large-scale distributed training setups.</p>
		Ensemble Retriever : LLaMA-7B with Hybrid RAG Framework	<p>PowerSGD is a <b>gradient compression technique</b> used in distributed deep learning. It achieves gradient compression by representing the full gradient vector with the largest <b>low-rank matrices</b>, capturing the primary directions of the gradient while ignoring smaller components. This compression mechanism significantly <b>reduces communication time</b> and overall training time while maintaining model accuracy within 1% of uncompressed SGD.</p> <p>The key insight behind PowerSGD is that only a small number of low-rank matrices are sufficient to <b>capture the primary directions of the gradient</b>, while ignoring smaller components. By representing the full gradient vector as a low-rank matrix approximation, PowerSGD can significantly reduce the communication load and overall training time, while maintaining model accuracy.</p>
		RAG Agent with LlaMa-3: Combined Approach using Adaptive RAG, Corrective RAG & Self-RAG	<p>PowerSGD is a distributed optimization technique that computes a <b>low-rank approximation of the gradient</b> using generalized power iteration (subspace iteration). This approach avoids expensive Singular Value Decomposition and <b>achieves efficient compression</b>. The algorithm rapidly <b>compresses gradients</b>, aggregates compressed gradients using all-reduce, and maintains test performance comparable to SGD.</p>

## Multimodal RAG

Figure 42 depicts the contents that have been extracted from the provided PDF. Tables being the first content that has been extracted.

**Figure 42**

*Table Extraction*

(Original from Team 8)

The screenshot shows a user interface for extracting data from a PDF. At the top, a green bar displays the message "Content extraction completed! Displaying summaries...". Below this, the text "Original Table Content:" is followed by a large block of text detailing the performance metrics of various SGD variants (Algorithm, CIFAR-10, CIFAR-100, MNIST) across different levels (Level 1 to Level 50). The metrics include Mean, Std, and Percentage values. Below the main text, there are sections for "Table Summary" and "Variables", which list the metrics used. At the bottom, there is a section for "CIFAR-10 Data" with a numbered list of specific data points. On the right side of the interface, there are several small icons: a magnifying glass, a document, a plus sign, and a refresh symbol.

Content extraction completed! Displaying summaries...

Original Table Content:

Algorithm CIFAR-10 CIFAR-100 MNIST Mean Std Percentage Mean Std Percentage Mean Std Original SGD  
0.3451 ±0.2230 100% 0.5576 ±0.0796 100% 0.8685 ±0.1198 100% PowerSGD (Rank 1) 0.0040 ±0.0028 1.16%  
0.0069 ±0.0056 1.24% 0.0708 ±0.0736 8.15% Top-K SGD (Level 1)\* 0.0071 ±0.0051 2.06% 0.0069 ±0.0056  
1.24% 0.0035 ±0.0024 0.40% PowerSGD (Rank 2) 0.0137 ±0.0079 3.97% 0.0112 ±0.0093 2.01% 0.0946  
±0.0659 10.89% Top-K SGD (Level 2)\* 0.0069 ±0.0053 2.00% 0.0069 ±0.0049 1.24% 0.0072 ±0.0050 0.83%  
PowerSGD (Rank 4) 0.0361 ±0.0306 10.46% 0.0231 ±0.0182 4.14% 0.2361 ±0.1519 27.19% Top-K SGD (Level  
4)\* 0.0065 ±0.0051 1.88% 0.0085 ±0.0059 1.52% 0.0156 ±0.0078 1.80% PowerSGD (Rank 30) 0.1101 ±0.0758  
31.91% 0.0437 ±0.0306 7.84% 0.4909 ±0.2228 56.53% Top-K SGD (Level 30)\* 0.0178 ±0.0151 5.16% 0.0385  
±0.0306 6.91% 0.1706 ±0.1761 19.64% PowerSGD (Rank 50) 0.1339 ±0.0542 38.81% 0.0482 ±0.0327 8.65%  
0.6132 ±0.1155 70.63% Top-K SGD (Level 50)\* 0.0262 ±0.0169 7.59% 0.0758 ±0.0401 13.59% 0.1715 ±0.1183  
19.75%

Table Summary

Here are the values, trends, and data points I've captured from the table:

Variables:

- Algorithm (CIFAR-10, CIFAR-100, MNIST)
- Mean (a metric to measure the performance of the algorithm)
- Std (a metric to measure the standard deviation of the performance)
- Percentage (a metric to measure the percentage of improvement)
- Rank (the ranking of the algorithm)

CIFAR-10 Data:

- Original SGD: Mean = 0.3451, Std = 0.2230, Percentage = 100%
- PowerSGD (Rank 1): Mean = 0.0040, Std = 0.0028, Percentage = 1.16%
- Top-K SGD (Level 1)\*: Mean = 0.0071, Std = 0.0051, Percentage = 2.06%
- PowerSGD (Rank 2): Mean = 0.0137, Std = 0.0079, Percentage = 3.97%
- Top-K SGD (Level 2)\*: Mean = 0.0069, Std = 0.0053, Percentage = 2.00%

Similarly, Figure 43 shows the image that has been extracted fed to the model for summarization. The model then identifies the trends and unique values from the supplied image. This is later stored in the vector database (Chroma in this case) along with the texts and tables.

**Figure 43**

*Image Extraction*

(Original from Team 8)

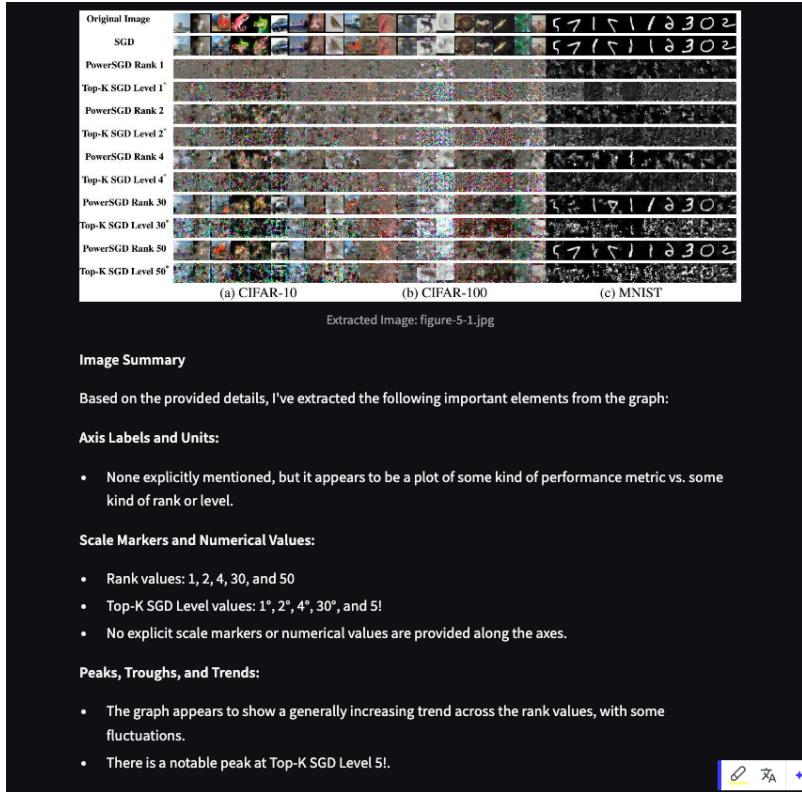


Figure 44 and Figure 45 shows for the question asked, the model is able to answer with not only texts that are extracted but is also able to provide the relevant tables and diagrams accordingly.

**Figure 44**

*Q&A example*

*(Original from Team 8)*

Enter your question:

What is PowerSGD, and how does it achieve gradient compression?

**Answer:**

```

▼ { ⓘ
  "content" :
  "Based on the provided text, I can help you extract the values, trends, and
  data points related to PowerSGD and its application in distributed learning.

  **Values:**

  * Rank values: 1, 2, 4, 30, 50
  * Level values: 1, 2, 4, 30, 50
  * Compression ratios: varying for different ranks and levels

  **Trends:**

  * PowerSGD is a gradient compression algorithm that achieves compression by
  retaining only the top-k elements of the gradient vector.
  * The algorithm uses a combination of rank and level to determine the
  compression ratio, with higher ranks and levels resulting in more aggressive
  compression.
  * PowerSGD has been shown to be effective in reducing the communication
  overhead of distributed learning while preserving the model's accuracy.

  **Data Points:**

  * Figure 1: Reconstructed images from GradInv across datasets
  * Table 1: SSIM values for reconstructed images, including mean, standard
  deviation, and percentage of SSIM relative to the baseline obtained with
  original SGD
  * Results highlight the effect of each compression method on gradient inversion
  and provide insight into the privacy-preserving capability of each algorithm.

```

**Figure 45**

*Q&A on image*

*(Original from Team 8)*

## Ask a Question

Enter your question:

At which ranks are data points shown for the PowerSGD and Top-K SGD algorithms in the graph?

**Answer:**

```

▼ { ⓘ
  "content" :
  "Based on the text, the data points for PowerSGD and Top-K SGD are shown at the
  following ranks:

  * PowerSGD: Rank 1, Rank 2, Rank 4, Rank 30, Rank 50
  * Top-K SGD: Level 1, Level 2, Level 4, Level 30, Level 50"

```

## **5. Data Analytics System**

### **5.1 System Requirements Analysis**

#### ***System Boundary, Actors and Use Cases***

Figure 46 outlines the architectural components and functionalities of the ML Chat App Subsystem designed to assist ML researchers and students. The system is enclosed within a defined boundary, indicating its scope and interactions with external entities. At the core of the subsystem lies the User Interface, which serves as the primary interaction point for users. Through this interface, users can upload PDF documents and select appropriate model configurations. Once a user initiates a query, the system's application component processes the input and forwards it to the AI Modules for further analysis.

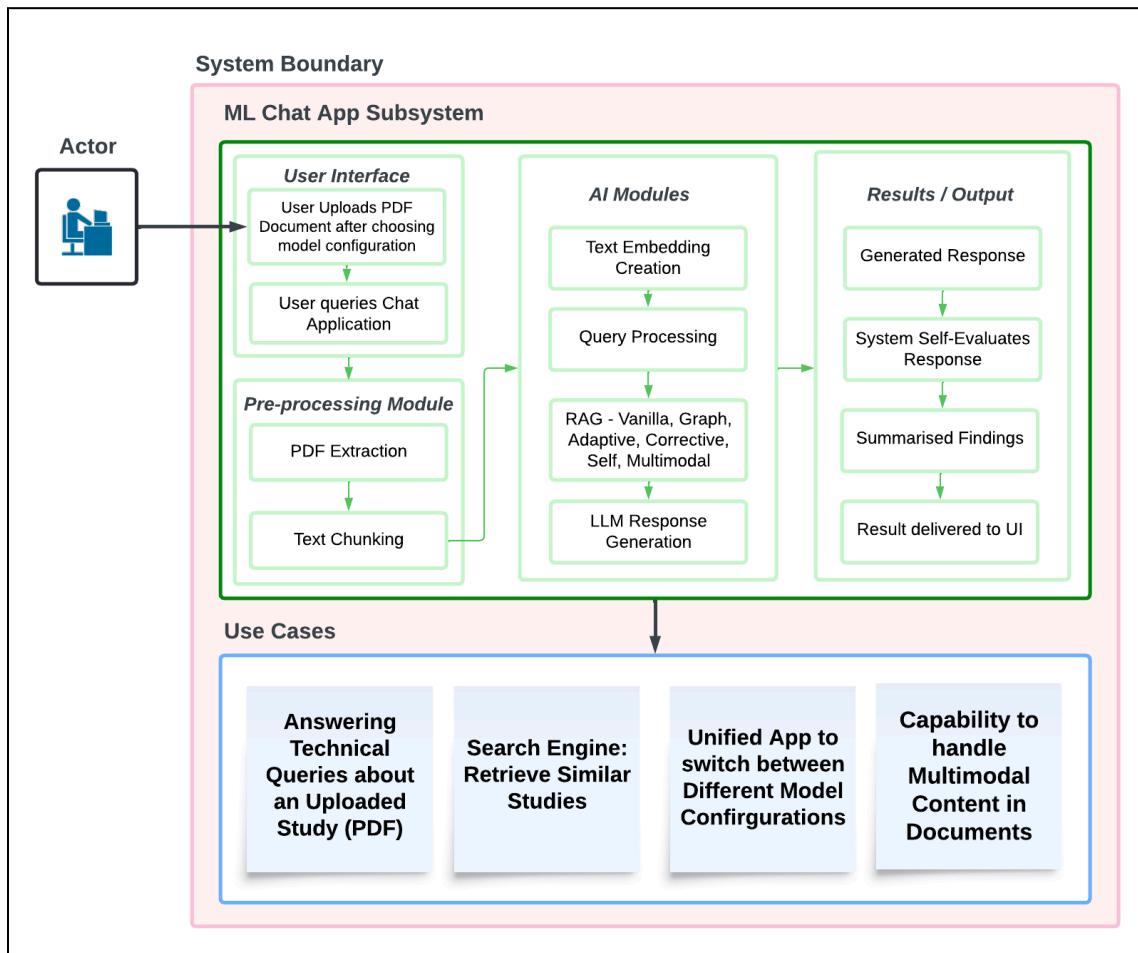
The AI Modules encompass various components responsible for handling different aspects of the query processing. Text Embedding Creation converts the input text from the uploaded document into a numerical representation, facilitating machine understanding. Query Processing involves understanding the user's intent and formulating an effective response. The RAG (Retrieval Augmented Generation) component plays a crucial role in retrieving relevant information from a corpus of over 25,000 research papers. Each model configuration follows a different approach to Retrieval Augmented Generation.

This retrieved information is then used to generate a comprehensive and informative response in the Visualization Module. To ensure the quality and accuracy of the generated responses, the system incorporates a self-evaluation mechanism. This component assesses the response and provides feedback to improve future outputs. Finally, the Results/Output component delivers the generated response to the user interface, where it is presented in a clear and understandable format.

