KMS HEALTHCARE

Using Language Models For Extracting Insights From Patient's Medical Records

A Research & Development Perspective

Authors: **KMS Healthcare Innovation Labs**

Published date: **August 28, 2023**

Abstract

In the current technological landscape, chatbots powered by Large Language Models (LLMs) have established a stronghold in the healthcare sector. However, decoding unstructured medical terms from patient histories is a significant challenge. This study emphasizes the conversion of hierarchical data structure, such as FHIR, into natural language. We utilize the Langchain agent framework for effective reasoning and decision-making, guiding the process of generating contextually accurate and meaningful responses. Moreover, Llama-Index offers a flexible storage solution, ensuring efficient retrieval of relevant information from the healthcare data landscape. Remarkably, this approach has achieved 87.98% accuracy in generating responses without extensive baseline model training. The extension of this work may consider a novel LLMs approach for representing the healthcare field and some valuable tools for developing a question-answering and summarization system. Overall, this research holds promise in enhancing chatbot efficacy within the healthcare domain.

Keywords:

LLMs | FHIR | REALM | RAGs | Langchain | Llama-Index | Vector Database | flexible storage

Table of Contents

1.Introduction

1.1. Overview

In recent years, the emergence of Generative AI, particularly Large Language Models (LLMs) has unlocked significant advancements in the development of a conversational chatbot, providing the system with the capability of generating a human-like response. This technology significantly enhances the decision-making process of medical professionals, elevating the efficiency and accuracy of patient care.

1.2. Challenges with Healthcare Data

In healthcare, a significant problem arises when we need to find very accurate information in a patient's medical records. FHIR, a standard for healthcare data exchange, uses formats like JSON and XML. These formats are complex for both healthcare professionals and conventional chatbots to interpret. To overcome these challenges, we establish the criteria for a via solution, encompassing several key aspects:

(i) The chatbot should be able to **access the patient's medical record, quickly retrieve relevant information, and provide a personalized response to the user.**

(ii) The chatbot should provide accurate information about the patient while **ensuring no factual inaccuracies or generation of incorrect content related to that patient's data.**

(iii) The chatbot should **demonstrate a noticeable medical knowledge**, be able to engage in medical discussion and deliver well-informed responses.

Although the third **(iii)** aspect plays a pivotal role in leveraging the inherent values of the solution, it's important to acknowledge that our focus is primarily on the first two in the current release. Our use case is centered around the first aspect **(i)**, with the second **(ii)** serving as an evaluator for our approach.

Notably, our methodology builds upon the foundational principles of the Retrieval-Augmented Language Model Pre-Training Model (REALM). Although we don't adopt this exact training model, its concepts, especially the **Retrieval Augmentation Generations (RAGs)** introduced by Google (Guu et al., 2020), underpin our approach. RAGs address essential elements contributing to the synthesis of domain-specific knowledge from Language Model Frameworks such as Langchain and Llama-Index. These pivotal aspects include key features and limitations that won't be explored in this paper. Instead, we assume the readers have prior knowledge in these aspects before moving on to our proposed solution section.

1.3. Scope

This research proposes an automated question-answering system tailored for healthcare professionals to enhance patient care and outcomes. Our main contributions are:

1. We retrieve and preprocess patient history records from the CONNECT API, transforming them into a human-readable format. This helps us prevent token limit issues when utilizing Azure OpenAI Services.

2. Our approach integrates Llama Index and Langchain, greatly enhancing information retrieval and the decision-making process in healthcare.

3. We introduce an automated evaluation process to validate the accuracy and reliability of our system's responses against reference documents.

