

COMP4801 Final Year Project



The University of Hong Kong

Department of Computer Science

Final Report

Generative AI for Researching Archaeology

Submitted on: 21st April 2025

Supervisor: Dr. Schnieders, Dirk

Submitted By: Lam Jiho (3035824273)

Abstract

The increasing integration of Artificial Intelligence (AI) in academic research presents exciting opportunities for archaeology, a field burdened by the challenges of processing vast amounts of specialized literature. In this project, we explored the development of a Generative AI system specifically designed to address the unique needs of archaeological research. While current AI models demonstrate strong general language capabilities, they face significant limitations in domain-specific interpretation, context window constraints, and access to paywalled academic resources.

To overcome these barriers, we implemented a Retrieval-Augmented Generation (RAG) pipeline enhanced by fine-tuning techniques and structured document processing. Our system employs CERMINE for intelligent PDF-to-XML conversion, LangChain for semantic text chunking and embedding, and ChromaDB for efficient vector-based retrieval. This architecture enables archaeologists to query complex research questions while maintaining accuracy through source-grounded responses.

The resulting tool significantly improves research efficiency by automating the literature review process, enabling rapid synthesis of archaeological knowledge, and providing context-aware analysis of specialized texts. Beyond its practical applications, this project serves as a case study in adapting cutting-edge AI technologies to humanities research, demonstrating how customized implementations can bridge the gap between general-purpose AI and domain-specific research needs. Our work aims to not only advance archaeological methodology but also contribute towards broader discussions about AI's role in transforming academic research practices across disciplines.

Acknowledgements

I would first like to express our gratitude to Dr. Peter Cobb for his guidance in initiating this interdisciplinary research direction, and his expertise invaluable throughout this project.

I am also grateful to our supervisor Dr. Dirk Schnieders for his mentorship throughout the project's development. His feedback and thoughtful advice were instrumental in shaping our research goals, methodology, and implementation.

I extend my thanks to my teammate, Fan Yian, whose dedicated participation and seamless cooperation made this project possible.

Table of Contents:

1. Introduction.....	6
1.1 Project Background.....	6
1.2 Current Situation and Limitations.....	7
1.3 Project Objective.....	9
2. Methodology.....	10
2.1 Overview.....	10
2.2 System Architecture.....	10
2.2.1 Document Conversion & Structuring.....	11
2.2.2 Text Processing & Embedding.....	11
2.2.3 Retrieval-Augmented Generation.....	12
2.2.4 Response Generation & User Interface.....	13
2.3 Retrieval-Augmented Generation (RAG) Framework.....	13
2.4 Document Processing & Embedding.....	16
2.4.1 Conversion.....	16
2.4.2 Splitting.....	19
2.4.3 Embedding.....	19
2.5 Model Selection.....	20
2.6 Prompt Engineering and Optimization.....	21
2.7 Infrastructure and Deployment.....	22
3. Implementations and Results.....	24
3.1 Outline.....	24

3.2 Model Selection Finalisation.....	24
3.3 Document Conversion Efficiency.....	25
3.3.1 CERMINE's Strengths.....	26
3.3.2 Limitations and Cleanup Challenges.....	26
3.4 Performance Evaluation.....	28
3.4.1 Accuracy.....	31
3.4.2 Precision.....	32
3.4.2 Recall.....	33
3.4.3 F1-Score.....	33
3.4.4 Qualitative Analysis.....	34
3.4.5 Comparison to Baseline GPT-4.....	36
3.5 User Interface.....	37
3.5.1 Flask-Based Local Web UI.....	37
4. Future Works.....	39
4.1 Web Interface and Scalable Deployment.....	39
4.2 Collaboration with Archaeologists.....	39
4.3 Enhancing Prompt Engineering.....	40
4.4 Expanding the Dataset.....	40
4.5 Improving Document Conversion and Cleanup.....	41
5. Conclusion.....	42
Bibliography.....	43

List of Figures/Tables

Figure 2.2.1 System Architecture of the Project.....	11
Figure 2.3.1 Ingestion and Retrieval with Embeddings [13].....	14
Figure 2.3.2 Overview of LangChain’s Workflow [15].....	15
Figure 2.4.1.1 CERMINE Workflow [17].....	17
Figure 2.4.1.2 Example of Failed XML Conversion.....	18
Figure 2.4.1. Splitting and Embed Workflow [18].....	19
Figure 2.4.3.1 Vector Database Workflow [23].....	20
Figure 2.7.1 Screenshot of our Gradio Implementation.....	23
Figure 3.3.1 Screenshot of Converted XML file with Metadata.....	26
Figure 3.4.1 Visualization of Precision and Recall [32].....	28
Figure 3.4.2 Table of Answerable Questions used for Testing.....	30
Table 3.4.3 Table of Unanswerable Questions used for Testing.....	30
Table 3.4.4 Classification Breakdown of Testing Results.....	31
3.4.4.1 Table with Long Answer Question Example.....	35
Figure 3.4.5.1 Incorrect Output by Baseline GPT-4.....	36
Figure 3.4.5.2 Correct output with our RAG-enabled system.....	36
Figure 3.5.1.1 Screenshot of Web Interface.....	38
Figure 3.5.1.2 Screenshot of Application in Use.....	38

1. Introduction

1.1 Project Background

The application of Artificial Intelligence (AI), particularly machine learning, has seen substantial growth in fields spanning all forms of research, with researchers in the field of archaeology increasingly leveraging these technologies for tasks such as artifact classification, site prediction, and complex data analysis. For instance, Labba et al. (2023) showcase AI tools designed to empower archaeologists with data analytics capabilities without requiring programming expertise, highlighting AI's expanding accessibility and utility within archaeological research workflows [1]. Similarly, other studies have highlighted the role of machine learning in enhancing site detection and artifact interpretation, as demonstrated by the Saruq Al Hadid project, which utilized AI to predict the locations of previously undiscovered structures [2].

Despite these advancements, archaeologists continue to face substantial challenges in managing the ever-expanding body of academic literature. AI-assisted systems, such as the AGNES project, have begun to address this issue by employing Named Entity Recognition (NER) to index extensive archaeological texts, enabling more accurate and context-aware search capabilities. In specific case studies, AGNES uncovered up to 30% more relevant findings than traditional methods, highlighting the inefficiencies in current literature review practices. Research suggests that archaeologists may spend a significant amount of their time manually reviewing literature, a process made increasingly difficult by the specialized vocabulary and interdisciplinary nature of the field [3].

This growing adoption of AI in humanities research is well-documented, with EU reports emphasizing its transformative potential for archaeology and historical studies, particularly in site localization and enriched data interpretation [4]. However, the use of AI for archaeological knowledge management remains relatively underdeveloped and underutilized [19], particularly in the context of interpreting and synthesizing large bodies of academic literature, highlighting the need for domain-specific AI systems.

1.2 Current Situation and Limitations

While AI has found success in various archaeological applications, such as artifact restoration, predictive modeling, and historical text translation [5][6][31], the use of Generative AI specifically tailored for archaeological research remains underexplored. This presents a clear opportunity for the development in how archaeologists interact with, synthesize, and extract insights from academic literature and data.

Despite the performance of current AI models like GPT-4o, these systems are primarily trained on broad, general-purpose datasets and are not fine-tuned for the nuances of archaeological discourse [3]. As a result, they often struggle with accurately retrieving and interpreting domain-specific information, such as specialized terminology, cultural context, or references to historical datasets and recent findings. This lack of specialization presents a challenge when using these models to support archaeological research.

Furthermore, limited access to key academic resources presents a significant challenge. Many important archaeological publications remain behind paywalls or are difficult to find and

obtain, restricting AI developers' ability to train models that engage with the full scope of current research literature [7][8]. Even when access is possible, technical limitations such as the context window constraint, such as with GPT-4o being capped at 128K tokens, restrict the model's capacity to handle multiple full-length papers simultaneously [9]. This presents a challenge for users and researchers who wish to incorporate various research texts into a single query to perform more comprehensive and context-rich output generation, as currently available Generative AI models struggle or are not able to perform comprehensive cross-referencing or synthesize information from large volumes of input data.

Thus, current limitations in AI architecture present significant challenges for archaeological research applications. Without integrated retrieval capabilities, these models struggle to maintain consistent contextual understanding across lengthy academic and research documents, or when requiring unavailable source materials. These constraints reduce the effectiveness of Generative AI for various research workflows, as they cannot reliably track arguments through extended texts, or effectively incorporate fragmented research sources, or adequately support complex analyses requiring the synthesis of multiple specialized works. These limitations are significant for archaeological studies, where examination of various interconnected sources and domain-specific knowledge is required.

1.3 Project Objective

Working with Dr. Peter Cobb, this project aims to bridge the gap between AI and Archaeology by developing a domain-adapted Generative AI system tailored specifically for archaeological research. While existing AI models demonstrate strong capabilities in general natural language processing, they fall short when applied to domain-specific contexts like archaeology, struggling with limited literature access, and the ability to synthesize insights across multiple sources. This project seeks to address these challenges by creating a research assistant tool that leverages Generative AI to support archaeologists in navigating and interpreting large volumes of academic content.