**Figure 46**

*System Boundary, ML Chat App Subsystem, and Use Cases*

*(Original from Team 8)*



The primary Use Cases as illustrated in Figure 20 are:

1. Answering Technical Queries: The system can provide informative and accurate responses to technical questions related to ML research. Users can ask questions about specific algorithms, techniques, or concepts, and the system will leverage its knowledge base to offer helpful explanations.
2. Retrieving Similar Studies: Researchers can use the system to discover relevant studies from a vast corpus of over 25,000 papers. By entering keywords or phrases, users can

quickly identify similar research that can inform their work. Therefore, the application also serves as a search engine for referring to similar studies published in the past.

3. Extracting Key Insights from Uploaded PDFs: The system can extract essential information and key points from uploaded PDF documents. This can be particularly useful for summarizing lengthy papers or identifying relevant findings.
4. Simplifying Literature Reviews for ML Research: The system can assist in the process of conducting literature reviews by summarizing and organizing relevant research. This can save researchers time and effort, allowing them to focus on their own research.

### ***Data Analytics and Machine Learning Capabilities***

The system offers a robust suite of analytics and machine learning capabilities designed to process, understand, and generate insightful responses from extensive research literature, particularly in the field of machine learning and AI. It provides a valuable tool for researchers and professionals seeking to:

1. Efficiently process and analyze research literature: The system can extract text from PDF documents, break it into manageable chunks, and convert it into embeddings for contextual analysis. This enables efficient retrieval and similarity analysis, crucial for generating context-aware responses to user queries.
2. Retrieve relevant information and generate comprehensive responses: Leveraging the power of RAG models and LLMs, the system can retrieve pertinent information from a vast corpus of research papers and augment responses with contextually appropriate data. This ensures that users receive accurate and informative answers, whether sourced internally or through web searches.

3. Adaptively refine response accuracy: Adaptive and corrective mechanisms allow the system to refine response accuracy by routing queries to appropriate retrieval approaches and handling cases where context is insufficient. Additionally, a self-correction mechanism helps mitigate potential hallucinations from the LLM's responses.
4. Discover relevant research and extract key insights: The system can efficiently retrieve similar studies and extract key insights from a large database of scientific literature. This aids in tasks like literature reviews and discovery of related studies, streamlining the research process.
5. Receive informative and well-structured responses: Generated responses are evaluated for relevance and coherence, ensuring they meet the user's informational needs. The results are delivered in a user-friendly format, with the system self-evaluating response quality and summarizing key findings.
6. Engage in interactive and informative conversations: The system provides an adaptive conversational interface that supports dynamic interactions, adjusting responses based on the context and substance of user questions. This enables an intuitive and educational dialogue, aiding in understanding complex machine learning literature and supporting academic research and professional information needs.
7. Handle a variety of content formats: Beyond text, the system can process non-textual elements like tables from PDFs, ensuring comprehensive content parsing and analysis. This allows the system to address questions that may require interpretation of tabular data, a significant aspect of scientific literature.
8. Continuously improve performance: The system incorporates continuous learning mechanisms to refine performance based on real-time user interactions and feedback.

This ensures the system remains relevant and accurate, adapting to evolving user demands.

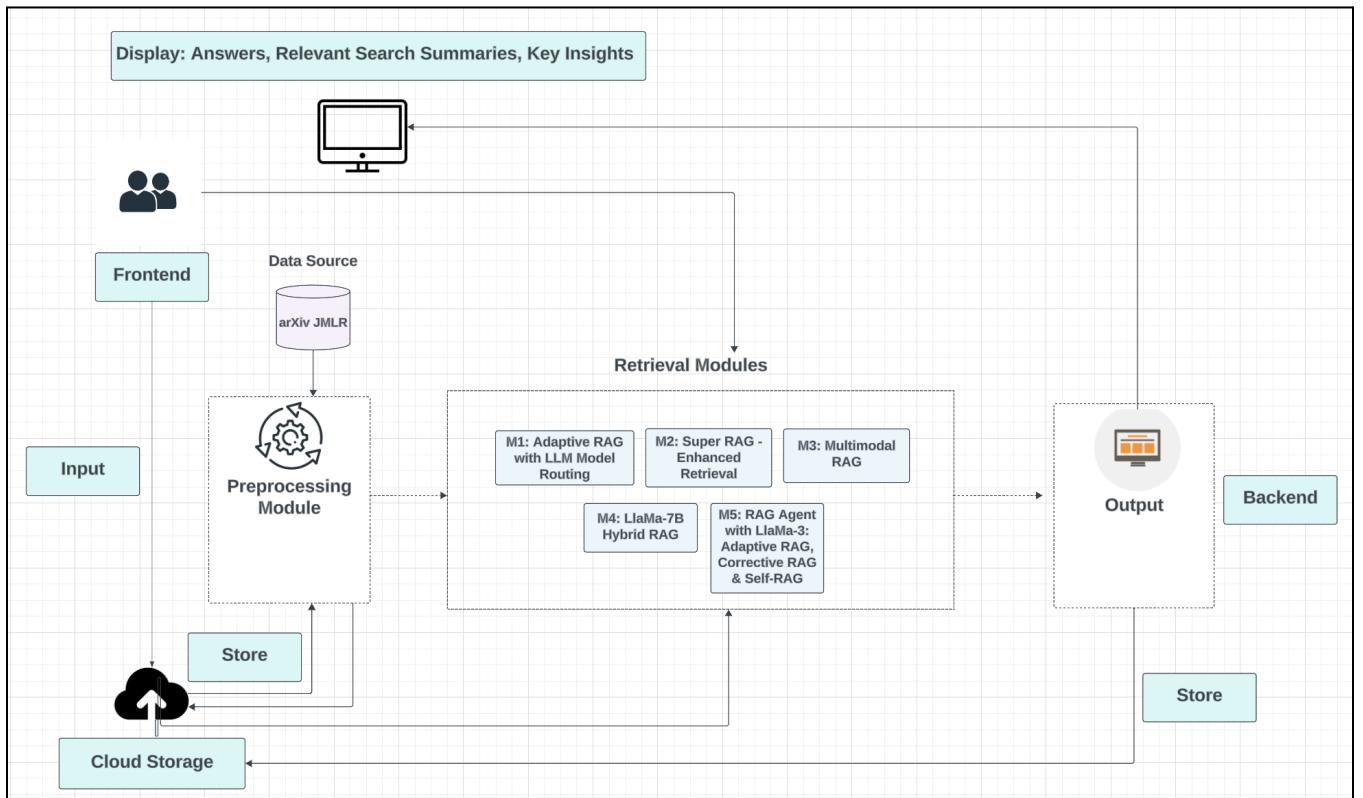
## 5.2 System Design

### *System Architecture and Infrastructure Design*

**Figure 47**

*System Architecture*

(Original from Team 8)



The system architecture as shown in Figure 47 is designed to ensure efficient processing and retrieval of information from a vast corpus of research papers, leveraging the various Retrieval-Augmented Generation (RAG) models. The architecture follows a modular design that integrates several high-performance components.

At its foundation, the system begins by processing research papers sourced from arXiv and JMLR. These papers are broken into smaller chunks for more manageable processing by

splitting and chunking these papers to facilitate further processing. Then the data undergoes preprocessing that involves extracting text and tables. The embedding model then transforms this data into vector embeddings, which are utilized for retrieval purposes. These embeddings are stored in a FAISS (Facebook AI Similarity Search) Vector Database, acting as the primary retrieval system.

The architecture integrates multiple RAG models: Adaptive RAG, Super RAG, Multimodal RAG, and LLaMA-7B Hybrid RAG. Each model is responsible for different types of retrieval and routing based on the user's query or PDF uploads. These models interact with the FAISS vector database for retrieval and then send the retrieved information through an evaluation framework that provides responses to the user.

Also, a routing mechanism has been integrated to intelligently select the appropriate model depending on the type of query. This mechanism ensures that the most suitable model handles the user request, optimizing retrieval performance. The response generation component is driven by a large language model (LLM) that creates high-quality summaries or customized insights, tailored specifically to the user's input.

When the internal FAISS vector database doesn't provide sufficient context, the system employs a series of adaptive strategies to ensure accurate results. A Query Classifier first determines the type of query, classifying it as Factual, Analytical, Opinion, or Contextual. Based on the query type, the system applies Adaptive Retrieval Strategies tailored to the specific nature of the question. Additionally, LLM Integration is used to enhance the retrieval and ranking process, ensuring the most relevant documents are surfaced. If these documents still lack enough context, the Model Routing mechanism engages a fallback process to generate a response, ensuring the system provides accurate and relevant information, even in complex scenarios.

The system integrates external web searches, such as Tavily and Wikipedia, to broaden the scope of retrieval. Tavily acts as a fallback mechanism, ensuring that accurate web-based content is retrieved, while Wikipedia supplements the system with general knowledge. Together, these external sources enhance the system's ability to provide comprehensive and accurate answers, extending beyond the primary database.

Lastly, a local RAG Agent utilizing LLaMa-3 has been implemented to further enhance retrieval accuracy. This agent combines Adaptive RAG, Self RAG and Corrective RAG, which reduces errors such as hallucinations and ensures that the retrieved information aligns with the user's query.

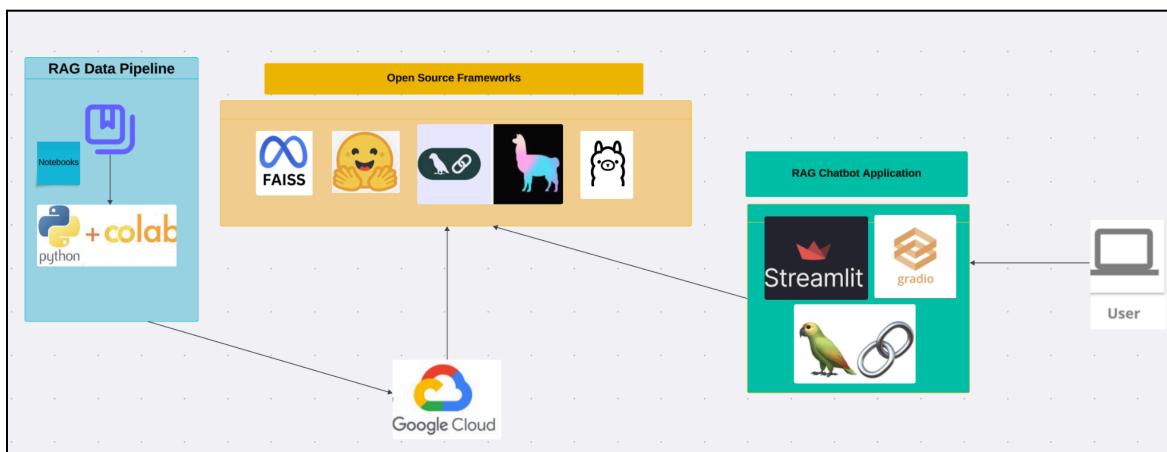
The system also features a search engine capability, allowing users to search directly within the research repository, identifying relevant papers based on content or abstract similarity. By integrating specialized retrieval models and external web searches into a cohesive architecture, the system offers a robust and versatile solution for users, ensuring that queries are thoroughly addressed.

### ***Integration with Supporting Platforms, Frameworks, and Cloud Environment***

**Figure 48**

*Supporting platforms, frameworks and a cloud environment*

*(Original from Team 8)*



The system integrates with various platforms and frameworks as shown in Figure 48 to ensure efficient processing, embedding, retrieval, and overall system performance. These integrations leverage state-of-the-art tools for managing and processing vast volumes of data while ensuring scalability, flexibility, and high computational performance.

1. Google Cloud Platform: Google Cloud Platform (GCP) stores and processes research papers from sources like arXiv and JMLR, providing a scalable infrastructure for handling large data volumes. The system leverages GCP's storage and compute services, including Google Colab, to extract and split papers into manageable chunks, optimizing retrieval efficiency. This ensures the system scales seamlessly with an increasing number of papers while maintaining high processing speed and quick data access.
2. Hugging Face: Hugging Face is integrated for its powerful pre-trained models and to embed the data. For security and controlled access, Hugging Face tokens are utilized to authenticate API requests and ensure secure communication with the Hugging Face platform. These embeddings convert text into vector representations, enabling efficient similarity searches and retrievals. Hugging Face's models ensure high-quality and accurate embeddings, helping the system enhance the retrieval of relevant information.
3. FAISS (Facebook AI Similarity Search): For high-performance similarity search and retrieval, the system relies on FAISS, a powerful library developed by Facebook AI. FAISS acts as the core retrieval database, storing the vector embeddings generated by Hugging Face models. It enables the system to perform approximate nearest neighbor searches across massive datasets, ensuring quick and accurate retrieval of relevant documents based on the user's input. FAISS is designed to handle very large datasets efficiently, making it ideal for this system, which processes and stores embeddings from

over 25,000 research papers. The system utilizes FAISS to ensure that relevant information is surfaced in a timely manner, even in the context of large-scale data.

4. Web Search Integration (Wikipedia, Cohere, and Tavily): In cases where the internal FAISS database doesn't provide sufficient context for a query, the system integrates external web search APIs such as Wikipedia and Tavily. When the system determines that the retrieved information is insufficient to satisfy the query, it will initiate additional retrieval processes or fallback mechanisms to ensure a more comprehensive response. By using APIs, the system can dynamically reach out to these web sources, retrieve relevant content, and integrate it with the system's internal retrieval, ensuring that the user gets the most accurate and complete response possible.
5. User interface (UI): The user interface (UI) is constructed using Streamlit and Gradio, two famous open-source frameworks geared for generating interactive applications. Streamlit enables the system to give a user-friendly front-end where users may input queries or upload PDFs. It provides an intuitive, engaging experience, allowing users to engage with the system easily. After analyzing the input, the applicable responses are displayed back using the same interface. Streamlit's versatility enables real-time engagement, where users may upload documents, ask inquiries, and receive results rapidly, thus boosting the overall usefulness of the system. On the other side, Gradio is also deployed to enhance the system's UI, delivering easy-to-use components for constructing chat interfaces. Gradio allows users to interact with the model in a basic manner, either by uploading files, making text queries, or providing other inputs. It interfaces smoothly with multiple machine learning frameworks, enabling speedy deployment and effective testing of models. With Gradio's intuitive design, the system

gives users with an accessible interface to connect with the backend model, boosting the overall user experience by enabling different ways to access and receive responses. Together, Streamlit and Gradio create a strong, adaptable, and user-friendly front-end, boosting the entire functionality and interactivity of the system.