2.Proposed Solution

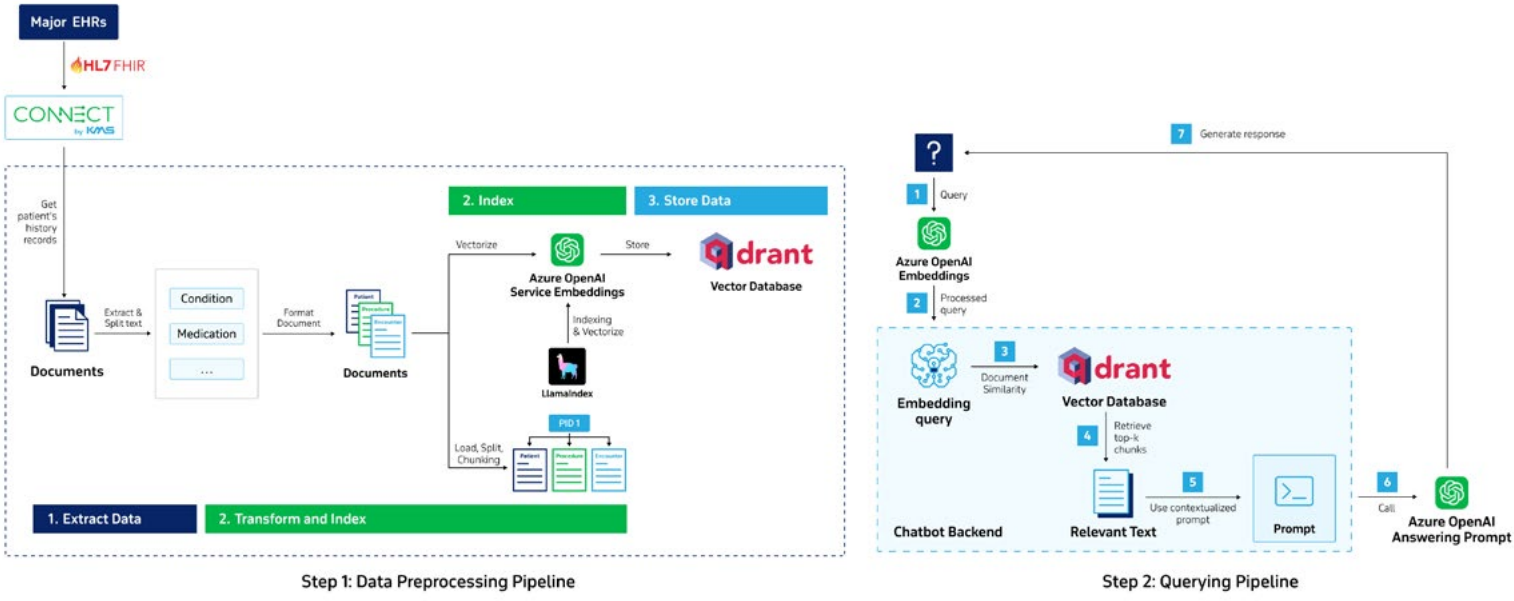

Figure 2. High-level architecture overview

Figure 2 shows the diagram that outlines our strategy for an advanced question-answering system, with the primary goal of accurately addressing user queries and summarizing information. Our implementation encompasses two distinct Question-Answering systems: one relying on Index Storage and the other on Qdrant Storage for retrieval and querying. Our objective is to conduct a comparative analysis of these systems in relation to their efficiency and accuracy of information retrieval over patient's medical data of FHIR.

Our approach incorporates various stages, including data preprocessing, transformation, indexing, and efficient query handling. Moreover, this system utilizes CONNECT to retrieve the patient's historical records, thereby augmenting its functionalities through complex processing stages. This involves getting a patient's data via CONNECT API, facilitating the extraction of significant insights.

2.1. Data Extraction

The initial phase involves a comprehensive data analysis, contrasting it with CONNECT API documents, leading to leaving out redundant fields for the latter. Subsequently, JSON data is transformed into a list of words, effectively streamlining the hierarchical structure into a singular level of list strings. Once this data structuring process is accomplished, we identify key attributes within the structure, subsequently converting them into coherent Natural Language expressions through the utilization of LLMs.

Conversely, during the preliminary structure assessment, frequency characteristics are identified within each resource type. These findings are then preserved as part of an entity termed "Additional_Retriever." This approach ensures a refined and streamlined data presentation, the extraction of significant data attributes through LLMs, and the incorporation of frequency-based information for enhanced data retrieval and analysis as in Figure 3.

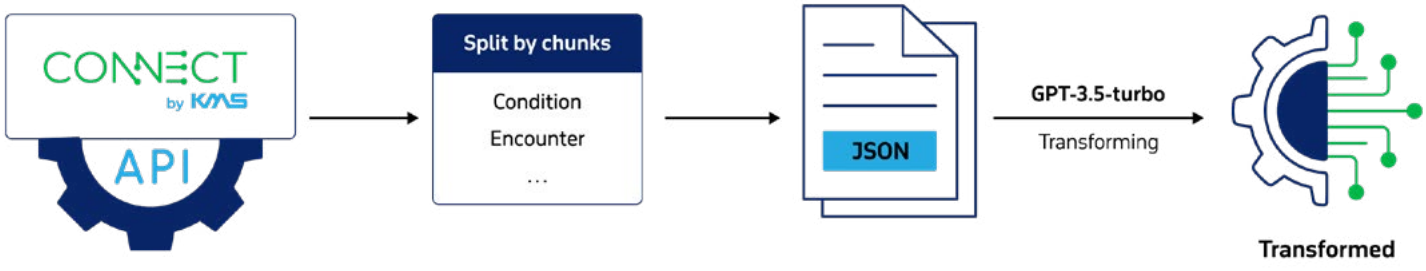

Figure 3. The preprocess layout for EHR data

2.2. Data Transformation

A. Qdrant Approach - Convert FHIR into Natural Language

The Preprocess Layer would eliminate several insignificant pieces of information from the original dataset by flattening FHIR JSON format in hierarchical order. At the Transformed layer, as depicted in Figure 4, we extract the key-values pair from the previous step and transform them into a meaningful sentence with LLMs.

Sample of transformation

<Originin>

```
{
    "resourceType": "Bundle",
    "type": "transaction",
    "entry": [{
        "resourceType": "Condition",
        "recordDate": "2014-06-06T15:54:16Z"
    }, {
        "resourceType": "Medication",
        "prescribe": {
            "text": "Penicillin V Potassium"
        }
    },
    . . . ]
}
```

- -

<Preprocessed>

```
[{
    "resourceType": "Condition",
    "recordDate": "6 June 2014 at 15:54"
},
{
    "resourceType": "Medication",
    "prescribe, text": "Penicillin V Potassium"
},. . . ]
```

- -

<Transformed>

```
[{
    "Resource type is Condition",
    "Record's date was in 6 June 2014 at 15:54pm"
},
{
    "Resource type is Medication",
    "Was prescribed a dosage of Penicillin V Potassium"
},. . . ]
```

Figure 4. Example for three processes in the preprocessing pipeline.