The system is designed to process PDF texts directly provided by researchers and users, enabling customized input and facilitating more targeted literature exploration. By incorporating techniques such as Retrieval-Augmented Generation (RAG), semantic chunking, and document-level structuring, the tool will provide context-aware, accurate, and traceable outputs grounded in real and customizable archaeological sources. This not only reduces the time spent on manual literature review but also enhances researchers' ability to uncover connections and insights across documents.

Motivated by the rapid advancement of AI technologies, our team also views this project as an opportunity to gain hands-on experience in developing applications with large language models. The process of tailoring AI capabilities to a specialized domain like archaeology allows us to explore the full potential of current generative technologies while contributing to the future of digital research tools in the field of archaeology.

2. Methodology

2.1 Overview

This section outlines the technical foundation and implementation strategy behind the development of our Generative AI system. It provides an explanation of the key technologies, model architecture, and design decisions that support the project's objectives. Specifically, it will detail the tools and frameworks used, such as Retrieval-Augmented Generation (RAG) as a core methodology, alongside advanced techniques such as fine-tuning and prompt engineering, leveraging technologies such as LangChain, CERMINE, ChromaDB, and various AI models such as OpenAI's GPT, along with the underlying algorithms and infrastructure setup that enable document processing, embedding, and retrieval.

2.2 System Architecture

This section will provide a brief explanation on the overall system architecture of our implementation, where the system is built around a modular, scalable pipeline specifically designed to process archaeological research papers and support accurate, context-aware AI-assisted research. The architecture is composed of four primary stages, each optimized to handle domain-specific challenges in working with complex academic literature. The pipeline enables seamless integration between document input, semantic indexing, and AI response generation, providing a foundation for interactive archaeological research assistance.

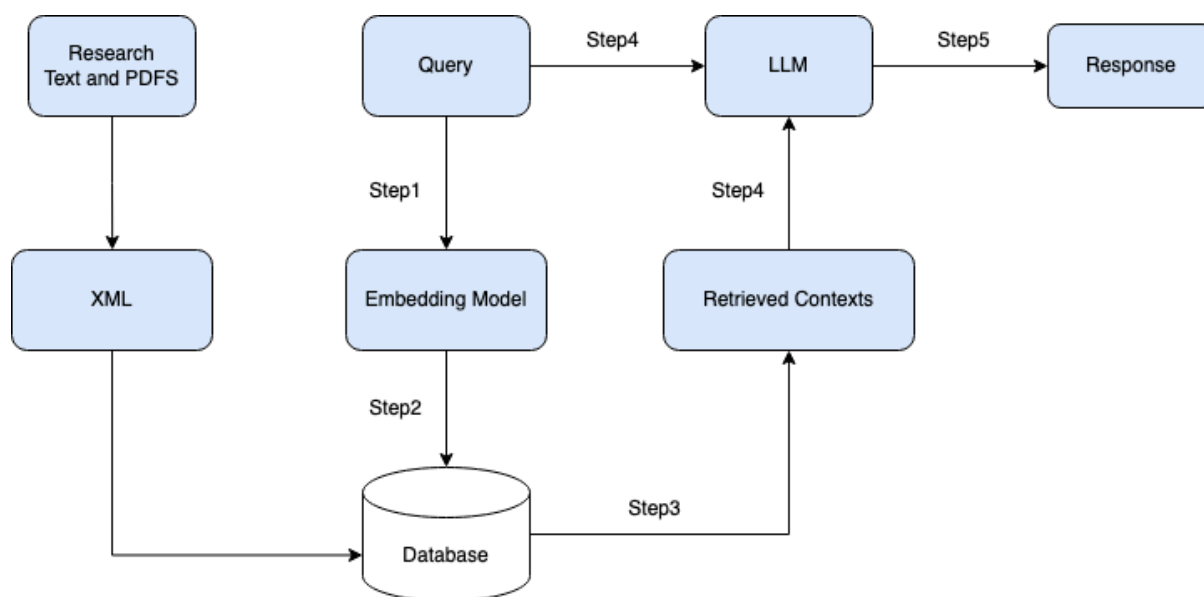


Figure 2.2.1 System Architecture of the Project

2.2.1 Document Conversion & Structuring

The workflow begins with converting raw research papers, in PDF format, into structured XML documents using CERMINE [10], a machine learning-based content extraction tool. CERMINE is chosen for its ability to retain rich metadata (e.g., section headings, references, author info), which is often lost in plain text extraction. The resulting XML files maintain a semantic structure, allowing for more organized processing and interpretation of different text types (e.g., body paragraphs vs. footnotes or references). This structured representation significantly improves the precision and consistency of downstream processing.

2.2.2 Text Processing & Embedding

Once the XML files are parsed, the text is processed and segmented into overlapping chunks to preserve contextual continuity. Currently, chunks are split into windows of 3800 characters with a 500-character overlap using LangChain's text splitter tools, though the exact

parameters (window size, overlap, and splitting method) can be adjusted based on the specific requirements of the task, the model's context window limitations, or downstream processing needs.

These chunks are then embedded into high-dimensional vector representations using embedding models, which capture the semantic meaning of the text, enabling similarity-based retrieval. Both the embeddings and their corresponding text chunks are stored in ChromaDB, a fast and scalable vector database. ChromaDB is optimized for similarity search, allowing the system to efficiently identify the most relevant information in response to user queries.

2.2.3 Retrieval-Augmented Generation

When a user submits a query, the system generates an embedding of the query using the same embedding model. This embedding is compared against those stored in ChromaDB using vector similarity metrics (e.g., cosine similarity) to retrieve the most relevant text chunks from the indexed literature.

This Retrieval-Augmented Generation (RAG) approach supplements the generative model's capabilities with real-time access to domain-specific information. Rather than relying solely on the language model's internal training data, the system retrieves and incorporates external evidence to generate accurate, relevant, and up-to-date responses. This drastically reduces the risk of hallucinations and improves the quality of domain-specific answers, which is particularly important in archaeology, where precision and source-based interpretation are critical.

2.2.4 Response Generation & User Interface

The final stage of the pipeline involves integrating the retrieved text chunks into a prompt template using LangChain. This templated prompt is then passed to an AI model, which generates a context-aware response informed by the retrieved literature. The output is displayed through a web-based user interface, allowing for quick testing, interaction, and demonstration of the system's functionality.

This interface enables users, archaeologists, researchers, or students to interact with the AI system, which responds with detailed information drawn from its research database.

2.3 Retrieval-Augmented Generation (RAG) Framework

The Retrieval-Augmented Generation (RAG) framework is central to the system's ability to produce accurate and contextually grounded responses based on archaeological literature.

Unlike traditional generative models that rely solely on pre-trained internal knowledge, RAG enhances the output quality by incorporating real-time retrieval of relevant external documents [11]. This hybrid approach ensures that the model is not only generating correct responses but also ensuring that they are backed by verifiable academic content, thereby significantly reducing hallucinations and enhancing domain specificity [12].

In this system, when a user submits a query, it is first embedded into a dense vector using the same embedding model that was used during the document ingestion phase. This query vector is then used to search ChromaDB, a vector database optimized for sourcing semantic similarity, thus is able to identify the most relevant chunks of various archaeological texts.

These chunks are selected based on vector similarity, meaning the system retrieves passages that are conceptually aligned with the query, even if the vocabulary or phrasing differs.

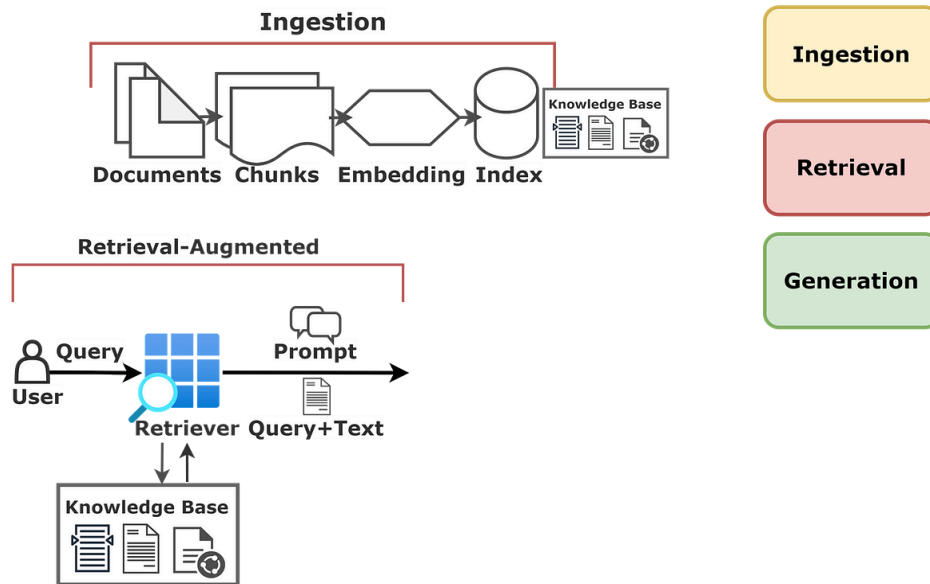


Figure 2.3.1 Ingestion and Retrieval with Embeddings [13]

The retrieved text segments are then passed into LangChain, a powerful open-source framework designed for building language model-powered applications that interact with external data [14]. LangChain serves as the orchestration layer in this pipeline, dynamically formatting and constructing the input prompt for the large language model. It integrates the retrieved chunks alongside the user's query into a structured, context-rich prompt, ensuring the model has access to the most relevant background information when generating a response [15].