6. LangChain and Llama Index: Llama Index is used to efficiently index and organize the data, enabling faster and more accurate retrieval of relevant documents. LangChain manages the interaction between the user's queries and various retrieval models, ensuring smooth multi-step operations such as document retrieval, summarization, and analysis to provide comprehensive responses.

### ***Data Management Solution and Database Design***

The system employs a FAISS-based Main Vector Database as its core data management solution, optimized for efficient similarity searches using vector embeddings of text data extracted from research papers. FAISS is crucial for its high-performance, approximate nearest-neighbor search capabilities, which ensure quick and relevant document retrieval. The system handles several critical aspects of data management as mentioned below:

1. Corpus Creation: The system processes a large corpus of around 25,000 research papers by chunking them into smaller sections, making them more manageable for processing and retrieval. These chunks are embedded using sentence-transformers, which create vector representations of the text. These vectors are stored in FAISS and Chroma DB, enabling efficient similarity-based retrieval when a user submits a query.
2. Vector Database Structure: A main vector database is created using FAISS, which is designed to support multiple RAG models. It employs approximate nearest neighbor (ANN) algorithms to quickly match user queries to the most relevant document chunks, thus minimizing computational costs and improving retrieval times. This allows the

system to access the relevant chunks of research papers with high precision, ensuring optimal results for both simple and complex queries.

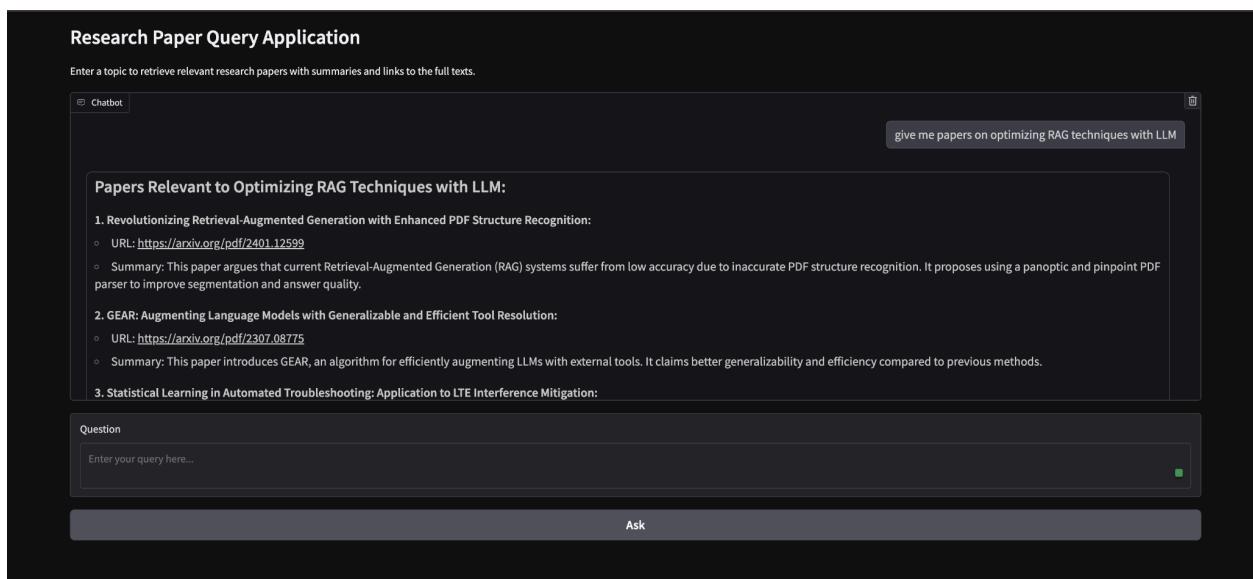
3. Data Embedding and Retrieval: The system uses MiniLM and other embedding models to transform textual data into vector representations. These embeddings are stored in FAISS, allowing for fast retrieval based on vector similarities. Each RAG model accesses the database to pull relevant data for user prompts or queries. For further scalability and enhanced retrieval efficiency, Chroma DB is integrated as a supplementary vector store, ensuring additional support for organizing and managing embeddings when required.
4. Scalability: The FAISS-based architecture is designed to handle large datasets, ensuring that the system can easily scale as the number of research papers and their embeddings grow. As the corpus expands, the system's modular approach allows for the seamless addition of new embeddings and research papers without compromising performance.

### ***User Interface and Data Visualization Design***

**Figure 49**

*Example of the interface for the application*

*(Original from Team 8)*



The user interface (UI) as shown in Figure 49 is designed with simplicity and functionality, using Gradio as the framework for front-end interaction. It allows users to upload research papers in PDF format, submit queries, and use an integrated search engine to find relevant research topics or papers. By entering keywords or questions, the system retrieves the most relevant results.

The UI supports real-time interaction, offering immediate responses on queries. The system processes user input quickly, retrieves pertinent documents, and presents results in a user-friendly format. Regardless of their technical background, users can navigate the system with ease, whether conducting literature reviews, extracting data, or addressing technical machine learning queries.

This seamless interaction, paired with real-time feedback and advanced retrieval methods, provides an efficient solution. The UI ensures that researchers, students, and professionals alike have access to powerful tools for exploring, analyzing, and extracting insights from vast amounts of data. In addition to this, the interface is built with scalability, allowing future enhancements. This ensures the system can adapt to growing user needs.

### **5.3 Intelligent Solution**

#### ***Development of AI and Machine Learning Models***

This project aims to utilize modern AI methodologies for comprehensive PDF analysis, allowing users to rapidly extract significant insights from enormous machine learning literature. We established four fundamental models, each tailored to tackle a distinct facet of document retrieval, multimodal analysis, and query response production. The models are enumerated as follows:

- **Adaptive Retrieval-Augmented Generation with Large Language Model :** This technique employs an adaptive retrieval strategy to dynamically identify the most pertinent research papers in response to user queries. Upon retrieval of the documents, a large language model (LLM) is utilized to provide a thorough response, guaranteeing that the information provided is both precise and contextually pertinent. This model is optimal for real-time dynamic retrieval where the most pertinent documents need to be picked and summarized in response to user input.
- **Super RAG-Enhanced Retrieval :** This methodology employs local embeddings of the PDF documents stored in the system to improve retrieval precision. The model processes user queries by retrieving documents and refining them using an LLM, thereby ensuring that only the most relevant results are presented. Best suited for cases where high-precision retrieval is required, especially in research environments that demand reliable and narrowed-down findings.
- **Multimodal RAG :** This paradigm stretches beyond text-based information by including multimodal data such as tables, figures, and other visual features contained in PDFs to create a more holistic retrieval process. By incorporating these non-textual aspects, the model provides richer, more informative replies. Ideal for research papers that include key information in tables or figures, boosting the range of the insights gleaned from a text.
- **RAG Agent with LLaMA-3 :** This sophisticated RAG agent that incorporates three techniques:
  - **Adaptive RAG** dynamically modifies the retrieval according to the query's context.

- **Corrective RAG** iteratively refines the replies by incorporating additional relevant data
- **Self-RAG** learns from prior interactions, improving reaction accuracy over time.

These strategies combine to provide the RAG Agent with enhanced skills in delivering accurate and informative solutions for research topics.

- **Hybrid RAG** : The Hybrid RAG model combines classic semantic search strategies, such as BM25, with retrieval-augmented generation (RAG) models to capture both semantic and syntactic similarities between queries and documents. This dual method increases retrieval accuracy by examining not only what the document says but how it is structured. Suitable for inquiries that require both semantic understanding and syntactic alignment between the user's input and the document's content.

### ***Model Integration and Workflow***

The above models are incorporated into a seamless system that utilizes a routing mechanism to choose which model to use based on the nature of the inquiry. The response generating component is driven by the underlying LLM to create high-quality summaries or customized insights. To help the retrieval process, the main vector database was generated by training on the whole library of research publications from JMLR and arXiv. This repository is divided into text chunks, which are embedded and stored as .pkl files, making it easy to import into any of the models.

Additionally, a search engine feature allows users to query the repository directly. This application may find related research papers based on abstract or content similarities, functioning as a thorough research helper for people seeking relevant literature in machine learning and artificial intelligence.

## ***Implement Designed System***

The final proposed solution is made of four intelligent components, each matching to a distinct use case, and offering users with different alternatives for researching and understanding machine learning literature. The system includes numerous models designed for adaptive, improved, multimodal, and hybrid retrieval of research papers, promoting extensive exploration and question-answering. The major use case is the retrieval of appropriate machine learning articles based on user queries. The user can also upload their own documents and ask questions, allowing for an interactive investigation of the text. Below is a breakdown of the primary functionalities:

- **Document Retrieval and Exploration:** The user can type a query or ask questions about specific ML topics. The system finds relevant research papers using Adaptive RAG with LLM (M1) and refines the results with Super RAG-Enhanced Retrieval (M2). The Multimodal RAG (M3) improves the results by merging text, tables, and figures, delivering comprehensive insights. The Hybrid RAG (M4) provides both semantic and syntactic matching to discover the most accurate papers.
- **Paper Upload and Question-Answering:** Users can upload their own research papers (e.g., PDFs), and the system processes these publications, extracting key sections (abstract, classifications, tables, etc.). Users can then ask precise questions regarding the content, and the system will provide extensive answers, leveraging the integrated RAG and LLM models.
- **Research Paper Search Engine:** The search engine utilizes embeddings to identify and prioritize pertinent research papers according to user inquiries. The metadata from an extensive dataset of research publications, including title, abstract, categories, and DOI, is

imported into a DataFrame. Abstracts are processed to produce sentence embeddings utilizing a pre-trained model, such as 'BAAI/bge-small-en-v1.5'. These embeddings are indexed using FAISS for efficient similarity retrieval. Upon entering a query, it is converted into an embedding, and the FAISS index is queried for the nearest matches. The search results yield the highest-ranking papers, organized by similarity scores. The title, DOI, abstract, and additional metadata for each paper are presented. The search engine collaborates with the Gemma LLM to produce summaries and insights from the most pertinent papers related to the question, offering succinct overviews of essential research findings.

### ***Input and Output Requirements, Supporting Systems and Solution APIs***

- Input datasets: The primary dataset comprises research publications taken from JMLR and arXiv. These documents are processed, and their text is chunked and embedded for easier retrieval. User queries serve as the secondary input to the models, allowing the system to provide similar papers or produce insights.
- Expected outputs: The system outputs a variety of findings, including retrieved research papers related to the user's query, coupled with created summaries or insights supplied by the LLM. For multimodal data, tables and figures are given in the response for a more informative analysis. For uploaded papers, users receive targeted responses to their questions along with relevant sections from the material.
- Supporting systems: The system is supported by cloud infrastructure to store and handle big databases of research publications. The models are trained and deployed using GPU and Google Colab resources to enhance performance. A search engine interface is implemented, enabling users to query and obtain relevant publications immediately.

- Solution APIs: Cohere API is applied for advanced embeddings, enabling for semantic search and clustering of research publications. Hugging Face API gives access to pre-trained transformer models, which are utilized for creating embeddings and text processing. This enables the system to retrieve the most relevant sections and deliver high-quality answers to user requests. The user interaction is handled using a web-based UI that includes real-time question-answering and retrieval possibilities.

#### **5.4 System Support Environment**

##### ***Google Colab and Jupyter Notebooks***

Google Colab and Jupyter Notebooks serve as the primary platforms for running and developing AI models, particularly for computationally intensive tasks like the Retrieval-Augmented Generation (RAG) system.

- Google Colab: Provides access to strong GPU acceleration, allowing for efficient AI model processing and training, which is critical for activities like large-scale data retrieval, real-time inference, and model fine-tuning in the RAG system. It allows for quick prototyping and execution of machine learning workflows in a collaborative, cloud-based environment without the need for local setup.
- Jupyter Notebooks: Provides a highly dynamic programming environment, perfect for iterative experimentation and thorough data analysis. It connects smoothly with local or cloud resources, allowing for rich visualizations and inline code execution.

Together, Google Colab and Jupyter Notebooks streamline the development and deployment of AI models by providing robust tools for experimentation, collaboration, and execution of complex AI workloads.

### ***Google Drive***

Acts as the central repository for storing the project's dataset. This cloud-based solution provides easy access and sharing of large PDF collections and other necessary files.

### ***GitHub***

Utilized as the version control system and code repository. This enables collaborative development, tracking changes, and maintaining different versions of the project codebase.

### ***Streamlit and Gradio for UI***

Streamlit and Gradio are python libraries for creating powerful web applications, particularly well-suited for generative ai and machine learning projects. It allows developers to build interactive web apps directly from Python scripts, without requiring extensive web development knowledge.

### ***Ollama***

Ollama is an open-source framework for executing large language models (LLMs) locally. It provides a varied library of pre-trained models, is simple to deploy, and enables GPU acceleration. Ollama supports customization, fine-tuning, and multimodal capabilities, as well as an active community for configuration sharing.

### ***Langchain***

LangChain is an open-source framework for developing applications that use large language models (LLMs). It includes modular components and tools for building AI-powered apps, such as the LangChain Expression Language for assembling action chains, interaction with external data sources, and support for Retrieval Augmented Generation. LangChain provides fundamental components such as prompt templates, LLM wrappers, and memory management, enabling a wide range of applications from chatbots to code creation.

### ***Hugging Face***

This library gives users access to innovative pre-trained models like BERT and GPT, making tasks like text creation, translation, and question answering easier to complete. Hugging Face's open-source platform promotes cooperation by letting developers exchange models and datasets, hence expediting AI research and application development. Hugging Face is critical in a PDF-based Retrieval Augmented Generation (RAG) system because it enables efficient document retrieval and question answering by exploiting sophisticated LLMs and embedding techniques. Its simple API and model repository make it easier to integrate and fine-tune models for domain-specific applications, improving overall system performance.

## **6. System Evaluation and Visualization**

### **6.1 Analysis of Model Execution and Evaluation Results**

This section gives an analysis of the execution and evaluation outcomes for the retriever-augmented generation (RAG) model, evaluated on a set of 20 documents in PDF format. For each document, a set of questions were presented to measure the model's ability to retrieve important information and create accurate responses. The responses were compared to ground truth answers, and several metrics were produced based on this comparison. These metrics, given in the sample figure, reflect a subset of the data to explain the format and structure of the results. The final values are averaged across all 20 PDFs, offering a thorough perspective of the model's overall performance, as shown in Table 6.