B. Qdrant and LLaMa Index Approach - Embedding

The operation of OpenAI embedding words is a transformative process in natural language understanding. By converting words into numerical values, this technique captures semantic relationships, contextual nuances, and linguistic features of words. The embedding process involves training on extensive text corpora, enabling the algorithm to learn intricate linguistic patterns. During inference, the algorithm maps each word to its corresponding vector, facilitating mathematical operations to capture semantic similarities, analogies, and context. These vectors effectively represent words in a continuous vector space, where spatial distances reflect semantic relationships. Consequently, this operation forms a fundamental basis for a range of Natural Language Processing (NLP) tasks, from text classification and sentiment analysis to machine translation and chatbot interactions. The resulting embeddings empower AI systems to comprehend and manipulate language, opening avenues for advanced language-driven applications across various domains.

C. Qdrant and LLaMa Index Approach - Indexing

The operation of Data Indexing words serves as a cornerstone in enhancing data accessibility and retrieval efficiency. Through a systematic process, raw data is meticulously contrasted with established document templates, allowing for the exclusion of redundant fields. This initial refinement streamlines the dataset, optimizing its relevance to the intended purpose. Furthermore, the transformation of JSON data into a list of words restructures complex hierarchical data into a simplified format, promoting uniformity and ease of processing.

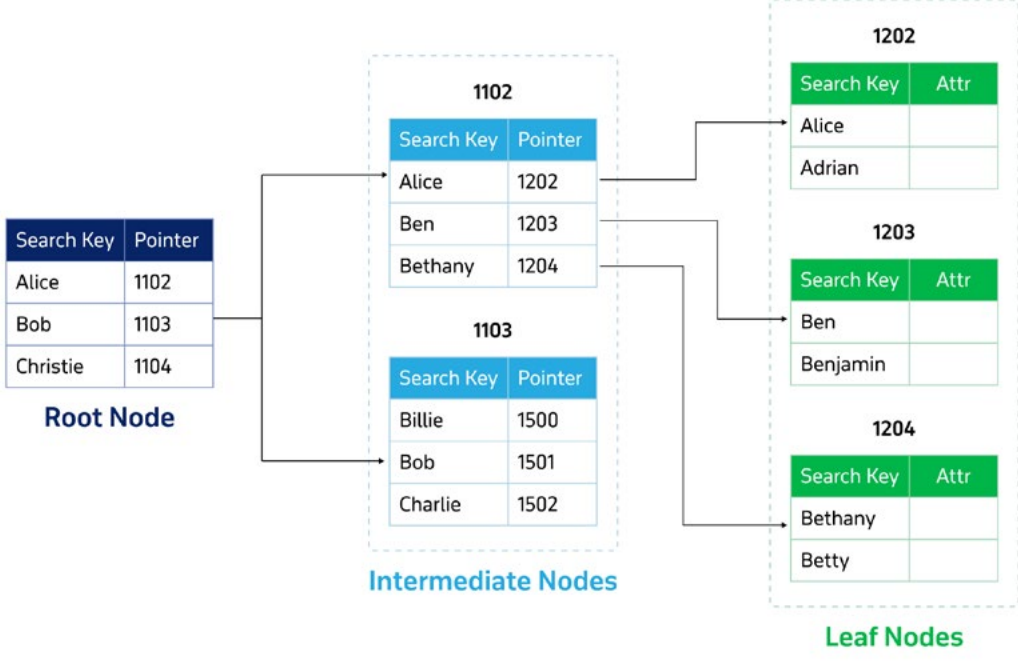

Figure 5. Data Indexing Diagram

2.3. Vector Representation Storing

To manage relationships and features within each patient record, a vector database is employed. This database enables the chatbot to represent values as vectors, capturing essential characteristics as described in Figure 6.

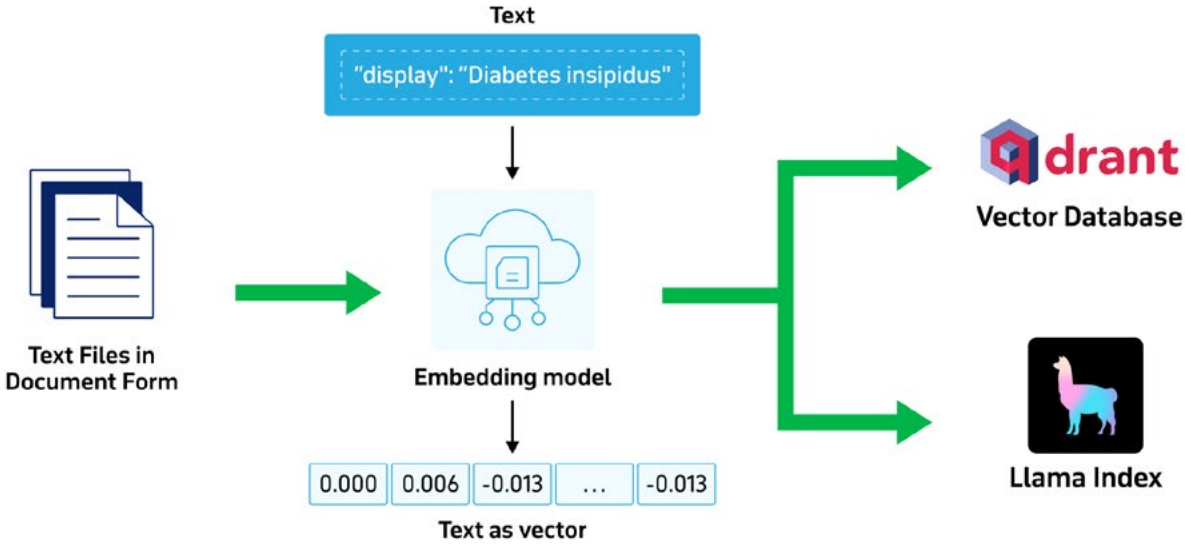

Figure 6. Vectorize an Store Data

A. LLama Index Approach

With LLama Index's adaptable data connectors, users can seamlessly incorporate data from various sources, and structuring it into optimized formats for LLMs. This enables efficient analysis of large-scale private data without the need of model retraining.