LangChain's modular design also enables flexible experimentation with different retrieval strategies, prompt templates, and memory mechanisms, allowing our team to iterate and refine the RAG system efficiently [16]. In this project, LangChain played a key role in

maintaining the coherence and traceability of the AI's responses by tightly coupling the retrieval process with generation, which was essential in achieving our group's goals.

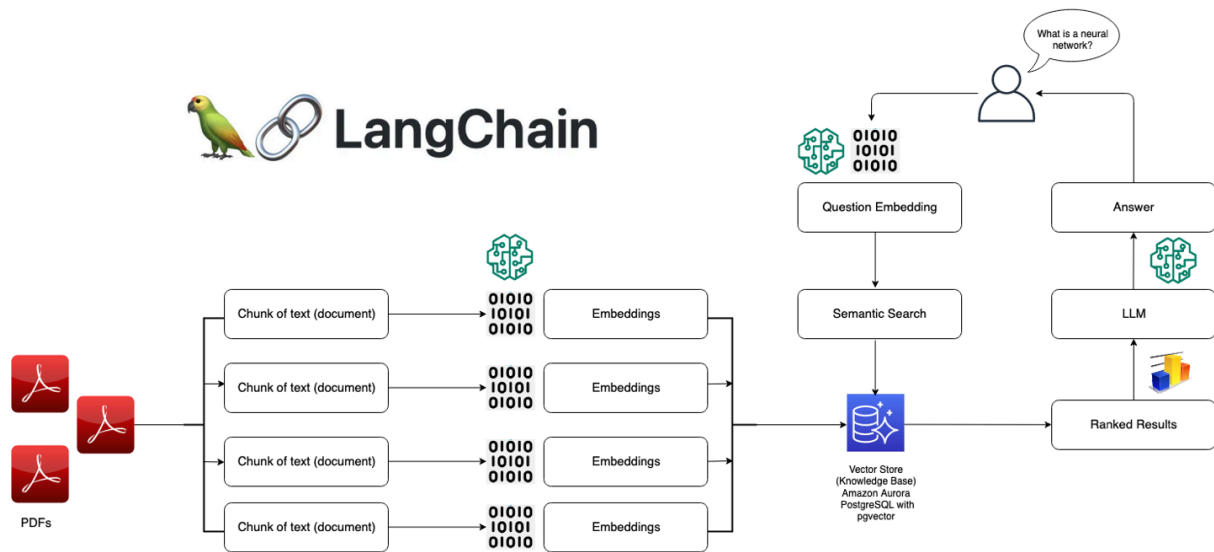


Figure 2.3.2 Overview of LangChain's Workflow [15]

Through the integration of the RAG framework, we were able to make significant progress in bridging the gap between general AI capabilities and the highly specialized needs of archaeological research. This allows the system to generate responses that are not only linguistically correct but also firmly rooted in domain-relevant and up-to-date literature.

2.4 Document Processing & Embedding

The document processing and embedding stage is an essential step for enabling accurate semantic search and context-aware generation in our system. This phase transforms raw archaeological research papers into a structured and searchable vector format suitable for downstream AI tasks. It comprises three main steps: conversion, splitting, and embedding.

2.4.1 Conversion

Before settling on our current pipeline, we experimented with several approaches to convert archaeological research PDFs into a format suitable for large language models and Retrieval-Augmented Generation. Initially, we tested a range of OCR (Optical Character Recognition) tools, including commercial solutions like Adobe Acrobat's built-in OCR engine. While these tools were capable of extracting text, they presented several limitations: the process was slow, often inconsistent across documents with complex formatting or diagrams, non-automatable for large batches, and costly when scaling beyond a handful of files. These drawbacks made them unsuitable for a seamless, large-scale pipeline intended to support efficient querying and knowledge extraction.

To address these issues, the pipeline now starts with the conversion of academic PDF documents into structured XML using CERMINE, a tool specialized in extracting metadata and content from scientific literature. CERMINE preserves essential structural components such as section headers, figure references, bibliographies, and citation markers, which are critical for parsing scholarly texts in a meaningful way [10].

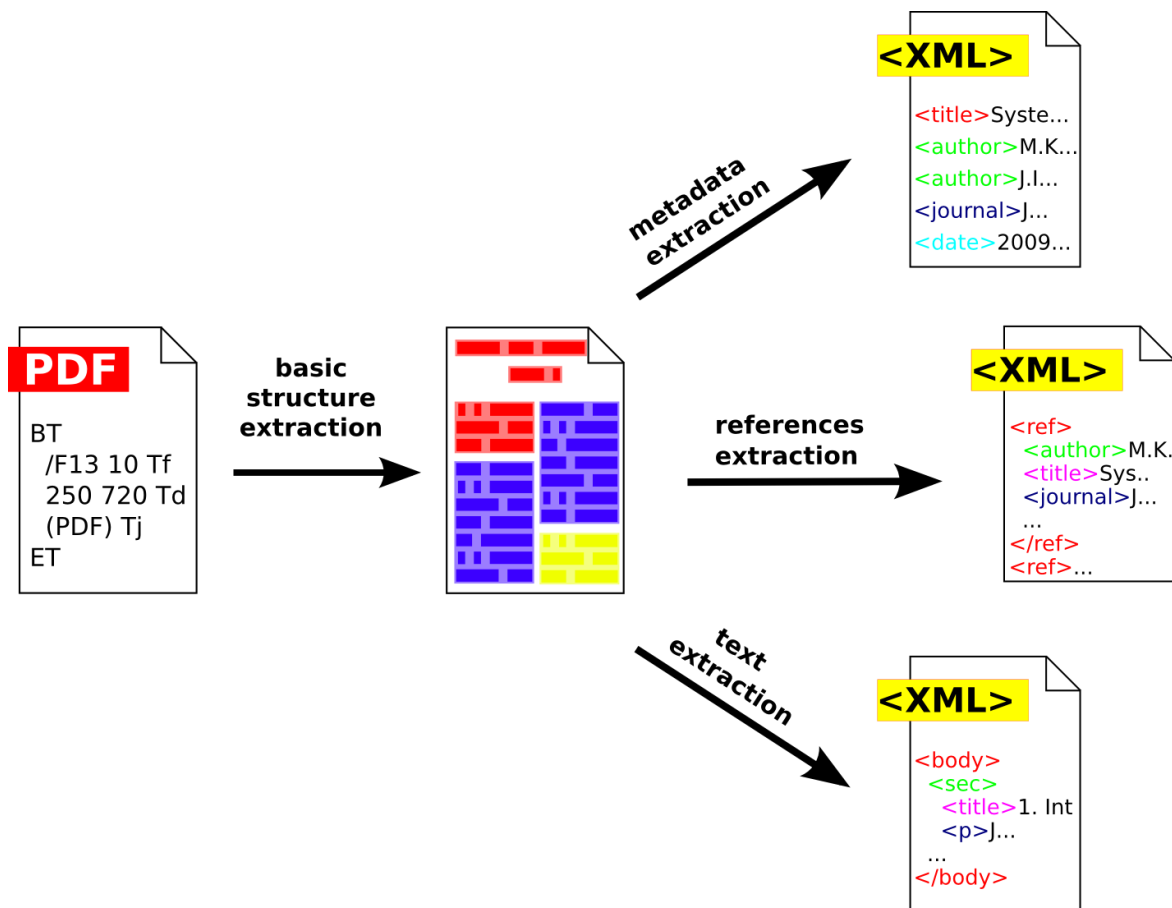


Figure 2.4.1.1 CERMINE Workflow [17]

However, in practice, CERMINE does not always produce clean or usable XML outputs. Some common issues include garbled characters (e.g., sequences of ???), or structural inconsistencies, which was often the result from scanning PDFs which contained language characters which were not supported, or complex formatting, or degraded source quality.

```

| | <p>????????
?. ?. ?. ?. ?. ?????? ??????</p>
| | <p>«????????»
?????????? ?? ?????????? ??????</p>
| | <p>????????
??????? ?????? ?????????????
?. ?????? ?????? ??????</p>
| | <p>????????
?????? ?????????, ?????? ??????</p>
| | <p>????????? ??????
?. ?????? ??, ?????? ??????</p>
| | <p>????????? ??????
?????? ?????????
????????? ???. ??????
?????? ??????
???
????????? ?????? ?????????? ?????????? ?????????????????