#### ***Ensemble Retriever RAG***

The results for the ensemble retriever RAG model give crucial insights into its performance. In terms of answer quality, the model displayed a context relevance score of 0.6075, an answer relevance score of 7.5 (on a scale of 1 to 10), a groundedness score of 0.8159, and an answer correctness of 0.7270. Additionally, the human judge score, representing evaluator impressions of the responses, was recorded as 7.72. These metrics show that the model works reasonably well in obtaining important context and providing accurate, grounded replies.

On the computational side, the model displayed an average response time of 10.15 seconds, with CPU consumption starting at 3.20% and finishing at 3.16%, and an average CPU usage during execution of 2.79%. GPU resources were more highly utilized, with an average GPU usage of 72.72%. These numbers show that the model is optimized for GPU processing, helping to its efficiency in handling resource-intensive activities.

## ***RAG Agent with LlaMa-3: Combined Approach using Adaptive RAG, Corrective RAG & Self-RAG***

This model leverages a combination of techniques to improve the quality of responses and the overall performance metrics are:

- Context Relevance: 8.67 - The model effectively selects relevant information from the knowledge base.
- Answer Relevance: 8.95 - The generated responses are highly relevant to the query.
- Groundedness: 8.755 - The responses are well-supported by evidence from the knowledge base.
- Answer Correctness: 8.41 - The responses are accurate and factually correct.
- Human Judge Score: 8.77 - Human evaluators rated the overall quality of the responses highly.

Overall, this model demonstrates strong performance across various correctness metrics, suggesting it effectively combines the strengths of different RAG techniques to generate high-quality responses.

### ***Adaptive RAG***

The Adaptive RAG with Routing model performs well on a number of evaluation metrics, demonstrating its capacity to find and produce precise, pertinent responses. The model is good at choosing relevant material from the knowledge base, as evidenced by its high Context Relevance score of 9.0, which guarantees that the context presented closely matches the question. Likewise, the model consistently generates answers that are precisely related to user inquiries, as evidenced by the Answer Relevance score of 9.0. There are sporadic holes that could be filled for even greater growth, even if the groundedness score of 7.0 indicates a general

solid foundation in knowledge base evidence. The model's high factual accuracy is confirmed by the Answer Correctness rating of 9.017, which further supports the model's dependability in providing accurate answers. Finally, the Human Judge number of 7.437 indicates that the model's quality has been received favorably generally.

### ***Multimodal RAG***

The model uses a combination of techniques to extract, process and then store images (graphs), texts and tables. Thus, improving the overall quality of the response generated. The performance metrics are :

- ***Context Relevance: 7.5*** (out of 10): In the majority of instances, the model exhibits a strong capacity to choose relevant data from the knowledge base (text, table, and picture), while remaining aligned with the query context.
- ***Answer Relevance: 7.8***: The responses provided are typically relevant to the queries, demonstrating that the model effectively tailors answers to the situation; nevertheless, minor modifications might increase relevance even more.
- ***Groundedness: 7.3***: The answers are generally well-supported by the knowledge base, while there are times when the underlying evidence might be better.
- ***Answer Correctness: 6.9***: While there is a good degree of accuracy, there is space to improve the factual accuracy of replies in order to increase overall reliability.
- ***Human Judge Score: 7.11***: Human assessors praised the general quality of the replies.

Overall, the model performs effectively and has a good degree of quality. Improvements in groundedness and accuracy might boost this score even further.

## ***Super RAG***

Overall, this model demonstrates strong performance across various correctness metrics, suggesting it effectively combines the strengths of various techniques to generate high-quality responses. The Super RAG model demonstrates a balanced performance across several key metrics, though there is some room for improvement in specific areas.

- Context Relevance: 9.0 (out of 10): The model does a great job of choosing pertinent data from the knowledge base and making sure that the context it provides is in line with the query.
- Answer Relevance: 8.625: The generated responses typically remain highly relevant to the question, demonstrating the model's ability to generate answers that are appropriate for the context.
- Groundedness: 6.0: Although the answers are rather well-founded, there are sporadic discrepancies in the strength of the knowledge base's evidence.
- Answer Correctness: 7.0 - Although there is room for improvement in terms of making sure all solutions are factually correct, the accuracy of the responses is good.
- Human Judge Score: 6.75 - The model's overall quality was deemed satisfactory, this rating might be further raised with increases in groundedness and correctness.

**Table 21**

*Comparison of Overall Correctness of RAG Answers*

*(Original from Team 8)*

<b>Model</b>	<b>Context relevance</b>	<b>Answer Relevance</b>	<b>Groundedness</b>	<b>Answer Correctness</b>	<b>Human Judge Score</b>
Adaptive RAG with Routing	9.0	9.0	7.0	9.017	7.437
Super RAG	9.0	8.625	6.0	7.0	7.75
Ensemble Retriever RAG	6.08	7.5	8.16	7.27	7.72
Multimodal RAG	7.1	7.4	7.30	6.90	7.11
RAG Agent with LlaMa-3	8.87	8.95	8.755	8.41	8.77

## 6.2 Achievements and Constraints

The implementation and evaluation of various RAG (Retriever-Augmented Generation) models have yielded significant advancements in enhancing user experiences during literature surveys and studies on machine learning (ML) literature. The models analyzed - Ensemble Retriever RAG, RAG Agent with LlaMa-3, Adaptive RAG, Multimodal RAG, and Super RAG - each contributed notable improvements in information retrieval, groundedness, answer correctness, and overall response quality. The key achievements and constraints encountered during this evaluation are summarized below.

### **Achievements**

1. Enhanced Contextual Retrieval: Models like RAG Agent with LlaMa-3, Adaptive RAG, and Super RAG demonstrated exceptional ability to retrieve relevant information from the knowledge base, leading to highly accurate and informative responses.
2. Improved Answer Quality and Relevance: The combined approaches of RAG Agent with LlaMa-3 and Adaptive RAG resulted in highly relevant and accurate answers, aligning closely with human expectations and enhancing user satisfaction.

3. Effective Multimodal Data Utilization: The Multimodal RAG model successfully processed text, images, and tables, providing more comprehensive and informative responses, particularly for literature involving complex visualizations.
4. Optimized Resource Utilization: Models like Ensemble Retriever RAG showcased efficient GPU utilization, ensuring smooth and timely responses to user queries during literature surveys.
5. Enhanced Correctness and Groundedness: Models like Adaptive RAG and RAG Agent with LlaMa-3 demonstrated strong performance in providing factually accurate and evidence-based answers, bolstering user trust in the generated responses.

### ***Constraints***

1. Variability in Factual Accuracy: While most models performed well in terms of answer correctness, certain models, like Multimodal RAG and Super RAG, displayed inconsistencies, necessitating further refinement in content validation.
2. Subjectivity in Human Evaluation: Human judge scores for Super RAG highlighted potential discrepancies between automated evaluation metrics and human perception of response quality, underscoring the need for more nuanced evaluation methods.
3. Resource Efficiency: While GPU utilization was optimized, CPU efficiency could be further improved to ensure overall system performance and resource allocation.
4. Adaptive Model Limitations: The Adaptive RAG model, despite its strengths, exhibited occasional inconsistencies in groundedness, suggesting the need for refinement in its routing mechanism to ensure reliable evidence retrieval.

In conclusion, these achievements demonstrate the significant progress made in RAG model development for literature surveys and ML research. Addressing the identified constraints,

such as improving groundedness, factual accuracy, and human-alignment, will further enhance the quality and reliability of user experiences in future iterations.

### 6.3 System Quality Evaluation of Model Functions and Performance

This section gives a comparative examination of various RAG (Retriever-Augmented Generation) models based on important system performance measures, including response time, CPU utilization (at the start, finish, and on average), and GPU usage. The results, as represented in Table 22, offer insight into how each model ideally balances speed and resource efficiency.

**Table 22**

*Comparison of Overall Resource Utilisation Metrics* *(Original from Team 8)*

Model	Response Time	CPU start usages	CPU end usages	Average CPU usages	Average GPU usages
Adaptive RAG with Routing	60.62	2.125	2.8	2.462	0
Super RAG	5.298	3.106	0.9625	2.034	75.564
Ensemble Retriever RAG	10.16	3.20	3.156	2.786	72.72
Multimodal RAG	5.43	4.47	4.82	3.73	33.36
RAG Agent with LlaMa-3	6.86	3.08	3.067	3.237	15.63

#### ***Response Time***

The response time metric demonstrates considerable variability across models. Adaptive RAG with Routing had the longest response time, averaging 60.62 seconds, possibly because of its extensive routing methods aimed to improve retrieval accuracy at the price of speed. On the other hand, Super RAG and Multimodal RAG exhibit notably faster response times of 5.298

seconds and 5.43 seconds respectively, demonstrating simplified retrieval procedures. RAG Agent with LLaMa-3 and Ensemble Retriever RAG have moderate response times of 6.86 seconds and 10.16 seconds respectively, which make them ideal for applications where a balance between accuracy and response time is needed.

### ***CPU Usage***

Examining CPU use illustrates how efficiently each model utilizes processor resources. Multimodal RAG stands out with the greatest average CPU utilization at 3.73%, followed by RAG Agent with LLaMa-3 with an average CPU usage of 3.237%. Super RAG has an unusual CPU trend, starting at 3.106% and reducing to 0.9625% by the end, with an average usage of 2.034%, showing it offloads more processing to the GPU over time. Adaptive RAG with Routing displays a consistent CPU pattern, starting at 2.125% and finishing at 2.8%, averaging 2.462%, while Ensemble Retriever RAG averages at 2.786%. Overall, Multimodal RAG and RAG Agent with LLaMa-3 display the highest CPU reliance, with other models maintaining lower, more steady CPU requirements.

### ***GPU Usage***

GPU utilization is a significant measure, as it often corresponds with response performance in models that may leverage parallel processing. Super RAG leads in GPU consumption, with an average usage of 75.564%, making it the most GPU-optimized model in the set. Ensemble Retriever RAG also substantially relies on GPU resources, with an average of 72.72%, suggesting its need on GPU capacity to accomplish difficult retrieval tasks. Multimodal RAG exhibits moderate GPU consumption at 33.36%, balancing CPU and GPU resources, while RAG Agent with LLaMa-3 utilizes just 15.63% of the GPU, demonstrating a design that reduces

GPU dependency. Adaptive RAG with Routing does not employ the GPU at all, with a consumption of 0%, demonstrating a fully CPU-based processing architecture.

Each model's capabilities and applicability for different applications can be understood through these metrics:

- For activities prioritizing speed, Super RAG and Multimodal RAG are appropriate due to their rapid response times and effective GPU utilization.
- In CPU-intensive systems where GPU resources are limited, Adaptive RAG with Routing or RAG Agent with LLaMa-3 may be preferable due to their lower GPU requirement.
- Ensemble Retriever RAG and Super RAG balance CPU and GPU resources, making them adaptable solutions for applications that demand efficient context retrieval while maintaining respectable processing speeds.

This examination demonstrates how each model improves performance according to its architecture, supporting varied use cases across real-time applications, CPU/GPU-constrained environments, and jobs demanding balanced resource allocation.

#### 6.4 System Visualization

To provide a thorough perspective of each model's performance across important metrics, bar charts have been generated for four primary parameters: response time, CPU utilization (beginning, end, and average), and GPU usage. The results, as illustrated in Figures 15-18, offer insight into how each model properly balances speed and resource efficiency.

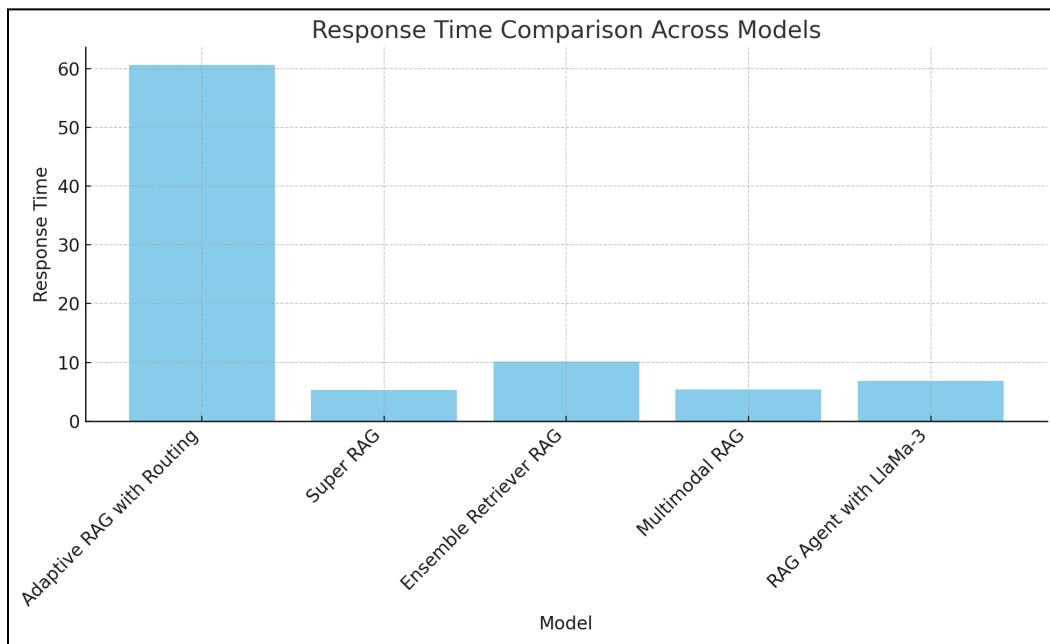
The response time chart, as shown in Figure 24, displays a clear disparity among the models in terms of processing speed. Adaptive RAG with Routing displays the highest reaction time by a large margin, reflecting its complicated routing algorithm, which favors accuracy over speed. In contrast, Super RAG and Multimodal RAG have the shortest reaction times, followed

closely by RAG Agent with LLaMa-3 and Ensemble Retriever RAG. This visual representation emphasizes Super RAG and Multimodal RAG as the most time-efficient approaches.