The knowledge base, such as organizational documents, is segmented into smaller segments to accomplish this. Each of these segments is then stored within a node object. Collectively, these nodes create an index alongside other nodes. This partitioning of the knowledge base is crucial due to the constrained token capacity of LLMs. This ensures a smoother and uninterrupted process of feeding documents into the system.
Upon loading the documents, an index is established from these documents, often utilizing tools like the VectorStoreIndex. This indexing process is a two-step procedure:

- This step involves vectorizing extracted sentences from JSON documents into numerical representation.
- These vectorized values are inserted into a collection, forming a set of vectors ready for searching within the vector database.

We have used Qdrant cloud to host our embeddings and textual documents for fast search and retrieval. In this project, patient information is stored in two different parts:

- **Patient Collection:** This is where all the important details about the patient are kept, like their medical history and personal information.
- **Additional Collection:** Here, extra information is stored, such as summaries and counts of records within certain time periods. This helps give an overview of patient data.

By separating information into manageable collections, Qdrant would make it easier to manage and find patient data, making healthcare processes smoother. In the following section, we will move on to our solution for our chatbot assistance system as an example that embodies the potential of these technologies from LLMs framework to data storage. Moreover, we also set size and distance to define the embedding size and how the similarity between vectors is calculated.

B. Data Organization in Qdrant and LLama-Index

Our solution leverages OpenAI embeddings to encode entire documents into numerical values, subsequently housing them within a structured database. Upon receiving a user query, we employ similar embedding techniques to convert the query into a vector representation and proceed to measure its distance from various document vectors, defined in Figure 7, utilizing the COSINE distance algorithm. This process yields a selection of K retrievers comprising documents closely aligned with the query.

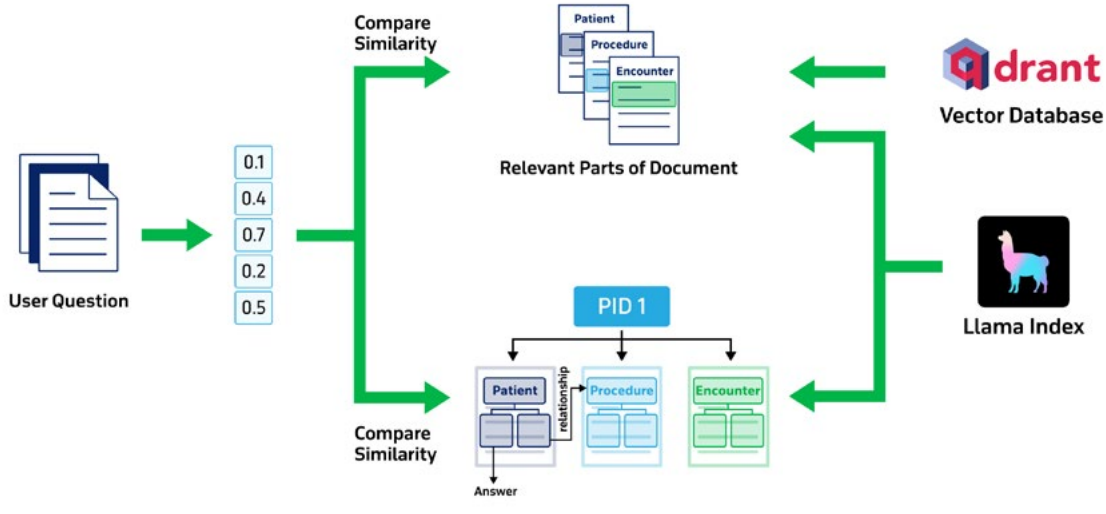

Figure 7. Search relevant document based on semantic similarity algorithm

As depicted in Figure 8, the initial segment encompasses the complete spectrum of patient data, securely stored in the primary collection of the database. The subsequent part incorporates supplementary data, such as summaries and record counts confined within specific date ranges; this is housed within the additional collection of the database. Lastly, the chat collection holds the history of conversations between users and the chatbot, streamlining the storage and retrieval of past records. This approach establishes a well-organized and effective system for saving, storing, and getting documents in the healthcare industry.