```

Figure 2.4.1.2 Example of Failed XML Conversion

To address this, a custom cleanup program was developed to automatically post-process the raw XML output. This tool detects and removes corrupted content, normalizes various structures, thus ensuring consistency in structural tags. It is fully integrated into the document conversion workflow so that each uploaded PDF passes through CERMINE and then through the cleanup module before any downstream processing. This not only increases reliability but also ensures that only high-quality, parseable content enters the semantic pipeline.

To enhance user accessibility, this entire process, from PDF upload to cleaned XML output, is embedded into the system's web interface, enabling seamless document conversion with minimal user intervention.

2.4.2 Splitting

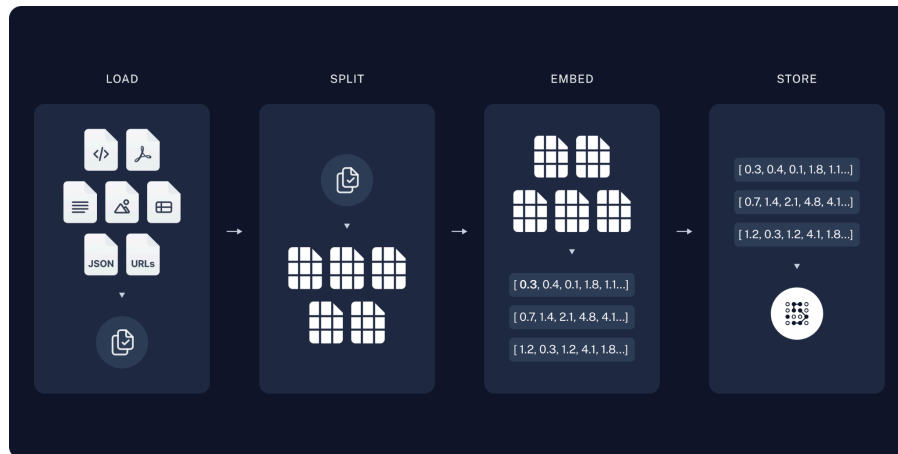


Figure 2.4.1. Splitting and Embed Workflow [18]

Once converted, the cleaned text is processed using LangChain’s text splitter, which divides it into smaller, manageable chunks for embedding [18]. In this implementation, we use 3800-character chunks with a 500-character overlap. The overlap ensures continuity between chunks, preserving semantic context across boundaries, especially important in dense academic writing where key information may span multiple paragraphs. However, these parameters are adjustable and can be fine-tuned based on the specific use case, the model’s context window limitations, or the nature of the source material. LangChain’s flexible splitting methods (by character, sentence, or token) further allow us to adapt the chunking strategy to different document types or downstream tasks.

2.4.3 Embedding

Each chunk is then transformed into a high-dimensional vector using OpenAI’s embedding model [21], which encodes the semantic meaning of the text into a numerical representation. These vectors are stored in ChromaDB, a vector database [20], which leverages SQL-like querying for efficient retrieval and management of embeddings. Unlike traditional SQL

databases that rely on exact keyword matching or structured joins, vector databases are able to use approximate nearest neighbor (ANN) searches to compare the semantic meaning of queries and documents through mathematical proximity.

By storing embeddings in a structured yet flexible format such as this, ChromaDB enables SQL-compatible operations (such as filtering and indexing) while also supporting vector similarity searches for high performance. This hybrid approach allows the system to retrieve the most relevant chunks based on conceptual similarity rather than exact wording, thus combining the strengths of relational database management with advanced AI-driven search capabilities.

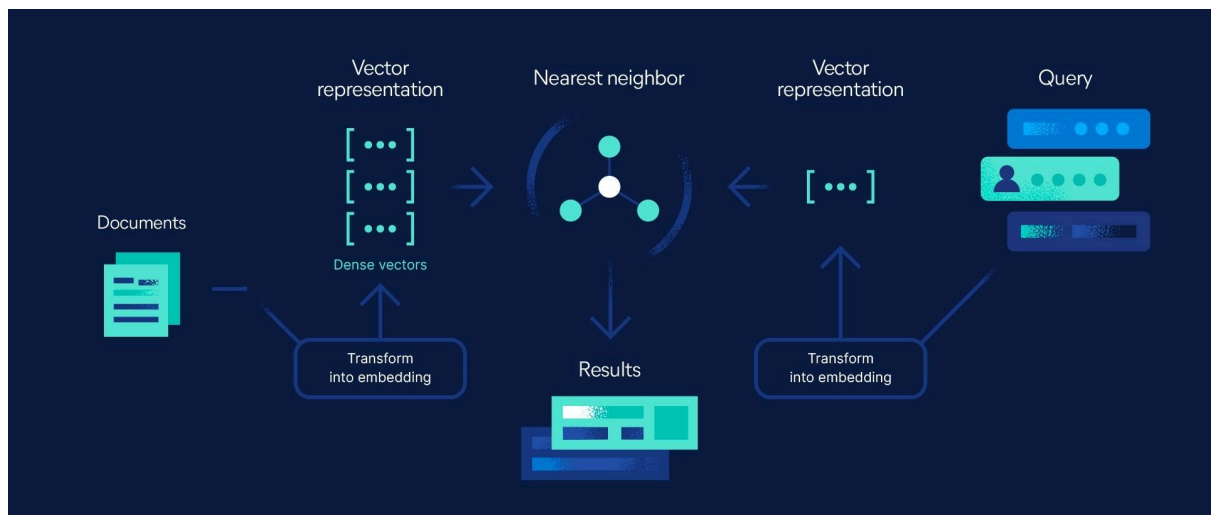


Figure 2.4.3.1 Vector Database Workflow [23]

2.5 Model Selection

While the Retrieval-Augmented Generation (RAG) framework significantly reduces the reliance on extensive model pretraining, there remains a need for targeted model adaptation to ensure outputs are accurate, contextually appropriate, and aligned with our project's aim.

As part of this exploration, we experimented with a range of models to determine their suitability for this use case. Notably, SciBERT, a BERT model trained on scientific publications [22], offered strong performance in parsing technical language and was particularly effective in understanding structured academic texts. Being freely available and open-source made SciBERT an accessible and cost-efficient option for experimentation.

We also tested Google's Gemini [24], which showed promising results in reasoning and general responsiveness across a variety of prompts. Like SciBERT, it was also freely accessible for development purposes. In contrast, GPT-4 (via OpenAI's API) required payment based on usage, introducing cost as a factor in sustained development. Despite this, GPT-4 delivered high-quality generation and strong performance across diverse prompts, making it a useful baseline for benchmarking.

2.6 Prompt Engineering and Optimization

Prompt engineering refers to the process of designing and refining the inputs (or prompts) given to a language model in order to guide it toward producing accurate, relevant, and useful responses [26]. Since large language models like GPT-4o generate outputs based entirely on the input they receive, the way a prompt is phrased, its structure, tone, and clarity, can significantly influence the model's performance [25]. In domain-specific contexts like archaeology, prompt engineering is especially important to ensure that the AI produces responses that are not only factually correct but also aligned with scholarly conventions and expectations.

In our system, LangChain plays a central role as it supports template-based prompting, enabling consistent formatting across diverse user queries. Simple queries like definitions

yield short, direct prompts, while more complex questions trigger more elaborate templates designed to encourage thoughtful, multi-layered responses.

Additionally, we introduced logic for dynamic prompt adaptation—automatically adjusting the level of detail, tone, and citation structure based on the nature of the query. These optimizations were essential to improving both the quality and usability of responses, especially for archaeologists seeking in-depth answers grounded in specific excavation reports, journal articles, or technical documentation.

Early in development, we found that standard prompts often led to overly generic or brief answers. To address this, we fine-tuned our templates to request longer, more detailed outputs, and made sure that each piece of generated content clearly cited the source document from which the information was retrieved, improving the academic reliability of the system.

2.7 Infrastructure and Deployment

The system is built on a modular and scalable backend infrastructure, primarily developed in Python, which integrates several key components: LangChain for prompt orchestration, ChromaDB as the vector database for semantic retrieval, and OpenAI’s API for access to advanced language models such as GPT-4o. This backend pipeline enables seamless interaction between document processing, vector search, and generative response generation, ensuring fast and accurate performance.

For the user interface (UI), we adopted a two-pronged approach to support both development testing and potential deployment:

- Flask was used to build a lightweight, local web UI. This interface allows researchers and developers to upload their own PDF files, submit queries, and view results through a clean, interactive dashboard. Flask’s simplicity and flexibility made it an ideal choice for prototyping and internal use.
- In parallel, we integrated Gradio [27], a Python library that enables the creation of browser-based UIs with minimal overhead. Gradio was particularly useful for temporary server hosting, live demonstrations, and remote user testing, allowing us to share our system easily with collaborators without requiring full deployment.