**Figure 50**

*Comparison of Response Time*

*(Original from Team 8)*

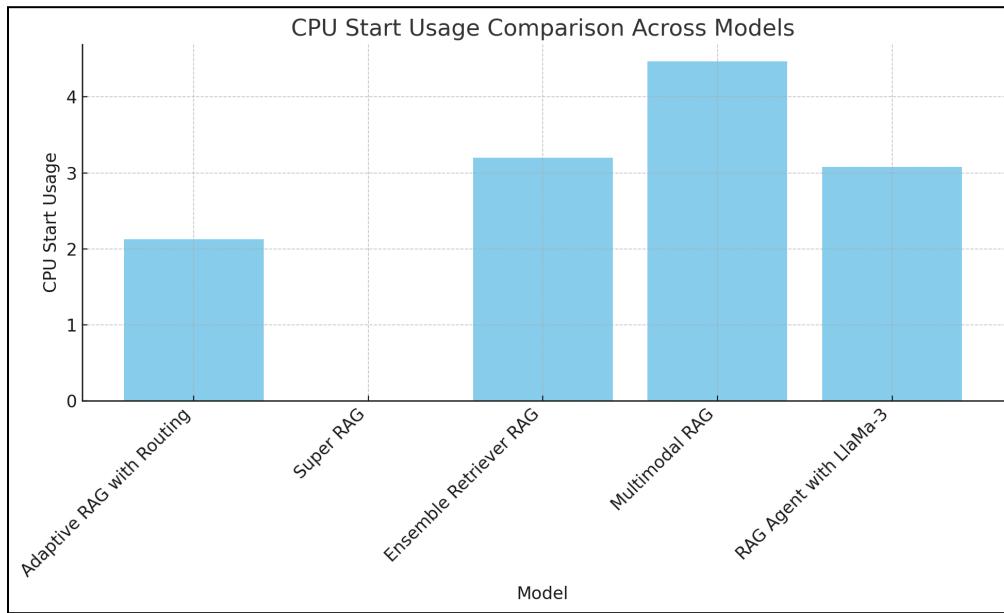


The CPU utilization charts, given in Figures 50-52, illustrate each model's demand on CPU resources at the start and end of processing, as well as the total average CPU usage during execution. Multimodal RAG displays the highest average CPU consumption, reflecting its balanced reliance on both CPU and GPU resources. Super RAG, strangely, begins with rather high CPU consumption but exhibits a decline toward the conclusion, suggesting an early dependency on CPU resources before shifting to the GPU. Ensemble Retriever RAG and RAG Agent with LLaMa-3 display moderate CPU consumption, while Adaptive RAG with Routing demonstrates consistent, but lower, CPU utilization, due to its CPU-based processing strategy.

**Figure 51**

*Comparison of CPU Start Usage*

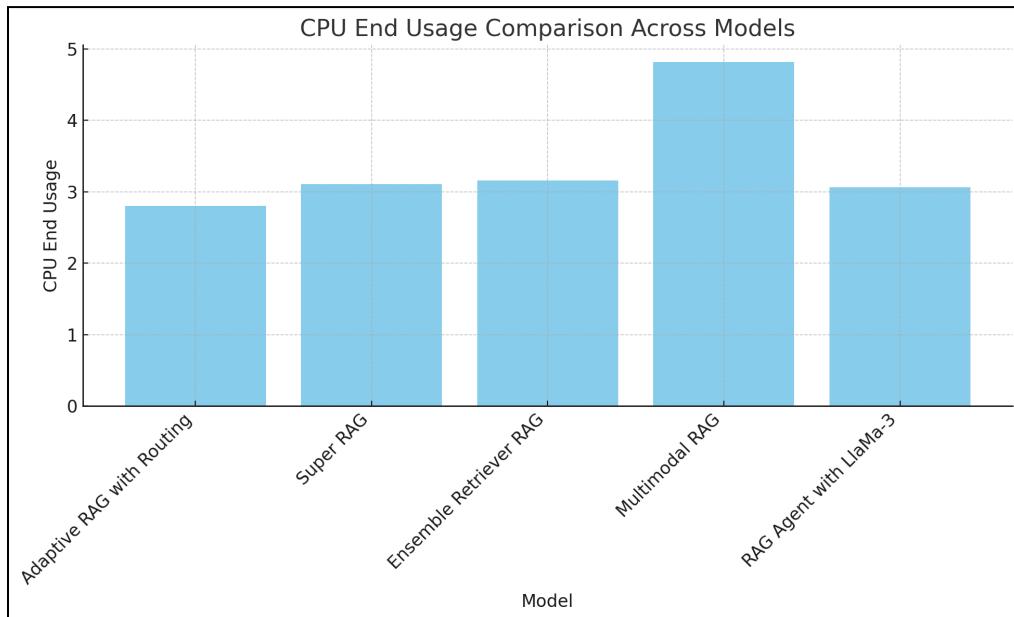
(Original from Team 8)



**Figure 52**

*Comparison of CPU End Usage*

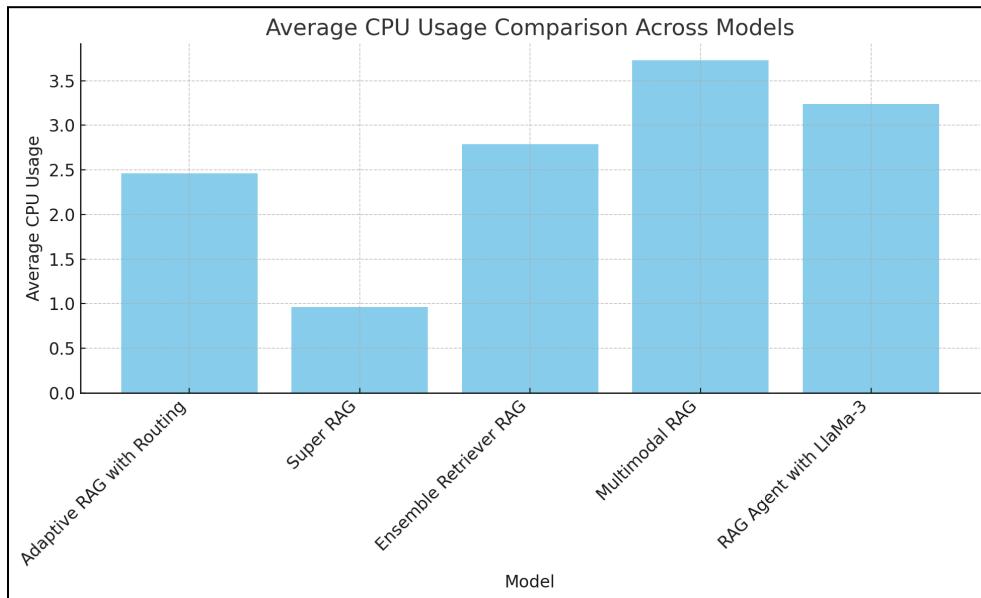
(Original from Team 8)



**Figure 53**

*Comparison of Average CPU Usage*

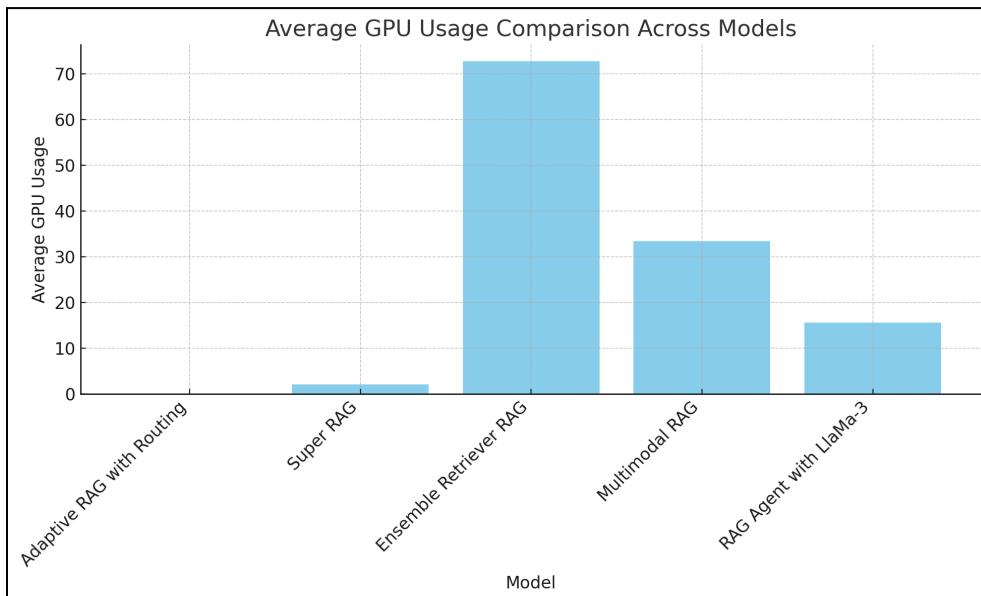
(Original from Team 8)



**Figure 54**

*Comparison of Average GPU Usage*

(Original from Team 8)



The GPU utilization chart, as illustrated in Figure 54, demonstrates that Super RAG has the most dependency on GPU resources, followed closely by Ensemble Retriever RAG, both of

which substantially employ the GPU to achieve faster response times. Multimodal RAG demonstrates moderate GPU consumption, showing a balanced approach between CPU and GPU resources. In contrast, RAG Agent with LLaMa-3 displays little GPU dependency, whereas Adaptive RAG with Routing does not use GPU resources at all, relying exclusively on the CPU. This visual analysis underlines the GPU optimization in Super RAG and Ensemble Retriever RAG and the CPU dependency of Adaptive RAG with Routing.

**Figure 55**

*Performance Comparison Across Key Evaluation Metrics*

(Original from Team 8)

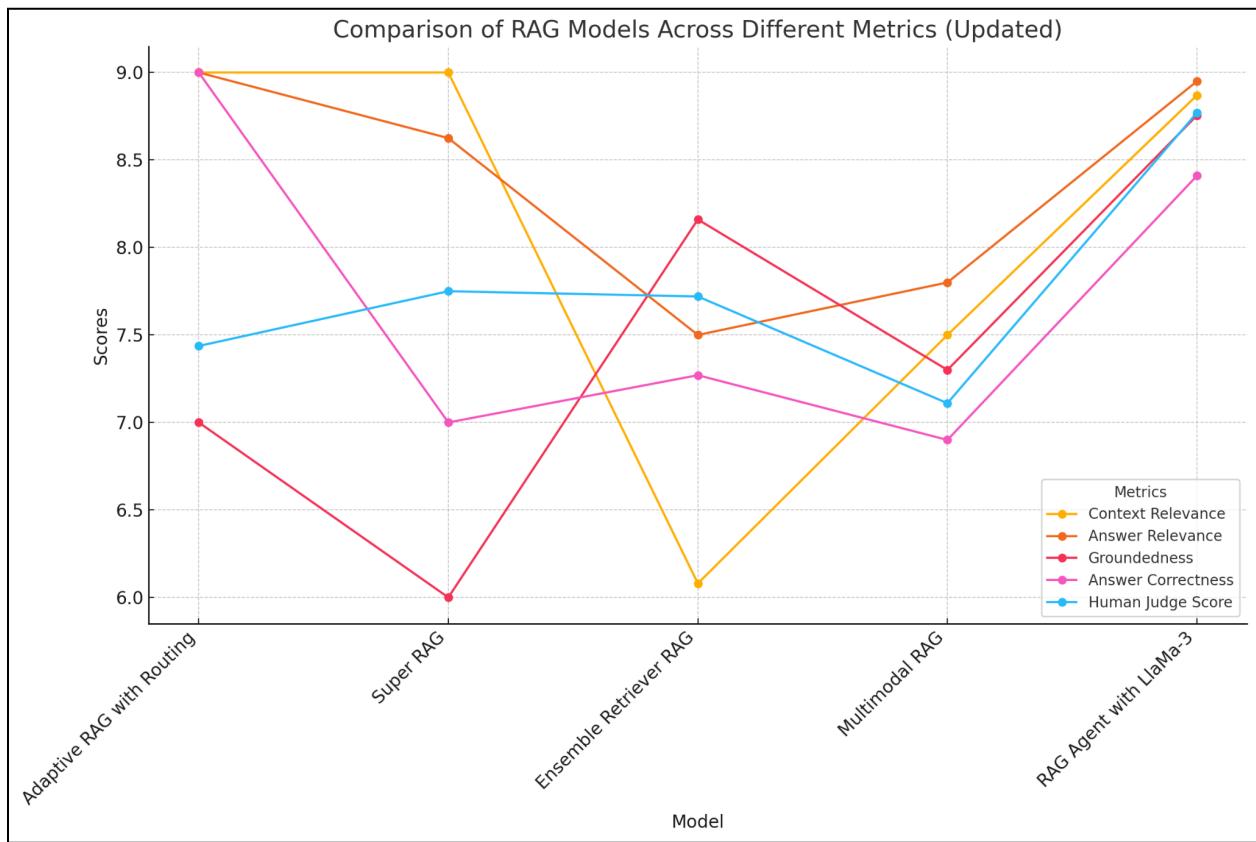


Figure 55 compares the five RAG variant models based on the following metrics: context relevance, answer relevance, groundedness, answer correctness, and human judge score. Super

Rag shows the best “Context Relevance” score whereas Adaptive Rag has the highest “Answer Correctness” score.”Answer Relevance” is highest for a RAG agent with Llama 3.

## 7. Conclusion

### 7.1 Summary

This research project built a comprehensive data and project management structure to provide a scientific literature review assistant for Machine Learning (ML) and Artificial Intelligence (AI). The project established a solid platform for model development and implementation by methodically obtaining, organizing, and maintaining research datasets.

The project used two main datasets: the arXiv dataset, which contains around 30k STEM-related research papers, and the Journal of Machine Learning Research (JMLR) dataset, which has over 2,800 ML-specific publications. The research articles were retrieved and analyzed, including titles, authors, abstracts, and comprehensive study findings, which enhanced the application's knowledge base for next steps.