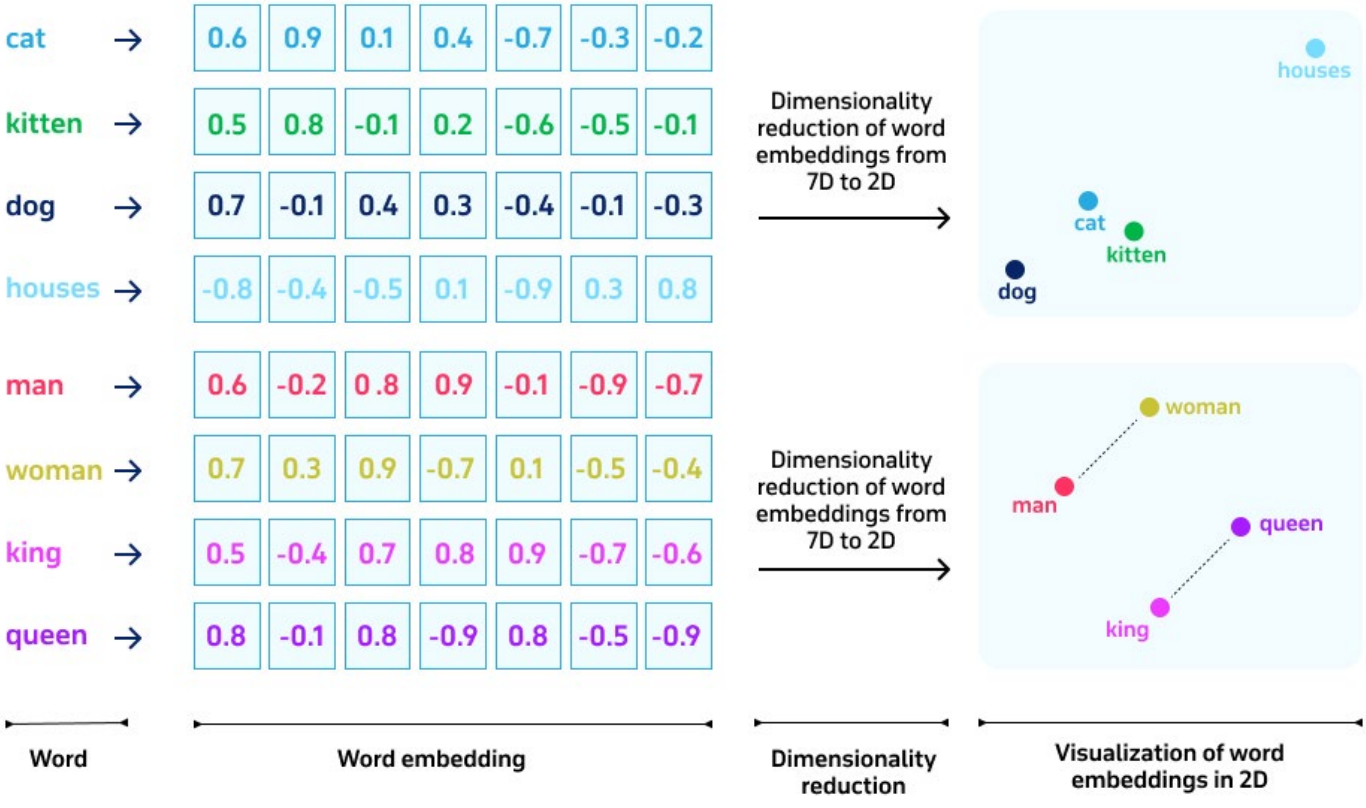

Figure 8. Example of information documents process in Vector Database

2.4. Query and Answer

When a user poses a query, the chatbot vectorizes it into a numerical representation and searches the vector database for relevant documents. While the raw output might match the context of the query, it may not be in a user-friendly format. To address this, we employ the Prompt Template from Llama Index to convert these outputs into natural, human-readable text and deliver the response to the user.

3.Result & Discussion

3.1. Evaluation

In the evaluation stage, we introduce an automated testing system that's designed to effectively assess the efficiency and accuracy of our chatbot's system in providing correct answers. This system uses LLMs to compare the content across various documents and generate a set of questions based on that content.

Upon analyzing the initial test documents, the system primarily leverages our application's specialized tools to extract answers for a wide array of user queries. These answers are then recorded in a Google Sheet. Subsequently, LLMs are employed once more to validate whether these answers align with the expected responses from the reference documents.

The outcomes are unequivocal: a "TRUE" result signifies a match between the answer and the expected one, and "FALSE" indicates a misunderstanding or the inability to procure a similar response. This information is illustrated in Figure 9.