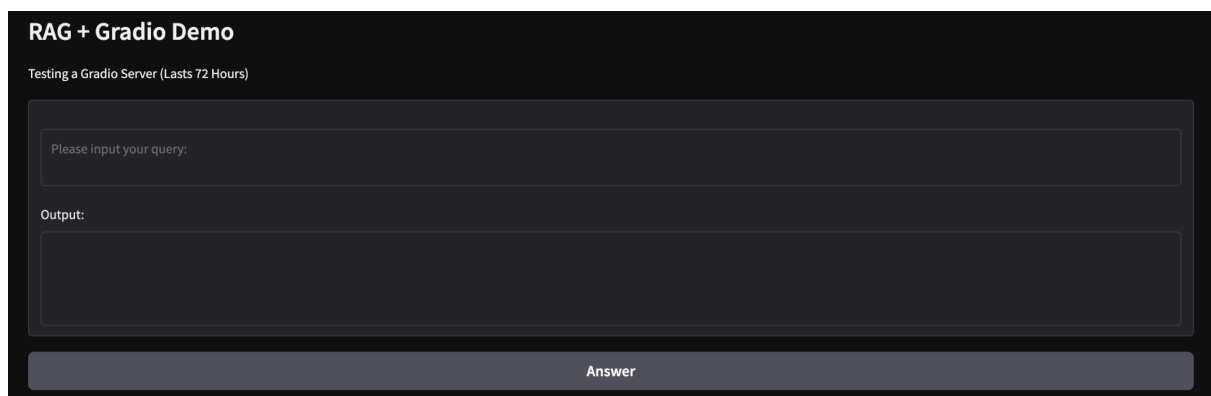


Figure 2.7.1 Screenshot of our Gradio Implementation

This dual-UI setup provided flexibility throughout development, where Flask acts as a more controlled, customizable implementation, while Gradio excels in the quick iterations and accessibility.

3. Implementations and Results

3.1 Outline

This chapter outlines our practical implementation of our Generative AI system and highlights key outcomes and results derived from its deployment.

3.2 Model Selection Finalisation

The selection of the appropriate language model was a critical step in the development of our system, as it directly impacts the quality, relevance, and flexibility of the generated responses. To make an informed choice, we conducted comparative testing across several advanced models, including Google's Gemini, SciBERT, and OpenAI's GPT-4, among others. Each model was evaluated based on key criteria such as accuracy on domain-specific tasks, ease of integration with our pipeline, cost, and performance on a small test dataset curated from archaeological texts and queries.

SciBERT, a transformer model pre-trained on scientific literature, showed promise in its ability to understand academic language and terminology. It performed reasonably well on certain retrieval tasks, particularly in extracting factual content from scholarly sources. Additionally, it was completely open-source and free, making it an attractive option for cost-sensitive use cases. However, its lack of native integration with modern frameworks like LangChain, as well as its limited generative capabilities, made it less suitable for our project's goals [22].

Google Gemini, another freely available model, offered impressive generative output but presented difficulties in integration due to API restrictions and limited documentation for

customization. While it produced initial satisfactory responses, it also lacked the depth and flexibility needed to support our full RAG pipeline without significant workarounds.

Ultimately, we selected OpenAI's GPT-4 as the core model for our system. Despite the associated cost, GPT-4 consistently outperformed the others in terms of output quality, robustness, and reliability. Its compatibility with the LangChain framework, seamless API integration, and capacity to handle complex prompts made it the most practical choice for a project requiring nuanced and context-rich responses. Furthermore, GPT-4 demonstrated strong performance on domain-specific tasks such as interpreting archaeological terminology, generating structured summaries, and maintaining contextual consistency across long-form queries.

While other models showed potential and remain viable options for future iterations or budget-conscious applications, GPT-4 provided the best balance of accuracy, usability, and ecosystem support, making it the ideal choice for the implementation of our project.

3.3 Document Conversion Efficiency

The first stage of our implemented pipeline, document conversion, focused on transforming unstructured archaeological PDFs into structured XML files using the aforementioned CERMINE, thus transforming texts to be more suitable for downstream processing.

3.3.1 CERMINE's Strengths

CERMINE demonstrated strong capabilities in handling a wide range of academic document layouts, from highly formatted journal articles to less structured excavation reports and textbooks. It effectively preserved critical metadata such as:

- Section headers and titles
- Bibliographic references and citation markers
- Author and publication metadata

```
<title-group>
  <article-title>University of Michigan Press University of Michigan Museum of Anthropological Archaeology Chapter Title: EVIDENCE OF THE PARTHIAN AND SASANIAN
  PERIODS</article-title>
</title-group>
<contrib-group>
  <contrib contrib-type="author">
    <string-name>: HENRY T. WRIGHT</string-name>
    <xref ref-type="aff" rid="aff0">0</xref>
  </contrib>
  <aff id="aff0">
    <label>0</label>
    <institution>Published by: University of Michigan Press, University of Michigan Museum of Anthropological</institution>
    <addr-line>Archaeology. (1981) Stable URL: https://</addr-line>
    <country country="US">USA</country>
  </aff>
</contrib-group>
```

Figure 3.3.1 Screenshot of Converted XML file with Metadata

This structured output greatly improved our ability to parse and segment text in a meaningful way, enabling more precise chunking and retrieval in later stages. Thus, CERMINE was able to retain a large portion of key metadata elements throughout the conversion process.

3.3.2 Limitations and Cleanup Challenges

Despite these strengths, 34 out of 219 processed documents (roughly 15.5%) required cleanup due to incomplete or flawed XML output. Common issues included:

- Garbled text and replacement characters (e.g., “???”): Often caused by embedded and unsupported fonts/characters, or scanned image-based PDFs, where character encoding was not preserved correctly.
- Non-standard formatting: Documents with multi-column layouts, footnotes, sidebars, or complex table structures frequently caused errors or omissions during extraction.

To mitigate these issues, we implemented an automated cleanup process that is integrated directly into the conversion pipeline. This algorithm scanned for and removed corrupted content and handled placeholder characters, as well as fixed and normalised various text structures caused by different PDF templates. Thus, if the PDF-to-XML conversion process outputs files containing texts that couldn't be preserved or confidently recovered, the cleanup program would detect and remove those portions. or, in some cases, discard the document entirely from the database.

This cleanup step, while essential, introduced a tradeoff, where several documents lost large portions of content, rendering them partially or entirely unusable by the AI. However, this step was required, as when working with generative AI, providing no information is significantly better than providing inaccurate or corrupted information [28]. Some reasons include the following:

- Hallucination risk: Incorrect or garbled text can lead to misleading outputs from the language model, potentially fabricating citations or drawing false conclusions [29].
- Loss of trust: In academic and research contexts, factual integrity is extremely important. A single hallucinated statement based on bad input can undermine the credibility of the entire system, and subsequent generated results.
- Retrieval quality: Faulty embeddings based on noisy or malformed text reduce the accuracy of vector-based retrieval, which degrades overall system performance.

By filtering out compromised content, we ensured the reliability of our system, thus making sure that the AI only generated responses grounded in verified, high-quality source materials. While this meant sacrificing volume, it helped maintain the intellectual rigor and domain specificity necessary for archaeological research [30].

3.4 Performance Evaluation

To assess the system's effectiveness, we implemented various evaluation metrics to objectively measure our implementation's performance. By applying precision, recall, and F1 score calculation to a custom benchmark set of 125 questions, we designed a process to evaluate both factual accuracy and the system's ability to avoid hallucinations. We detail how each metric is calculated, what it measures, and why it matters in the context of archaeological research. These metrics are complemented by comparisons to a baseline GPT-4 model without RAG, showing the tangible improvements offered by document grounding.

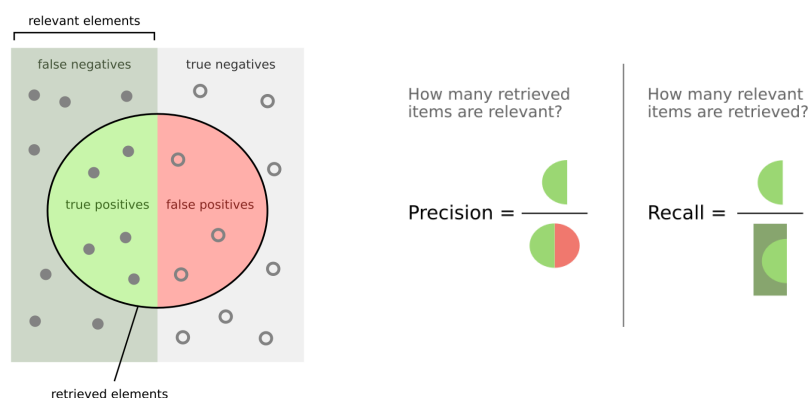


Figure 3.4.1 Visualization of Precision and Recall [32]

We developed a benchmark set of 125 questions where: 100 questions had answers which were answerable using the embedded data, and 25 questions that were unanswerable from the given materials. This was designed to test whether the AI could recognize when it should not fabricate an answer. The questions were primarily created manually, and answers were manually verified to ensure fairness and accuracy. This was essential because minor variations in phrasing could still yield correct answers, thus causing misclassification.