The project used the CRISP-DM lifecycle, an organized procedure that ensures flexibility and iterative refinement at each stage. Specific objectives were established to improve accessibility and comprehension of ML/AI literature for both technical and non-technical readers. Extensive efforts were taken to assure data quality, relevance, and analytical readiness, including the creation of vector databases for effective embedding storage. The project is capable of dynamic, context-aware interactions via the integration of advanced AI frameworks (e.g., LangChain, FAISS, LLAMA) with interactive platforms (e.g., Streamlit and Gradio).

Following project completion, the dataset entered a maintenance phase and was archived in Google Drive for future academic or professional usage. Using metadata categories like "cs.AI" and "cs.LG" from the arXiv dataset, documents were filtered to only contain publications that were relevant to machine learning (ML) and artificial intelligence (AI). Withdrawn documents and null records were eliminated. To ensure consistency across various document

formats, text collected from PDFs was normalized. In order to integrate table annotations and captions with other data formats, they were converted into vector representations. Further, to improve contextual awareness, text chunks were converted into embeddings using models such as all-MiniLM-L6-v2. At last, for effective retrieval, unified embeddings of text and image data were kept in vector stores.

Training (80%), validation (10%), and test (10%) subsets were created from the combined dataset. A balanced approach to model training and evaluation was guaranteed by this methodical division. Assuring scalability and cost efficiency, adaptive RAG reliably directs questions to an optimal retrieval mechanism based on the complexity. Corrective and Self-Reflective RAG techniques provided dependable and legitimate responses by improving response accuracy and minimizing hallucinations. With Super RAG, real-time data retrieval is incorporated with minimal structural changes, giving the LLMs access to current external information sources for better answer quality. To effectively handle and interpret large-scale datasets, the Hybrid RAG-LLaMA model uses ensemble retrieval techniques, making it ideal for intricate research domain queries. Multimodal RAG integrated text, tables, and images, broadening the range of information retrieval and enabling comprehensive analysis of scientific and technical documents.

Computational efficiency and retrieval efficacy were improved by fine-tuning vector storage, chunk sizes, and embedding models. By recognizing trade-offs between speed, accuracy, and scalability, comparative research on vector stores (such as FAISS and ChromaDB) assisted to guide the development of the optimal approach. To assure credible, cost-free validation, auto-rating systems were put in place together with specially designed evaluation criteria (RAG

Triad: Context Relevance, Answer Relevance, and Groundedness). Metrics including throughput, response time, and resource usage were all routinely tracked and improved.

Adaptive RAG: Demonstrated high scalability through intelligent and robust prompt routing. It ensured grounded, high-quality responses to diverse query types such as factual and analytical. Super RAG achieved dynamic knowledge integration using real-time data retrieval. It adapted seamlessly to evolving information sources, making it suitable for high-demand environments. Hybrid RAG shines in precision retrieval by combining dense and sparse methods. It optimized response creation for complex academic and research queries. Multimodal RAG processed diverse data formats, including text, tables, and images. It overcame the limitations of traditional text-only retrieval systems.

## 7.2 Benefits and Shortcomings

The benefits and shortcomings of the project are listed out and discussed in detail below.

### ***Benefits***

1. Efficient Literature Exploration: By making it easier to carry out literature reviews and extract information, helps researchers to quickly locate relevant study papers.
2. Multimodal Capabilities: Compared to conventional text-only retrieval methods, the system offers a more comprehensive understanding of complicated information by integrating textual, tabular, and visual data.
3. Advanced Search: This robust resource for academic research uses high-precision retrieval with vector databases and embedding models to enhance relevance
4. Enhanced Query Processing

- a. Adaptive Mechanisms: To improve the quality and relevance of results, the system continually alters retrieval algorithms according to the complexity of the query.
- b. When internal information is insufficient, including web-based resources such as Wikipedia and Tavily boosts robustness by expanding knowledge boundaries
- c. LLM hallucinations are reduced by self-correcting mechanisms, which guarantee that the generated reactions are precise and well-founded.

## 5. AI and Machine Learning Integration:

- a. Leveraging models like MiniLM and BAAI ensures high-dimensional semantic understanding for precise retrieval.
- b. Multiple Model Architecture: RAG models that are adaptive, super, hybrid, and multimodal handle a variety of application scenarios, ranging from multimodal data processing to text retrieval.
- c. User-Friendly Interface: Streamlit and Gradio give the system an easy-to-use user interface that makes it suitable for users with different levels of technical proficiency.

## ***Shortcomings***

### 1. Data and Model Dependency

- a. Restricted by Corpus: The quantity and caliber of the content that is retrieved has a significant impact on the research articles that are accessible. To expand outside the current corpus (like arXiv and JMLR), more data integration is needed.
- b. The limitations of embedding models include that, although they perform well, they may need to be adjusted when dealing with highly specific or niche requests.

2. Computational Resource Demands:
  - a. High Hardware Requirements: Using advanced RAG models and LLMs increase costs for operation because they demand powerful GPUs
  - b. Latency for Complex queries: Complex multimodal or fallback searches may result in greater latency, even while standard response times are acceptable.
3. Complexity in Integration: Dependency on External APIs: Using APIs such as Hugging Face, Tavily, and Cohere creates a number of possible concerns, including availability, cost, and reliance.
4. Challenges for Multimodal Data
  - a. Handling and embedding non-textual data (such as tables and graphs) involves substantial preprocessing, which may impact scalability as the corpus increases.
  - b. Increasing Resource Utilization: Multimodal computing takes more computing power, hence it requires to be better optimized before being widely used.
  - c. The model's capabilities and training significantly impact the graph interpretation's quality and accuracy. Failure to train the model for recognizing subtle patterns in graphical data or a lack of domain-specific knowledge could end up in misinterpretations or oversights.
5. Evaluation and Accuracy
  - a. Dependency on Human Input: Even with strong evaluation metrics, certain nuanced factors can still need human intervention, particularly with extremely complicated or ambiguous queries.

- b. Despite being mitigated by corrective and self-reflective RAG, the risk of hallucinations through LLM can still make it difficult to offer responses that are grounded on facts.
6. Accessibility and Usability
- a. The learning curve for complex features may be difficult for researchers who have no previous experience with AI processes, though the user interface is straightforward to use.
  - b. Restricted Offline Access: Depending too much on cloud-based solutions may render them harder to utilize in settings with inconsistent internet access.

### **7.3 Potential System and Model Applications**

The developed system and models offer a wide range of applications across academic and research domains, showcasing their versatility and potential to drive innovation and efficiency in data management and retrieval.

#### ***Research Assistance for Machine Learning and AI***

The system significantly streamlines the process of conducting literature reviews by summarizing and organizing research papers. Researchers can efficiently retrieve and analyze academic texts, enabling a deeper understanding of complex topics. By leveraging retrieval-augmented generation (RAG) models, the system supports detailed exploration of machine learning methodologies, offering contextualized insights directly from academic sources.

#### ***Interactive Educational Tools***

Designed to support learning, the system answers technical and conceptual questions with relevant, concise explanations. This capability facilitates academic exploration, enabling students

and professionals to interactively query machine learning methods and receive tailored responses. The system's ability to dynamically adapt responses based on query complexity enhances its utility as an educational aid.

### ***Enhanced Data Analysis and Insight Extraction***

The system processes multimodal content, including text, tables, and images from research papers, enabling comprehensive synthesis of information. Researchers can identify trends, extract key insights, and analyze diverse data formats, such as diagrams and tabular data, enhancing their ability to derive actionable conclusions, particularly in technical and scientific contexts.

### ***Industry and Enterprise Integration***

The system adapts seamlessly to large-scale data retrieval needs in enterprise settings. By integrating with external databases and APIs, it provides robust support for decision-making processes, dynamically retrieving current and relevant information. This capability is particularly beneficial for organizations requiring timely and accurate insights to drive strategic decisions.

### ***Scalable Query and Retrieval Frameworks***

The modular RAG-based architecture supports applications that demand high-speed, scalable, and precise data retrieval. Its design is well-suited for real-time systems, where low latency and high accuracy are critical. By enabling scalability and adaptability, the system addresses the growing demand for efficient data processing in complex, high-volume environments.

In conclusion, the system's capabilities extend beyond academic research to support educational, industrial, and large-scale operational needs. Its versatility ensures it can meet the

diverse requirements of users, driving innovation and informed decision-making across multiple domains.

#### **7.4 Experience and Lessons Learned**

The project provided valuable learning opportunities, offering technical, collaborative, and strategic insights that will inform future work in the field of retrieval-augmented generation and AI-driven research systems.

##### ***Technical Insights***

- *Model Performance:* By exploring various RAG configurations, the team learned the importance of tailored strategies for handling diverse retrieval and generation needs. The adaptive, corrective, and self-reflective mechanisms significantly enhanced the accuracy and reliability of the system's responses.
- *Optimization Techniques:* Experimentation with embedding models and vector stores highlighted the trade-offs between speed, accuracy, and relevance. For instance, while FAISS demonstrated exceptional speed, ChromaDB excelled in relevance, emphasizing the importance of selecting tools based on specific requirements.
- *System Integration:* Implementing a multi-modal retrieval system required mastering the interoperability of various tools, such as FAISS, ChromaDB, and external APIs like Tavily. This integration underscored the complexity of designing a cohesive system that balances efficiency with functionality.

##### ***Challenges Encountered***

- *Data Management:* Managing large-scale data while maintaining performance posed a significant challenge. This was addressed through innovative chunking and retrieval strategies that optimized processing efficiency.

- *Computational Resource Balancing*: Iterative testing and optimization were necessary to balance resource usage with response quality, ensuring the system's scalability and sustainability.
- *Addressing Hallucinations*: Large language model (LLM) hallucinations were mitigated by integrating corrective mechanisms, which ensured that generated responses were accurate and grounded in the retrieved context.

### ***Collaboration and Teamwork***

The team's diverse expertise contributed significantly to the project's success. Members specialized in areas such as model development, evaluation metrics, and user interface design, ensuring a well-rounded approach to system development. Regular feedback loops and collaborative problem-solving ensured alignment with project objectives and optimized outcomes.

### ***Broader Impacts***

This project highlighted the transformative potential of AI in research, enabling deeper engagement with academic literature and promoting the efficient dissemination of knowledge. The insights gained from this work could guide the development of similar systems in other domains, fostering innovation and accessibility in information retrieval.

### ***Future Considerations***

- *Scalability and Adaptability*: Enhancing the system's scalability and adaptability to support larger user bases and more complex queries remains a priority.
- *Hybrid and Ensemble RAG Configurations*: Further exploration of these configurations may improve both accuracy and efficiency, addressing the evolving needs of users.

In summary, this project not only demonstrated the practical application of advanced retrieval techniques but also provided a foundation for future research and development in the field. The lessons learned will inform ongoing efforts to refine and expand the system's capabilities, ensuring its relevance and impact in a rapidly evolving technological landscape.

## **7.5 Recommendations for Future Work**

The following are suggestions and possible iterations that can be considered for future enhancements to the project.

### ***Expanding Domain Coverage***

The application we developed currently specializes in machine learning and artificial intelligence, limiting its scope to these specific domains. Future development could expand its applicability to other critical areas such as medicine, climate science, social sciences, and engineering. Integrating domain-specific datasets and knowledge bases could facilitate interdisciplinary research, allowing users to explore and utilize knowledge across a broader spectrum of disciplines.

### ***Support for Other Languages and Enhanced Natural Language Interaction***

Future iterations of the application could include introducing multilingual capabilities, making it more accessible to a global audience and breaking language barriers in academic research. These capabilities could enable querying and summarizing research in various languages, such as French, Spanish, Chinese, and others. Additionally, usability improvements could focus on making the tool more intuitive for non-technical users.

Features like guided query suggestions to help frame effective questions, interactive tutorials, tooltips for in-app guidance, and voice-enabled commands for hands-free interaction would enhance accessibility. These enhancements would ensure that the assistant not only

supports multiple languages but also caters to a diverse range of users with varying levels of technical expertise.

### ***Integration with Publishing Platforms***

Another compelling idea would be to integrate the application we developed with preprint repositories and open-access platforms such as arXiv, bioRxiv, or PubMed Central, creating a one-stop solution for accessing and querying these papers. This integration would enable researchers to stay updated on the latest advancements in their fields by exploring cutting-edge research before formal publication. Additionally, the assistant could incorporate features to track citation trends and notify users of new papers that reference significant works in their area of interest. By providing a centralized platform for accessing, querying, and analyzing academic papers, the application would streamline research workflows and establish itself as an essential research tool.

### ***Continuous Learning and Updates***

Incorporating mechanisms for continuous learning would ensure that the application remains aligned with the latest research developments and methodologies. While the current system leverages sources such as Wikipedia and the Tavilly Search API for general knowledge retrieval, the vector store is currently limited to over 25,000 Machine-learning and AI papers. To maintain relevance in a rapidly evolving field, periodic updates to the vector store should be implemented by continuously ingesting newly published papers from reputable sources like arXiv, IEEE Xplore, or related conferences such as NeurIPS, ICML, and CVPR.

Additionally, integrating real-time indexing capabilities could allow the vector store to dynamically include the latest research, ensuring that the assistant stays current with cutting-edge advancements. By automating the ingestion and indexing of these papers into the vector store,

the application would not only expand its scope but also enhance the accuracy and timeliness of its responses, ensuring it remains a reliable and valuable resource for researchers navigating the ever-growing landscape of scientific literature.

## **7.6 Contributions and Impact on Society**

### ***Key Contributions of the Project***

We developed a Literature Review Assistant to revolutionize the way machine learning research is explored and synthesized. By integrating state-of-the-art Retrieval-Augmented Generation (RAG) frameworks—such as Adaptive RAG, Super RAG, and Local RAG Agent—alongside multimodal and hybrid models, and pairing them with a robust vector database and embedding models, we have achieved significant milestones, including the ability to process over 25,000 scientific papers across domains such as Machine Learning and AI.

We created a powerful tool that delivers precise, context-aware information, enabling researchers to efficiently navigate and extract insights from the vast scientific literature. A key innovation of the application lies in its multimodal capabilities, which extend beyond textual data to extract insights from other formats such as tables, graphs, and images, ensuring comprehensive analysis across diverse formats.