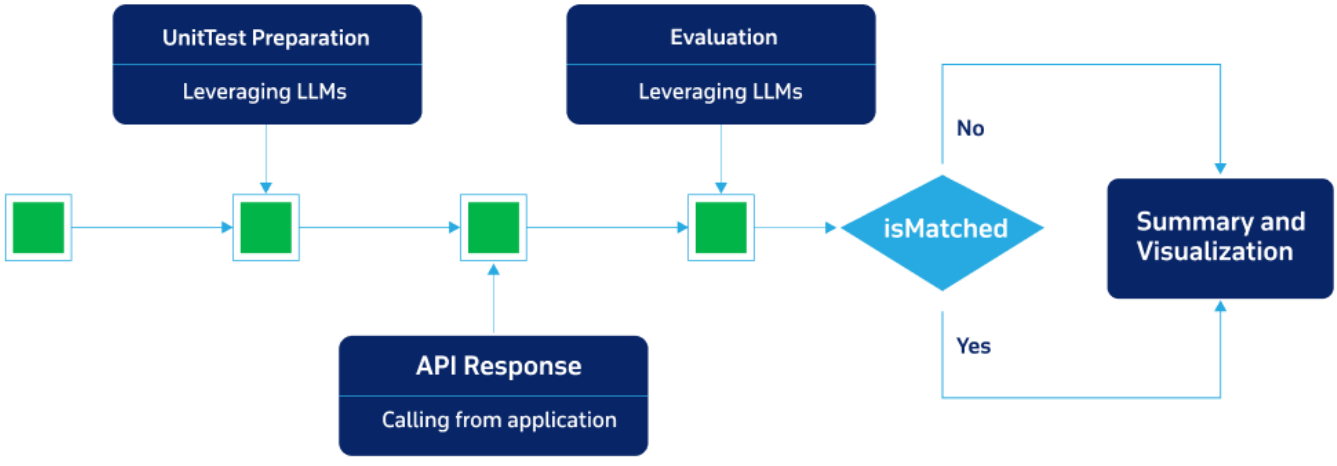

Figure 9. Evaluation process diagram

In summary, we aggregate the results of our evaluation by taking both correct and incorrect answers into consideration. To quantify the system's performance, we recall a formula that translates these results into numerical metrics.

Throughout the evaluation process, our automated testing pipeline utilizes LLMs to verify the precision and reliability of our application's response. The formula is presented below, with N denoting the total number of occurrences:

$$\text{Accuracy} = \frac{N_{correct}}{N_{correct} + N_{incorrect}}$$

3.2. Performance

When assessing the overall effectiveness of the deployed version, it's worth noting that the accuracy of the developed iterations displayed commendable performance levels.

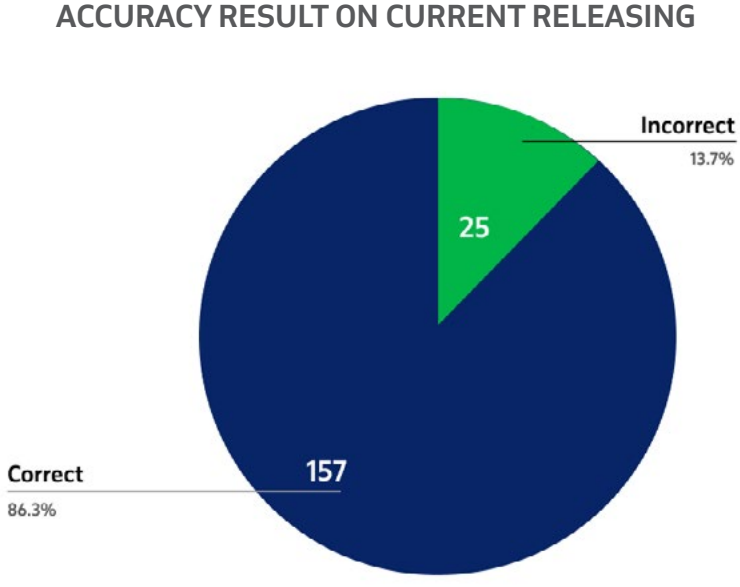

Figure 10. Proportion of Released Model's Accuracy

As illustrated in Figure 10, the distribution of Correct Answers takes account of a significant portion of the overall outcome, precisely at 86.3%. This stands in stark contrast to the relatively minimal representation of Incorrect Answers, accounting for a mere 13.7%, which is less than a fifth of the Correct Answers' percentage. These observations are grounded in the analysis of nearly 200 questions encompassing both unit test scenarios and reference documents. This lends a substantial degree of credibility to the outcome's validity and reliability. This performance analysis underscores the effectiveness and robustness of the implemented versions, reaffirming their capability to provide accurate responses across diverse contexts.