Some examples of the questions are as below:

Answerable Questions	Expected Answer	Source
During which historical period does the document provide evidence for the establishment of the Ultan Qalasi settlement?	The Sasanian period	Alizadeh2011-UltanQalasi.pdf
Which century is the earliest mention of the Ghilghilchay defensive long wall dated to?	7th century AD	AlievEtal2006-GhilghilchayDefensiveLongWall.pdf
Which two Sasanian rulers are traditionally credited with directing large-scale projects, including land reform and irrigation systems, according to historical accounts?	Kawad I and Husraw I Anushirwan	Alizadeh2014-BorderlandProjectsSasanian.pdf

Figure 3.4.2 Table of Answerable Questions used for Testing

These questions are derived from texts that were successfully converted into structured XML files and embedded into the vector database. Thus, the model should have access to the relevant source material, meaning that all questions in this set should be answerable based on the stored content.

Some examples of questions used to test hallucinations are as follows:

Unanswerable Questions	Source
What are the four main factors that explain the Elymaeans' rise as a major power in Khuzestan?	Alizadeh1985-ElymaeanOccupationofLowerKhuzestan.pdf
Which nomadic tribes threatened Parthia after 129 BC?	Early Parthian coins from Margiana.pdf

Table 3.4.3 Table of Unanswerable Questions used for Testing

These questions were deliberately crafted using archaeological texts that had failed XML conversion, thus they were not successfully parsed and were excluded from the embedding process. As a result, the model had no access to the information contained within those documents in the vector database. This setup ensured that any correct-looking answer would be the result of hallucination, rather than valid retrieval, allowing us to effectively evaluate the system's ability to handle unanswerable queries and to distinguish between retrieved knowledge and fabricated content.

The AI's performance on this test set was as followed:

- Correctly answered 92 out of 100 valid questions
- Incorrectly answered 6 out of 25 unanswerable ones (i.e., hallucinated 6 responses)
- Missed 8 correct questions

These observations translate to the following classification breakdown:

	Predicted Correct	Predicted Incorrect
Actually Correct	92 (True Positives)	8 (False Negatives)
Actually Incorrect	6 (False Positives)	19 (True Negatives)

Table 3.4.4 Classification Breakdown of Testing Results

3.4.1 Accuracy

During the interim reporting phase, we focused primarily on simple information extraction tasks, such as identifying named entities, dates, etc from archaeological texts. These questions were drawn from a manually curated set of 75 queries, all of which were intended to be answerable given the available input, of which, 68 correct answers were generated. This led to a raw accuracy of approximately 91%.

Using our current test case, with the curated set expanded to 100 answerable queries, the model returned 92 correct answers, thus resulting in a raw accuracy of 92%. While this marks only a 1% increase in raw accuracy, the improvement is notable given the larger and more diverse question set, which introduces greater complexity and reduces variance due to random chance. Moreover, raw accuracy alone does not account for important factors such as false positives (i.e., hallucinated answers to unanswerable questions) or false negatives (i.e., missed correct answers), which can significantly impact the system's reliability in real-world use.

These issues are addressed more thoroughly in the precision, recall, and F1 score metrics discussed below.

3.4.2 Precision

Precision measures how many of the model's seemingly correct responses were actually factually accurate. It is calculated as the number of true positives (correct answers) divided by the total number of positive responses (true positives + false positives). In our test set of 125 questions, 25 were intentionally unanswerable to simulate conditions where a well-grounded model should refuse to generate speculative or hallucinated answers. Precision is especially important in academic contexts like archaeology, where the cost of providing incorrect information can be significant.

Calculating the precision [32] using the aforementioned values:

$$Precision = \frac{TP}{TP + FP} = \frac{92}{92 + 6} \approx 93.88\%$$

Where TP (True Positive) = 92, and FP (False Positive) = 6.

A precision of 93.88% means that nearly all of the AI's responses were accurate. In archaeological applications, this high precision is crucial to avoid hallucinations, especially when users might act on these results in academic or field settings.

However, there is room for further optimization. Some false positives stemmed from hallucinations caused by incorrect or incomplete retrieval, ambiguous phrasing in prompts, or noisy input data [33]. Improving the conversion process (e.g., cleaner XML generation),

refining chunk overlap and formatting, and designing more context-sensitive prompts could help further reduce these errors and push precision even higher.

3.4.2 Recall

Recall evaluates the model’s ability to retrieve and correctly respond to all valid questions. It is calculated as the number of true positives divided by the total number of actual answerable questions (true positives + false negatives). False negatives occur when the model fails to provide a correct answer to a question it should have been able to answer, potentially due to context misalignment, misinterpretation, or insufficient retrieval coverage.

Calculating the recall [32] using the stored values:

$$Recall = \frac{TP}{TP + FN} = \frac{92}{92 + 8} = 92\%$$

Where FN (False Negatives) = 8, as (100 total valid questions – 92 correctly answered = 8)

This recall value indicates the model is consistently able to find the correct information when it exists. However, recall is sensitive to retrieval quality, and if the right chunk wasn’t retrieved (due to limitations in embedding similarity, window sizes, chunk boundaries, etc), the model is unlikely to answer correctly, even if the underlying LLM has the capacity.

3.4.3 F1-Score

The F1-score provides a balanced measure between precision and recall [34], especially useful in applications like ours where both false positives (hallucinations) and false negatives

(missed answers) are problematic. It is the harmonic mean of precision and recall, calculated as followed:

$$\begin{aligned} F_1 \text{ Score} &= \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \\ &= 2 \times \frac{93.88 \times 92}{93.88 + 92} \approx 92.93 \end{aligned}$$

This F1 score demonstrates strong overall robustness, balancing the model's ability to answer correctly and its caution in avoiding hallucinations, though some loss occurred, perhaps due to missed context or chunk mismatches, highlighting areas for further refinement in chunk overlap logic, source formatting, and prompt improvements.

The reason the F1 score is particularly useful in our case is because it takes both false positives (incorrect answers) and false negatives (missed correct answers) into account. This is significant when dealing with domain-specific tasks, like archaeological research, where a single incorrect or missed answer can significantly affect the model's usefulness in practice. Overall, the F1 score provides a more balanced view of model performance compared to accuracy alone. While accuracy can be skewed by imbalances in the number of answerable and unanswerable questions, the F1 score mitigates this bias by considering how well the model balances both precision and recall.

3.4.4 Qualitative Analysis

In addition to quantitative metrics, we conducted a qualitative review of the model's responses. In several cases, the answers were holistically correct, even if they missed specific

details or phrased the information differently than expected. These types of responses suggest that while the core understanding is overall accurate, the model may occasionally sometimes overlook certain things, such as complex terminology or multi-part reasoning.

An example of a long answer question is as follows:

Long Answer Questions	Answer should contain the following:	Source
The author mentions that Sasanian society had multiple language groups and that surviving texts are mainly male-centric and religious. What is one challenge this presents for understanding Sasanian society?	It can be difficult to fully understand the history and culture of non-Iranian language groups, and the perspectives of women and the poor may be marginalized. (Both points can be considered valid answers).	Daryaee2009-SasanianPersiaRiseFallEmpire.pdf

Table 3.4.4.1 Long Answer Question Example

Where the result of the AI would be read to assess whether the overall gist and intent of the answer aligned with the expected response. However, this method is inherently subjective and lacks the strict reproducibility of quantitative metrics.

We acknowledge that while this has assisted us in identifying potential strengths and weaknesses in contextual understanding, it does not provide a consistent or scalable framework for benchmarking. Therefore, establishing more standardized qualitative evaluation methods, potentially with rubric-based scoring or expert reviews, is a key consideration for future development, as outlined in the next chapter describing future works.

3.4.5 Comparison to Baseline GPT-4

For comparison, we also tested the same GPT-4 model without our RAG-based enhancements. This version of the model, which relied solely on its pre-trained internal knowledge, produced incomplete or factually incorrect answers, particularly on domain-specific archaeological queries. The absence of direct access to reference material meant that the model would often hallucinate inaccurate information.

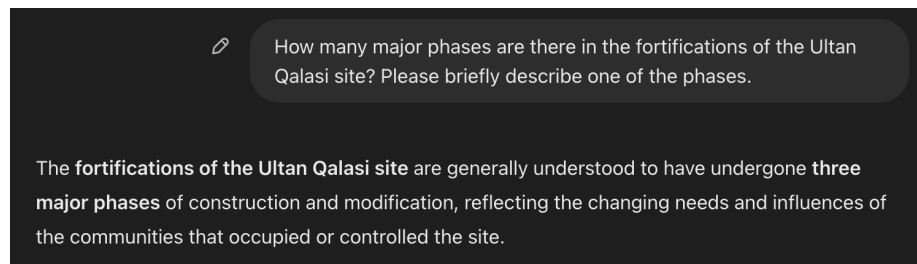


Figure 3.4.5.1 Incorrect Output by Baseline GPT-4

In contrast, our full pipeline showed improvements regarding such issues. When asked the same question across both versions, the RAG-enabled system was able to pull in the correct contextual chunks and provided answers aligned with the original source material. This would indicate a reduction in hallucinations and an increase in both the reliability and explainability of responses.