By incorporating advanced fallback mechanisms, knowledge graphs, hallucination checks, feedback scoring, semantic search techniques, and a comprehensive evaluation framework, the tool ensures high levels of accuracy and reliability, effectively minimizing errors and preventing hallucinations. Its modular design further enhances dependability by dynamically rerouting queries to external search APIs such as TavilySearchAPI and Wikipedia when necessary, ensuring consistent and relevant results.

Automating the time-consuming and repetitive aspects of literature reviews is a significant contribution because it allows researchers to shift their focus from manual tasks to more strategic and intellectually demanding endeavours. By streamlining processes such as searching, filtering, summarizing, and synthesizing vast amounts of information, the tool saves valuable time and resources. This is particularly useful in fast-evolving fields like machine learning and artificial intelligence, where staying updated with the latest research is crucial.

Moreover, the tool enables researchers to quickly identify trends, gaps, and connections across studies, facilitating deeper insights and more informed decision-making. It democratizes access to advanced research capabilities, making it equally valuable for seasoned scientists, educators, and students alike.

Lastly, we have also made strides in user accessibility by building an intuitive interface using platforms like Streamlit and Gradio. This user-centric approach empowers a wide audience—researchers, educators, and students alike—to seamlessly interact with features such as PDF uploads, adaptive retrieval, real-time responses, and capabilities for summarization, citation generation, and question-answering. Advanced functionalities like data augmentation and multimodal retrieval significantly broaden its utility for complex scientific inquiries.

In summary, this application stands out as a vital tool for advancing scientific research by combining technical innovation with user-focused features. Its ability to process PDFs, handle both textual and multimodal data, and deliver accurate, context-aware insights simplifies and enhances the research process, making it a preferred choice among researchers, educators, and students. By democratizing knowledge and broadening its reach, the tool strives to make research more accessible and impactful for a diverse audience. Its precision, adaptability, and

reliability cement its role as a highly valuable resource for advancing understanding through research and contributing to knowledge expansion.

### ***Contributions to Cultural, Economic, Educational, and Social Well-Being***

The Literature Review Assistant makes a significant impact across cultural, economic, educational, and social domains, by enhancing accessibility to research and simplifying complex processes such as navigating vast data, synthesizing information, and extracting meaningful insights.

Culturally, it fosters inclusivity by providing linguistically accessible and diverse research summaries, enabling collaboration across global boundaries. For instance, a researcher in a developing country with limited access or no access to expensive journal subscriptions can use the application to explore summaries of cutting-edge AI studies, bridging gaps in knowledge-sharing and empowering contributions to global innovation. Economically, the tool reduces the essence of manual literature reviews, saving startups and companies significant time and resources. For example, they can quickly identify advancements in areas such as transformer architectures, reducing research time and accelerating product development and deployment. Additionally, users can upload specific papers of interest and ask targeted questions, allowing them to extract relevant insights directly without reading through entire documents. The tool also eliminates the need for multiple other subscriptions by providing access to answers, summaries and insights, further enhancing efficiency and reducing costs. Educationally, its intuitive design simplifies complex topics, making it an invaluable resource for educators and students. A high school teacher introducing AI ethics, for instance, can use the tool to curate simplified insights, enabling students to navigate academic studies effectively while fostering independent learning. Socially, the assistant makes technical knowledge more accessible, enabling individuals and

communities to address practical challenges. For example, a small business owner could use the tool to explore research on AI and ML-related technologies, helping them stay updated and improve their services to better meet customer needs.

### ***Impact on Society***

The Literature Review Assistant significantly impacts society by streamlining research processes and driving advancements in machine learning and artificial intelligence. By processing extensive collections of research papers, it fosters cross-disciplinary collaboration and equips users to address complex challenges while expanding scientific knowledge. Its intuitive design broadens access to research, benefiting educators, students, researchers, and individuals from diverse backgrounds, and supporting the advancement of STEM education.

It reduces resource waste, supports energy efficiency, boosts productivity, and improves decision-making, demonstrating how AI can enhance human capabilities and promote knowledge-sharing. For instance, a researcher studying natural language processing models, such as transformers, can upload PDFs, pose specific questions (e.g., “What are the latest optimization techniques for transformer models?”), and receive concise, actionable insights in minutes, saving time and streamlining research efforts.

Ultimately, the Literature Review Assistant simplifies complex scientific research, enabling society to process vast amounts of data, extract meaningful insights, and keep pace with rapidly evolving fields, driving progress more efficiently than ever before.

## References

- Ainapure, A., Dhamane, S., & Dhage, S. (2023, October). Embodied Epistemology: A Meta-Cognitive Exploration of Chatbot-Enabled Document Analysis. In *2023 International Conference on Evolutionary Algorithms and Soft Computing Techniques (EASCT)* (pp. 1-6). IEEE. <https://ieeexplore.ieee.org/abstract/document/10392618>
- Asai, A., Wu, Z., Wang, Y., Sil, A., & Hajishirzi, H. (2023). Self-rag: Learning to retrieve, generate, and critique through self-reflection. arXiv preprint arXiv:2310.11511. <https://arxiv.org/pdf/2310.11511>
- Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*. <https://arxiv.org/abs/1810.04805>
- Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., ... & Wang, H. (2023). Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*. <https://arxiv.org/pdf/2312.10997>
- Han, Y., Liu, C., & Wang, P. (2023). A comprehensive survey on vector database: Storage and retrieval technique, challenge. *arXiv preprint arXiv:2310.11703*. <https://arxiv.org/abs/2310.11703>
- Hsieh, C. Y., Li, C. L., Yeh, C. K., Nakhost, H., Fujii, Y., Ratner, A., ... & Pfister, T. (2023). Distilling step-by-step! outperforming large language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*. <https://arxiv.org/abs/2305.02301>
- Khadija, M. A., Aziz, A., & Nurharjadmo, W. (2023, October). Automating Information Retrieval from Faculty Guidelines: Designing a PDF-Driven Chatbot powered by OpenAI ChatGPT. In *2023 International Conference on Computer, Control, Informatics*

*and its Applications (IC3INA)* (pp. 394-399). IEEE.

<https://ieeexplore.ieee.org/abstract/document/>

Jeong, S., Baek, J., Cho, S., Hwang, S. J., & Park, J. C. (2024). Adaptive-RAG: Learning to Adapt Retrieval-Augmented Large Language Models through Question Complexity. *arXiv preprint arXiv:2403.14403*. <https://arxiv.org/pdf/2403.14403.pdf>

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. D. L., ... & Sayed, W. E. (2023). Mistral 7B. *arXiv preprint arXiv:2310.06825*.  
<https://arxiv.org/pdf/2310.06825>

Siriwardhana, S., Weerasekera, R., Wen, E., Kaluarachchi, T., Rana, R., & Nanayakkara, S. (2023). Improving the domain adaptation of retrieval augmented generation (RAG) models for open domain question answering. *Transactions of the Association for Computational Linguistics, 11*, 1-17.

[https://direct.mit.edu/tacl/article/doi/10.1162/tacl\\_a\\_00530/114590/](https://direct.mit.edu/tacl/article/doi/10.1162/tacl_a_00530/114590/)

Team, G., Mesnard, T., Hardin, C., Dadashi, R., Bhupatiraju, S., Pathak, S., ... & Kenealy, K. (2024). Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*. <https://arxiv.org/pdf/2403.08295>

Thakur, A., & Gupta, R. (2024). Introducing Super RAGs in Mistral 8x7B-v1. *arXiv preprint arXiv:2404.08940*. <https://arxiv.org/pdf/2404.08940v1.pdf>

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M. A., Lacroix, T., ... & Lample, G. (2023). Llama: Open and efficient foundation language models. CoRR, abs/2302.13971, 2023. doi: 10.48550. *arXiv preprint arXiv:2302.13971*. <https://arxiv.org/abs/2302.13971>

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

[https://proceedings.neurips.cc/paper\\_files/paper/2017](https://proceedings.neurips.cc/paper_files/paper/2017)

Yan, S. Q., Gu, J. C., Zhu, Y., & Ling, Z. H. (2024). Corrective Retrieval Augmented Generation. arXiv preprint arXiv:2401.15884. <https://arxiv.org/pdf/2401.15884>

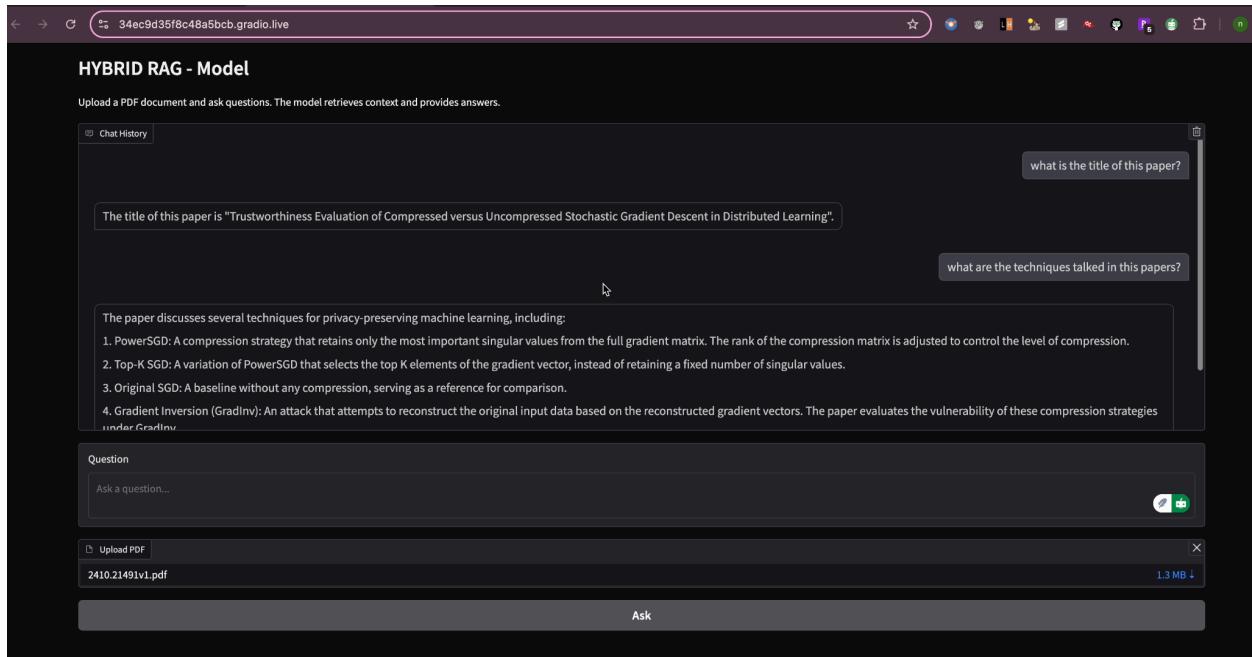
Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., ... & Wen, J. R. (2023). A survey of large language models. *arXiv preprint arXiv:2303.18223*.

<https://arxiv.org/abs/2303.18223>

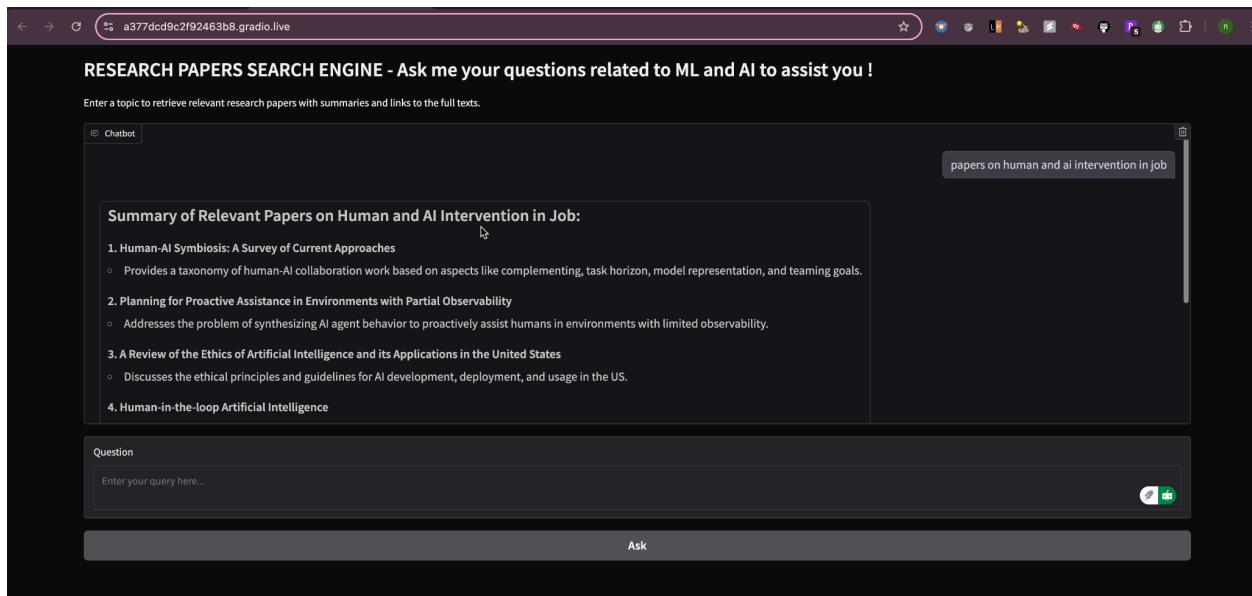
## Appendix A – System Testing

The GUI Screens for each Use Case can be seen in Figures A1 to A9. All the figures are Original from Team 8.