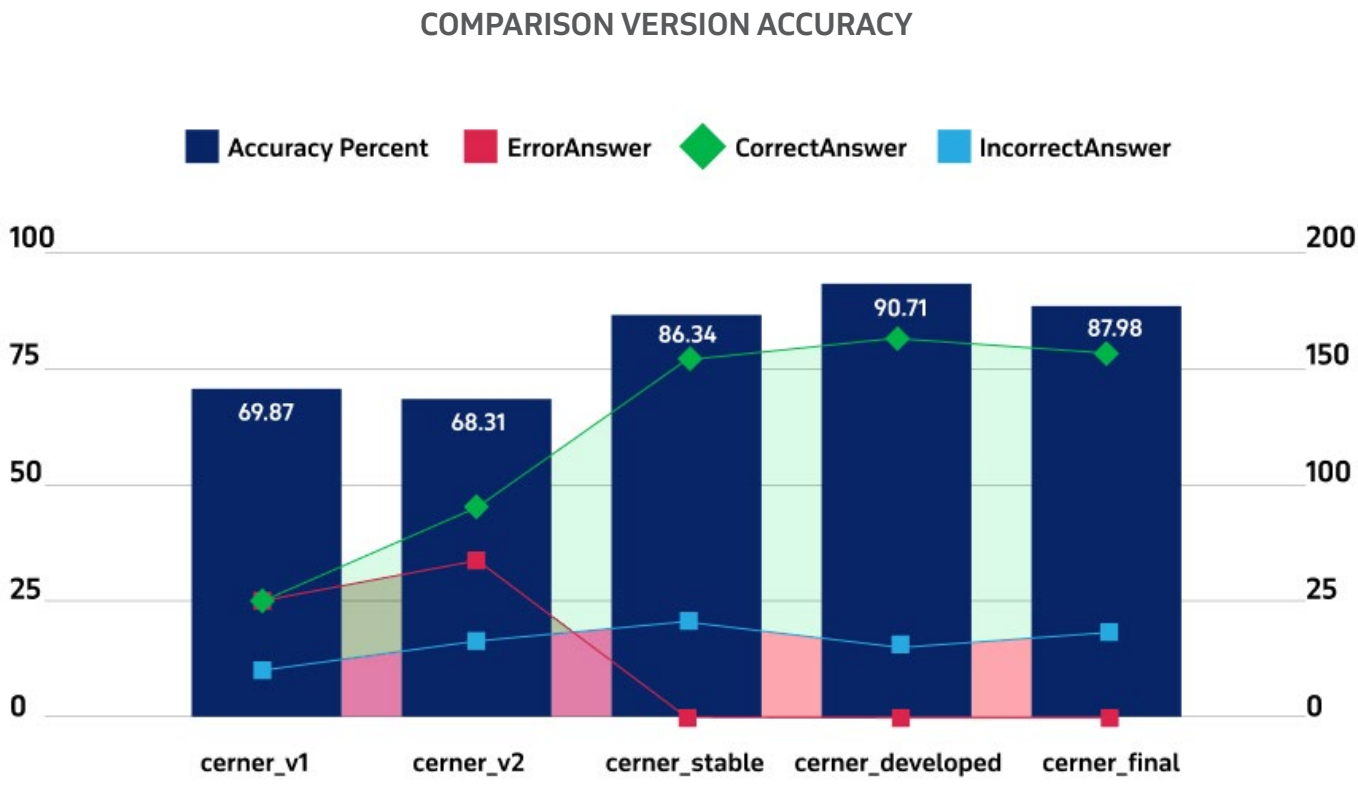

Figure 11. Comparison of every versions developed

In Figure 11, the initial version of our deployment encountered substantial challenges related to accuracy, evident in a notable prevalence of "ERROR" responses. This observation highlights the inherent instability of the Azure API, resulting in an accuracy plateau of merely 69%. Each subsequent version demonstrated a progressive improvement in accuracy. This progress was achieved through iterative adjustments to the preprocessing layers, optimizing data comprehensibility.

Remarkably, the version labeled "cerner_developed" showcased the most significant advancement, achieving a peak accuracy of 90%. However, our achievement was accompanied by some problems related to context sensitivity and API instability. Consequently, to achieve an optimal solution, we arrived at a final version that exhibited a commendable accuracy of 87.98%. This synthesis of successive refinements reflects our commitment to overcoming initial challenges, iteratively elevating accuracy, and ultimately delivering a robust and reliable solution for our target audience.

4. Demo Site

Allow us to introduce our demo website, seamlessly integrated with CONNECT Patient History Records. Within this platform, we will walk you through a demonstration using the patient profile of Nancy Smart. Throughout this presentation, we will provide a succinct overview of Nancy Smart's patient profile, as shown in Figure 12.

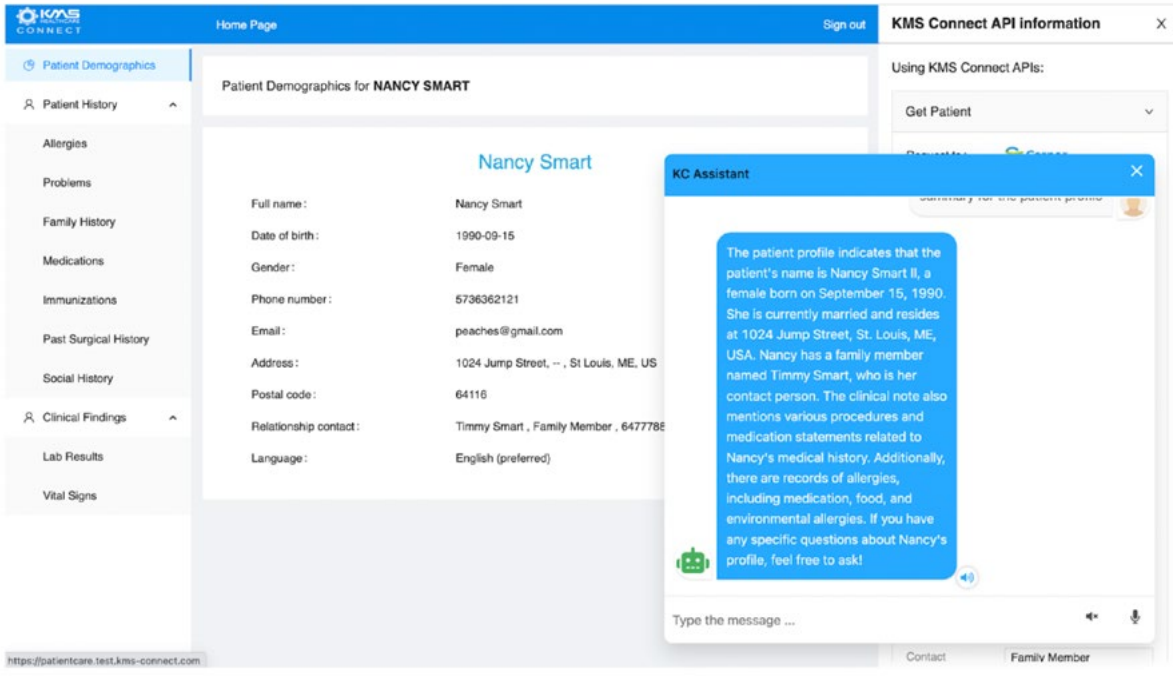

Figure 12: Demo site on the patient of Nancy

In Figure 13, you will see responses to user's inquiries, which can be automatically converted into speech if the voice feature is enabled in the message chat box. Users also have the flexibility to adjust the volume or replay the response by clicking the speaker icon adjacent to the response box, catering to different preferences. Additionally, users can opt for voice messages if they prefer not to communicate via text. This versatile approach empowers users to engage with the interface in a manner that best suits their needs and preferences.