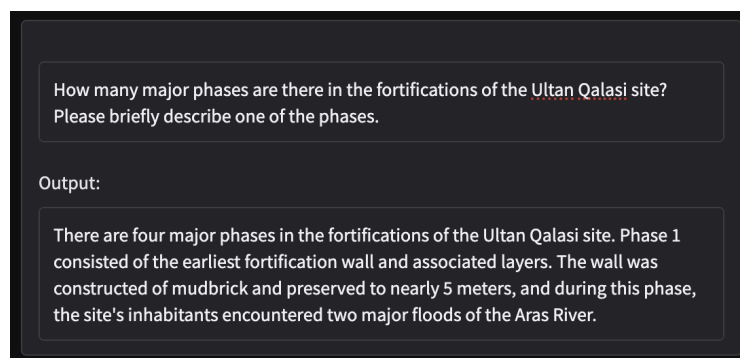


Figure 3.4.5.2 Correct output with our RAG-enabled system

3.5 User Interface

For the user interface (UI) of our AI application, we adopted a Flask-based approach to support local development, researcher interaction, and early-stage deployment. Flask, being a lightweight and highly flexible Python web framework, was chosen for its ease of integration with the rest of our Python-based backend, including LangChain, ChromaDB, and OpenAI's API. Its modularity also makes it ideal for future integration with production-grade servers and authentication systems.

As mentioned, we implemented a Gradio interface, a browser-based UI toolkit that allowed us to share the system for remote access, live demonstrations, and collaborative feedback sessions. Gradio requires minimal configuration and works seamlessly with our Python backend, making it ideal for quick deployments on temporary servers or for showcasing functionality without full-scale deployment.

3.5.1 Flask-Based Local Web UI

For local use, we implemented a clean, lightweight web interface using Flask. The design of the Flask UI emphasizes usability and intuitiveness, especially for non-technical users such as archaeologists or research assistants.

- On the left-hand side of the page, users can upload PDF or XML files directly into the system. This file uploader is integrated with our backend pipeline, triggering the CERMINE conversion and cleanup process upon submission automatically.
- On the right-hand side, a chat-like query interface allows users to enter natural language questions, which are then processed via the RAG pipeline.

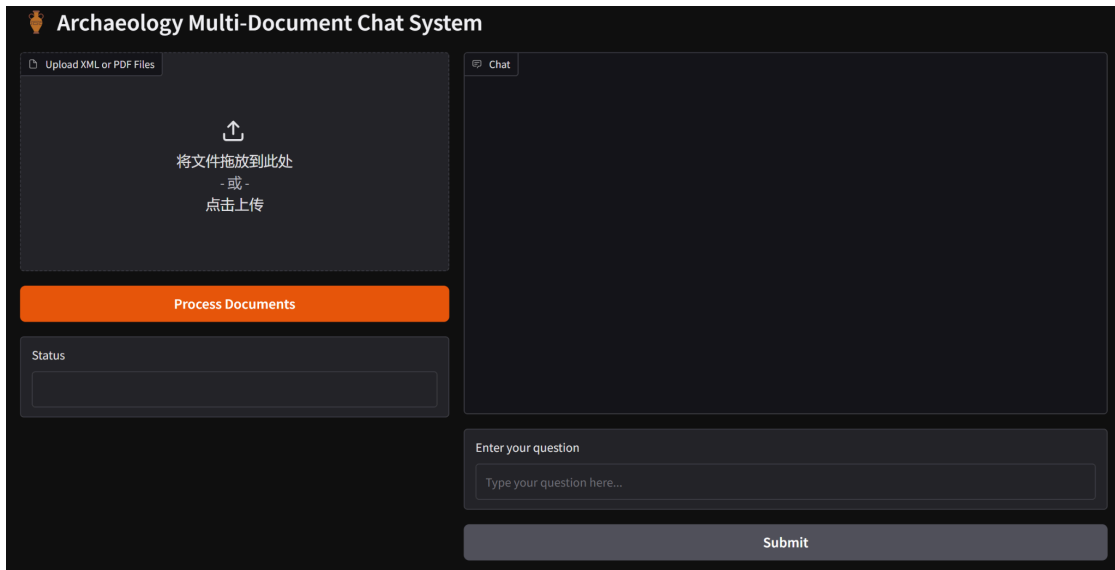


Figure 3.5.1.1 Screenshot of Web Interface

This layout would hopefully support a smooth, guided workflow: upload documents, ask questions, receive answers—with no need for separate scripts or command-line tools. Flask’s lightweight nature also makes it easily integrated with other server systems, supporting future deployment in cloud or academic environments.

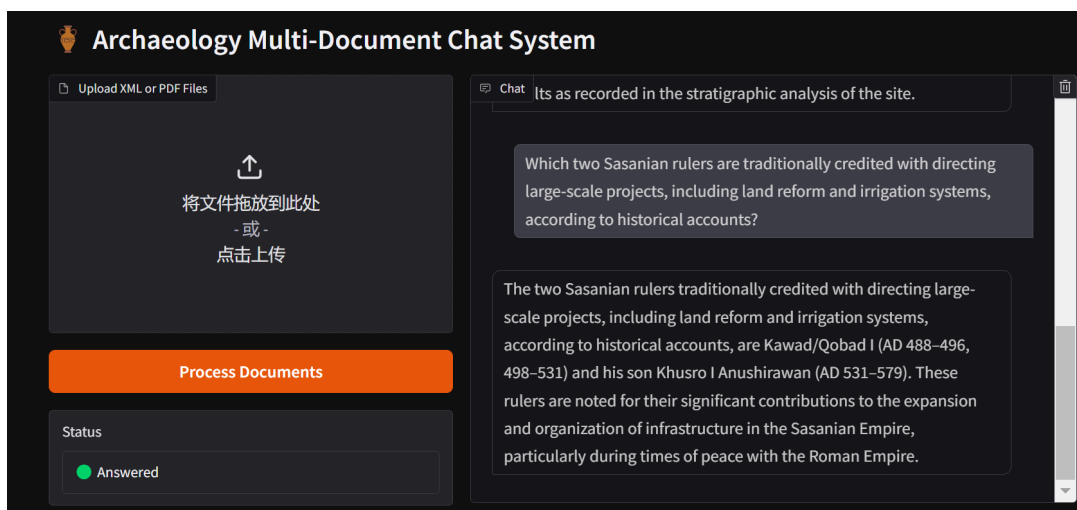


Figure 3.5.1.2 Screenshot of Application in Use

4. Future Works

In this section, we outlined various key areas for further development and improvement that could be made regarding our project's implementation.

4.1 Web Interface and Scalable Deployment

One of the major goals for future work is to transition our current local web based implementation towards a scalable server system that would significantly improve the accessibility and usability of the AI tool for a wider range of archaeologists.

By supporting a larger number of concurrent users, this system would allow archaeologists from various institutions and fields to upload their research, query the AI, and receive real-time, contextually grounded responses. This infrastructure would enable seamless collaboration, allowing researchers to test the model with their specific datasets and refine the AI's capabilities based on the increasing pool of resources uploaded by users.

Furthermore, by allowing archaeologists and researchers to directly interact with our application, it would assist in driving further improvements tailored to the needs of the archaeological community, and would directly support the next phase of work, where we aim to expand collaboration further and engage with experts in refining the system's performance.

4.2 Collaboration with Archaeologists

Engaging and collaborating with experts in the field of archaeology would assist us in performing long-form qualitative evaluations, refining question sets, and identifying gaps in

contextual interpretation, thus improving the AI's accuracy and relevance. Through direct feedback from archaeologists, we can enhance the model's ability to handle complex, long-form queries, improve its comprehension of archaeological terminology, and address any overlooked nuances in field-specific practices. Thus, it would allow for targeted adjustments and fine-tuning to improve its usefulness for archaeological research.

4.3 Enhancing Prompt Engineering

In the future, perhaps utilising the information received in the aforementioned archaeologists' feedback, improvements to our prompt templates could be made to better suit the needs of real-world archaeological research. This may involve tailoring the structure and tone of prompts to reflect the way archaeologists frame inquiries, incorporating domain-specific terminology, or adjusting the level of detail requested to align with academic expectations.

4.4 Expanding the Dataset

Expanding the dataset could also further enhance the AI's performance. Currently, the model operates on a relatively limited collection of texts (Approximately 217), which can cause constraints in its ability to generate comprehensive responses, particularly for niche or highly specialized queries.