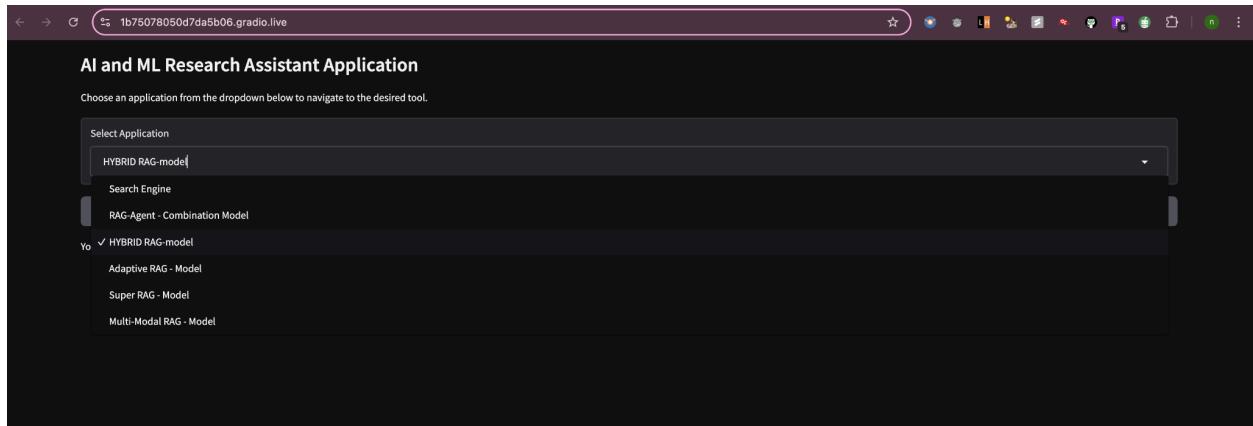
**Figure A1. Answering Technical Queries about an Uploaded Study**



**Figure A2. Research Paper Search Engine (Retrieve Similar Papers)**



**Figure A3. Unified App to switch between different model configurations/variants**



**Figure A4. Multimodal Content Extraction (Use Case: Handle Multimodal Content)**

Extracting content from the PDF...

Content extraction completed! Displaying summaries...

**Original Table Content:**

Name of kernel K7( $\|x - 2\|$ )/A7(0) a) Circular  $a e + t a i y T R 2 2 \arccos(\|x - 2\| \|x - 2\| \|x - 2\|)$  positive definite in  $R^2$   $2 \arccos(7 a 5 1 - (4^*) \text{ if } \|x - 2\| < 0 \text{ otherwise}$  b) Spherical . . . 3 positive definite in  $R^3$  1- gal + 4 (Peal) if  $\|x - z\| < 0$  zero otherwise c) Rational quadratic us a, od  $|x - z|^2$  positive definite in  $R^2$   $T - a [240 d] \text{ Exponential positive definite in } R^2 \exp(-\|x - z\|)$  e) Gaussian  $-y^2/2$  positive definite in  $R^2 \exp(-\|x - z\|^2)$  f) Wave positive definite in  $R^2$   $8 \sin(274 \|x - z\|)$

**Table Summary**

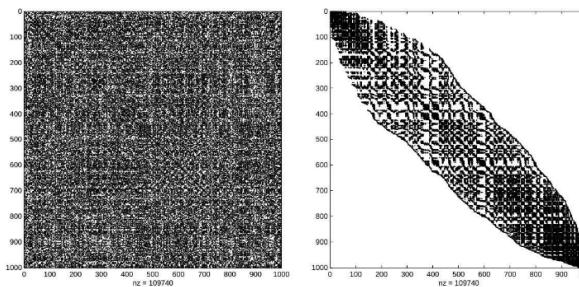
Here are the values, trends, and data points extracted from the table or text chunk:

**Values:**

1.  $\|x - 2\|$
2. A7(0)
3. a
4. t
5. a
6. e
7. l
8. 2
9. 4
10. a
11. 7
12. 5

**Figure A5. Image Extraction (Use Case: Handle Multimodal Content)**

context or information about the graph, I'd be happy to help you extract more specific details!



Extracted Image: figure-8-3.jpg

**Image Summary**

Here are the important elements extracted from the scientific graph:

**Axis Labels and Units:**

- X-axis: Not labeled, but appears to be a continuous scale with values ranging from 0 to 1000.
- Y-axis: Not labeled, but appears to be a continuous scale with values ranging from 0 to 109740.

**Scale Markers and Numerical Values:**

- X-axis:
  - 100
  - 200
  - 300
  - 400
  - 500
  - 600
  - 700

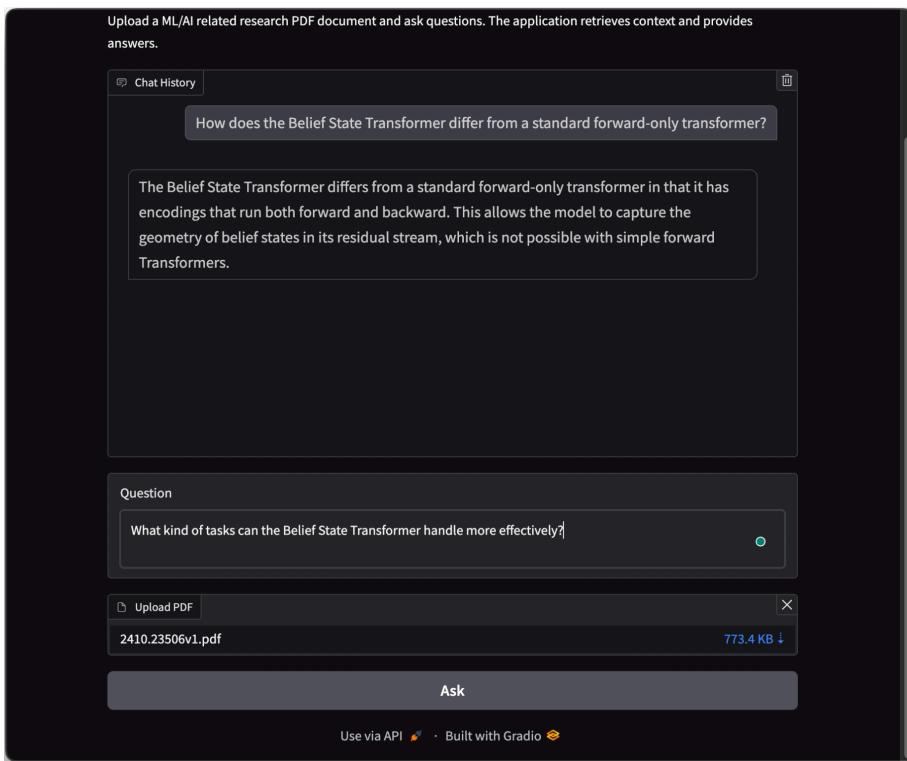
**Figure A6. Q&A on Extracted Content**

The screenshot shows a user interface titled "Ask a Question". The user has entered the question "whats the value of nz". The application has generated a detailed JSON response object:

```

{
  "content": "According to the text, the value of `nz` is mentioned as `nz = 109740`.",
  "additional_kwargs": {},
  "response_metadata": {
    "token_usage": {
      "completion_tokens": 22,
      "prompt_tokens": 4792,
      "total_tokens": 4814,
      "completion_time": 0.018333333,
      "prompt_time": 0.231790283,
      "queue_time": 0.002260815999999999,
      "total_time": 0.250123616
    },
    "model_name": "Llama3-8b-8192",
    "system_fingerprint": "fp_179b0f92c9",
    "finish_reason": "stop",
    "logprobs": null
  },
  "type": "ai",
  "name": null,
  "id": "run-7b146dd5-a905-46c6-b636-f55e15d8439a-0"
}
  
```

**Figure A7. RAG Agent with Adaptive, Corrective, Self-Reflective Approaches**



**Figure A8. Adaptive RAG with Routing**

**Adaptive RAG with Routing**

Upload a PDF file and ask a question based on its content.

2410.23396v1.pdf 1.3 MB ↴

Enter your question  
How does the HGRL framework leverage graph structures in managing multi-agent systems?

output

Please provide a concise answer to the question below based on the following documents:

Question: How does the HGRL framework leverage graph structures in managing multi-agent systems?

Answer: The HRL Framework is designed for use with multiple agents. It provides an easy way of using data from different sources, such as databases or web services (e., HTTP/2). In addition it allows you access and control over your own schema without having any third party software running at all! This means that if there are no other tools available then this approach can be used easily by anyone who wants their database management system up & working quickly while still maintaining its integrity through robustness across many layers within each agent layer .

**Figure A9. Super RAG with Dynamic Knowledge Integration**

**SuperRAG with Dynamic Knowledge Integration**

Upload a PDF file and ask a question based on its content.

2410.23393v1.pdf 1.2 MB ↴

Enter your question  
What are the main performance metrics used to evaluate the VAE-RL framework?

output

1.4/1.2

Question: What are the main performance metrics used to evaluate the VAE-RL framework?

Answer: The paper evaluates its proposed method using a combination metric that includes both exploration (the ability) as well measure how much it can improve over existing methods by comparing their performances against each other under various conditions such like when there is no censorship or if they have been censored before being applied again later down stream during training time etc.. Baxter's theorem#Baxter's formulae#  
##Introduction

## Appendix B – Project Data Source and Management Store

### Appendix B - Drive Link

**Figure B1. Data Source**

Team 8 - 298B > Appendix B – Project Da... ▾			
Type ▾	People ▾	Modified ▾	
Name	Last modified	File size	⋮
Train Dataset	9:05 PM me	—	⋮
Test Dataset	9:05 PM me	—	⋮
Main_Vector_DB	Sep 23, 2024 me	—	⋮

**Figure B2. Train Dataset**

Team 8 - 298B > Appendix B – Project D... > Train Dataset ▾			
Type ▾	People ▾	Modified ▾	
Name	Last modified	File size	⋮
source_metadata	May 13, 2024 me	—	⋮
extracted_JMLR_papers	Sep 7, 2024 me	—	⋮
extracted_arxiv_papers	May 13, 2024 me	—	⋮

... > extracted_JMLR_papers > downloaded_pdfs ▾			
Type ▾	People ▾	Modified ▾	
Name	Last modified	File size	⋮
paper_0.pdf	Mar 4, 2024 me	593 KB	⋮
paper_1.pdf	Mar 4, 2024 me	404 KB	⋮
paper_2.pdf	Mar 4, 2024 me	487 KB	⋮
paper_3.pdf	Mar 4, 2024 me	1.4 MB	⋮
paper_4.pdf	Mar 4, 2024 me	570 KB	⋮
paper_5.pdf	Mar 4, 2024 me	1.5 MB	⋮
paper_6.pdf	Mar 4, 2024 me	1.3 MB	⋮
paper_7.pdf	Mar 4, 2024 me	1.1 MB	⋮
paper_8.pdf	Mar 4, 2024 me	1.9 MB	⋮
paper_9.pdf	Mar 4, 2024 me	994 KB	⋮

**Figure B3. Test Dataset**

Name	Last modified	File size	⋮
Evaluation PDFs	8:22 PM Shruti Badrinaraya...	—	⋮
CSVs Prepared for Evaluation on Models	8:24 PM Shruti Badrinaraya...	—	⋮

**Figure B4. Primary Vector Database Corpus**

Name	Last modified	File size	⋮
merged_text_chunks.pkl	Sep 18, 2024 me	179.8 MB	⋮
merged_text_chunks_ml.pkl	Sep 25, 2024 me	907 MB	⋮
merged_text_chunks_ai.pkl	Sep 19, 2024 me	506 MB	⋮
merged_embeddings.pkl	Sep 18, 2024 me	585.5 MB	⋮
merged_embeddings_ml.pkl	Sep 25, 2024 me	2.88 GB	⋮
merged_embeddings_ai.pkl	Sep 19, 2024 me	1.61 GB	⋮
extract_DL_papers_part3.ipynb	Sep 11, 2024 me	109 KB	⋮
extract_DL_papers_part2.ipynb	Sep 10, 2024 me	359 KB	⋮
extract_DL_papers_part1.ipynb	Sep 10, 2024 me	132 KB	⋮
embeddings_usage_example.ipynb	Sep 23, 2024 Shruti Badri...	146 KB	⋮
Build_Vector_DB.ipynb	Sep 16, 2024 me	375 KB	⋮

**Figure B5. Data Management with Google Cloud Storage (GCS)**

Name	Type	Created	Storage class	Last modified	Public access	Version history	Encryption
paper_0.pdf	application/pdf	Mar 9, 2024, 7:23:52 PM	Standard	Mar 9, 2024, 7:23:52 PM	Not public	—	Google-n
paper_1.pdf	application/pdf	Mar 9, 2024, 7:23:45 PM	Standard	Mar 9, 2024, 7:23:45 PM	Not public	—	Google-n
paper_10.pdf	application/pdf	Mar 9, 2024, 7:20:54 PM	Standard	Mar 9, 2024, 7:20:54 PM	Not public	—	Google-n
paper_100.pdf	application/pdf	Mar 9, 2024, 7:20:54 PM	Standard	Mar 9, 2024, 7:20:54 PM	Not public	—	Google-n
paper_1000.pdf	application/pdf	Mar 9, 2024, 7:21:01 PM	Standard	Mar 9, 2024, 7:21:01 PM	Not public	—	Google-n
paper_1001.pdf	application/pdf	Mar 9, 2024, 7:21:05 PM	Standard	Mar 9, 2024, 7:21:05 PM	Not public	—	Google-n
paper_1002.pdf	application/pdf	Mar 9, 2024, 7:20:44 PM	Standard	Mar 9, 2024, 7:20:44 PM	Not public	—	Google-n

## Appendix C – Project Program Source Library, Presentation, and Demonstration

### Appendix C - Drive Link

**Figure C1. Project Repository**

Team 8 - 298B > Appendix C – Project Pr...			
Type	People	Modified	
Name	↑	Last modified	File size
Code		8:08 PM Shruti Badrinaraya...	—
Demo Videos		8:20 PM Shruti Badrinaraya...	—
Presentations		8:08 PM Shruti Badrinaraya...	—
Workbooks & Reports		8:07 PM Shruti Badrinaraya...	—

... > Code > Models			
Type	People	Modified	
Name	↑	Last modified	File size
M1: Ensemble Retriever RAG (Baseline)		8:41 PM Shruti Badrinaraya...	—
M2		8:09 PM Shruti Badrinaraya...	—
M3		8:09 PM Shruti Badrinaraya...	—
M4		8:10 PM Shruti Badrinaraya...	—
M5: RAG Agent with Llama3 (combined rag approaches)		8:10 PM Shruti Badrinaraya...	—
Search Engine		8:36 PM me	—

**Figure C2. Presentation**

Team 8 - Appendix C - ... > Appendix C – Project P...			
Type	People	Modified	
Name	↑	Last modified	File size
Team8 - Demo1 - PPT.pdf		10:08 PM me	6 MB
Team8 - Demo2 - PPT.pdf		9:40 PM me	6.7 MB

**Figure C3. Demonstration Videos**