By incorporating a broader dataset of archaeological material, we can significantly increase the breadth and depth of the knowledge base the AI can draw upon. This would make the system more representative of the many subfields and regional variations within archaeology. This could be aided by the aforementioned scalable deployment, as that would allow

archaeologists and researchers to upload their own curated datasets to the system. This collaborative model would not only enrich the existing knowledge base but also foster a feedback loop where the AI becomes increasingly tailored to the specific needs and interests of its users.

4.5 Improving Document Conversion and Cleanup

Despite the improvements in the document conversion pipeline using CERMINE, there are still challenges associated with extracting usable data from complex or poorly structured PDF documents. Refinements to the conversion and cleanup process could be made, such as recovering more usable data from difficult layouts, such as scanned documents.

Other PDF-to-XML conversion libraries and tools may be developed and implemented to achieve this, or alternative preprocessing methods could be explored to improve data quality and reduce the need for manual cleanup.

5. Conclusion

This project explored the development of a Retrieval-Augmented Generation (RAG) system tailored for archaeological research, with the goal of enhancing access to and interpretation of complex academic literature. By combining tools like CERMINE for document conversion, LangChain for prompt orchestration, ChromaDB for vector-based retrieval, and OpenAI's GPT-4 for generation, we built a prototype capable of delivering contextually grounded and accurate responses to answer archaeological queries.


The system demonstrated clear improvements over a standard generative model, particularly in reducing hallucinations and improving specificity. Through various metrics, we measured notable improvements in regards to the effectiveness of our approach. Our pipeline was further supported by a lightweight and user-intuitive web interface built with Flask, enabling easy document uploads and query interactions.

However, there are limitations and improvements to be made. Challenges in document conversion accuracy, prompt robustness, and database coverage all affect the system's performance. Addressing these issues will require further optimizations and improvements, alongside the help and collaboration of experts in the field of archaeology.

If these improvements are made, this AI model could become a useful tool for archaeologists, supporting their research and hopefully making their work more efficient and accessible. With continued development, the system has the potential to integrate seamlessly into existing research workflows and contribute meaningfully to archaeological research, especially in light of the growing adoption of generative AI across academic and scientific fields.

Bibliography

- [1] C. Labba, A. Alcouffe, E. Crubézy, and A. Boyer, "IArch: An AI Tool for Digging Deeper into Archaeological Data," *2023 IEEE 35th International Conference on Tools With Artificial Intelligence (ICTAI)*, pp. 22–29, Nov. 2023, doi: 10.1109/ictai59109.2023.00012.
- [2] "AI in Archaeology Applications | Restackio."
<https://www.restack.io/p/ai-in-archaeology-answer-historical-analysis-cat-ai>
- [3] M. Tenzer, G. Pistilli, A. Bransden, and A. Shenfield, "Debating AI in Archaeology: applications, implications, and ethical considerations," *Internet Archaeology*, no. 67, Feb. 2024, doi: 10.11141/ia.67.8.
- [4] M. Pasikowska-Schnass and Y.-S. Lim, "Artificial intelligence in the context of cultural heritage and museums," *European Parliamentary Research Service*, May 2023, [Online]. Available: [https://www.europarl.europa.eu/RegData/etudes/BRIE/2023/747120/EPRS_BRI\(2023\)747120_EN.pdf](https://www.europarl.europa.eu/RegData/etudes/BRIE/2023/747120/EPRS_BRI(2023)747120_EN.pdf)
- [5] M. Altaweel, A. Khelifi, and M. H. Zafar, "Using generative AI for reconstructing cultural artifacts: Examples using Roman coins," *Journal of Computer Applications in Archaeology*, vol. 7, no. 1, pp. 301–315, Jan. 2024, doi: 10.5334/jcaa.146.
- [6] T. But, "The Latest AI Innovations in Archaeology," *Historica*, Sep. 02, 2024.
<https://www.historica.org/blog/the-latest-ai-innovations-in-archaeology>
- [7] A. O. Historian, "Internet Archaeology - a free online Archaeology journal," *Seax Education*, Jun. 10, 2021.
<https://www.anoxfordhistorian.com/post/internet-archaeology-a-free-online-archaeology-journal>

- [8] B. Marwick, "Open Access to Publications to Expand Participation in Archaeology," *Norwegian Archaeological Review*, Oct. 2020, doi: 10.1080/00293652.2020.1837233.
- [9] S. Guizeni and S. Guizeni, "Exploring the usage constraints of GPT-4: An In-Depth Examination of Operational Limits," *Seifeur Guizeni - AI/ML Engineer & SEO Consultant*, Jul. 11, 2024.
<https://seifeur.com/gpt-4o-message-input-limit/>
- [10] "Content ExtRactor and MINER - User console." <http://cermine.ceon.pl/index.html>
- [11] "A Retrieval-Augmented Generation Framework for Academic Literature Navigation in Data Science." <https://arxiv.org/html/2412.15404v1>
- [12] F. Zhang *et al.*, "DH-RAG: A Dynamic Historical Context-Powered Retrieval-Augmented Generation Method for Multi-Turn Dialogue," *arXiv.org*, Feb. 19, 2025.
<https://arxiv.org/abs/2502.13847>
- [13] M. Ozkaya, "The RAG Architecture: Ingestion and Retrieval with Embeddings and Vector Search," *Medium*, Dec. 04, 2024. [Online]. Available:
<https://mehmetozkaya.medium.com/the-rag-architecture-ingestion-and-retrieval-with-embeddings-and-vector-search-9bb86adb0ff5>
- [14] "Introduction |  LangChain." <https://python.langchain.com/docs/introduction/>
- [15] "What is LangChain? - LangChain Explained - AWS," *Amazon Web Services, Inc.*
<https://aws.amazon.com/what-is/langchain/>

[16] “Build a Retrieval Augmented Generation (RAG) App: Part 1 |  LangChain.”

<https://python.langchain.com/docs/tutorials/rag/>

[17] “Content ExtRactor and MINEr - about.” <http://cermine.ceon.pl/about.html>

[18] “Text Splitters |  LangChain.”

https://python.langchain.com/v0.1/docs/modules/data_connection/document_transformers/

[19] S. Y. Kim, W. K. Lee, S. J. Jee, and S. Y. Sohn, “Discovering AI adoption patterns from big academic graph data,” *Scientometrics*, Jan. 2025, doi: 10.1007/s11192-024-05228-4.

[20] “Chroma Docs,” *Chroma Docs*. <https://docs.trychroma.com/docs/overview/introduction>

[21] “Vector embeddings,” *OpenAI*. <https://platform.openai.com/docs/guides/embeddings>

[22] Allenai, “GitHub - allenai/scibert: A BERT model for scientific text.,” *GitHub*.

<https://github.com/allenai/scibert>

[23] “What is a Vector Database? | A Comprehensive Vector Database Guide,” *Elastic*.

<https://www.elastic.co/what-is/vector-database>

[24] “Gemini,” Google DeepMind, Apr. 17, 2025. <https://deepmind.google/technologies/gemini/>

[25] “Text generation and prompting,” *OpenAI*.

<https://platform.openai.com/docs/guides/text?api-mode=responses>

[26] Wikipedia contributors, “Prompt engineering,” *Wikipedia*, Apr. 17, 2025.

https://en.wikipedia.org/wiki/Prompt_engineering

[27] G. Team, “Getting started with the Python client.”

<https://www.gradio.app/guides/getting-started-with-the-python-client>

[28] OpenAI *et al.*, “GPT-4 Technical Report,” *arXiv.org*, Mar. 15, 2023.

<https://arxiv.org/abs/2303.08774>

[29] Z. Ji *et al.*, “Survey of Hallucination in Natural Language Generation,” *ACM Computing Surveys*, vol. 55, no. 12, pp. 1–38, Nov. 2022, doi: 10.1145/3571730.

[30] L. Casini, N. Marchetti, A. Montanucci, V. Orrù, and M. Roccetti, “A human–AI collaboration workflow for archaeological sites detection,” *Scientific Reports*, vol. 13, no. 1, May 2023, doi: 10.1038/s41598-023-36015-5.

[31] J. Zimmer-Dauphinee, P. VanValkenburgh, and S. A. Wernke, “Eyes of the machine: AI-assisted satellite archaeological survey in the Andes,” *Antiquity*, vol. 98, no. 397, pp. 245–259, Dec. 2023, doi: 10.15184/aqy.2023.175.

[32] Wikipedia contributors, “Precision and recall,” *Wikipedia*, Mar. 21, 2025.

https://en.wikipedia.org/wiki/Precision_and_recall

[33] The Learning Agency, “Improving AI-Generated Responses: Techniques for reducing Hallucinations - The Learning Agency,” *The Learning Agency*, Jul. 16, 2024.

<https://the-learning-agency.com/the-cutting-ed/article/hallucination-techniques/>

[34] Wikipedia contributors, “F-Score,” Wikipedia, Apr. 14, 2025.

<https://en.wikipedia.org/wiki/F-score>