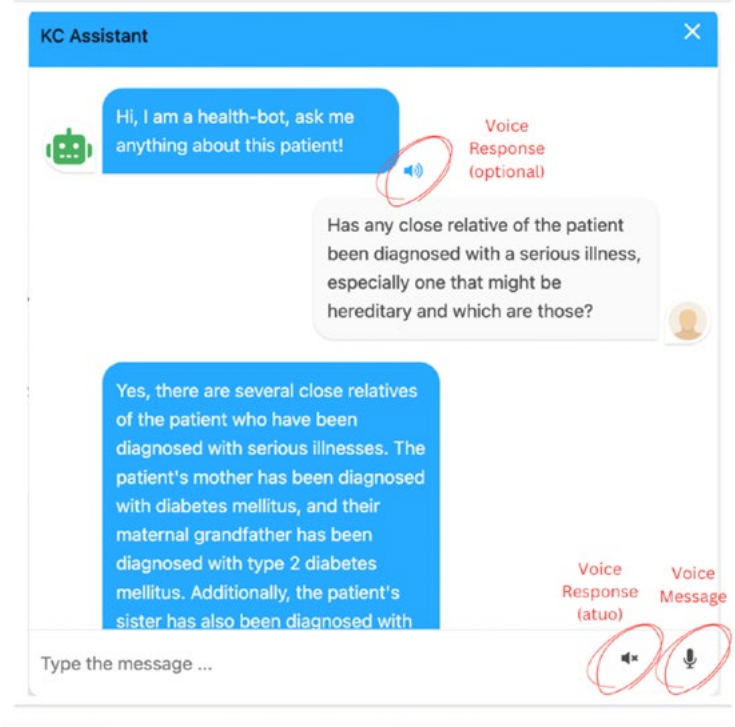

Figure 13: Review of chatbot features

In a different scenario, this chatbot is capable of managing various patient-related tasks through general questions, such as providing overviews or summaries, whenever a user logs into a different patient account, as shown in Figure 14. In cases where patient information is temporarily unavailable, the response will inform the user with the message, "Patient data is currently being updated", and will be made available as soon as the update process is complete.

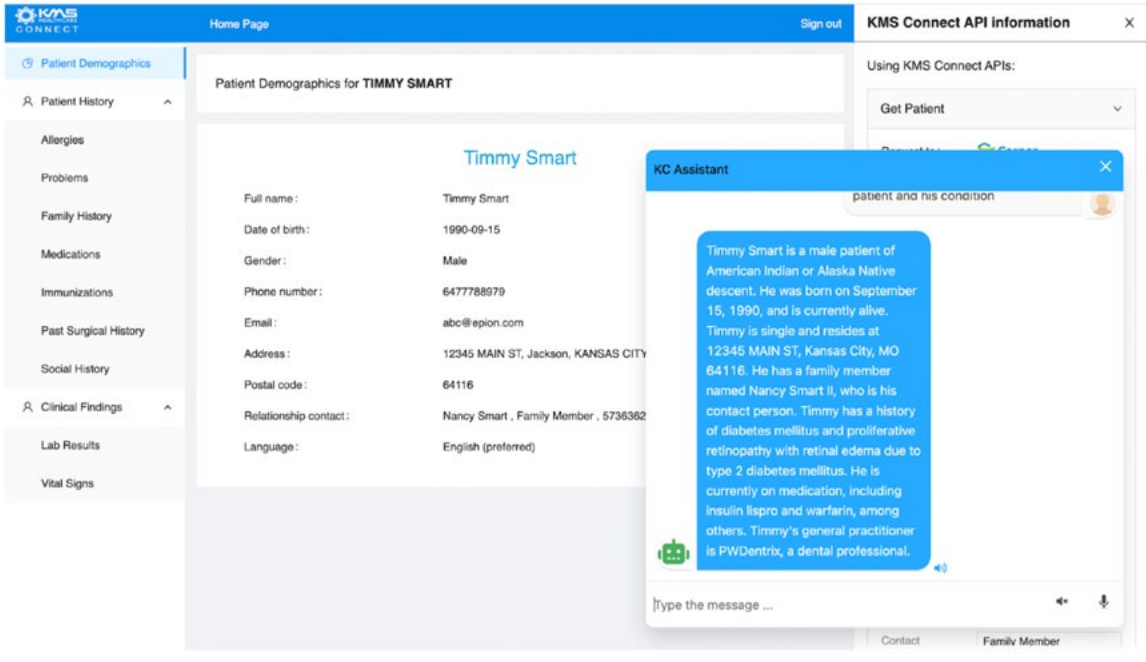

Figure 14: Demo site on patient of Timmy

5. References

1. N/A. (n.d.). System properties comparison pinecone vs. Qdrant vs. Redis. Pinecone vs. Qdrant vs. Redis Comparison.
https://db-engines.com/en/system/Pinecone%3BQdrant%3BRedis

2. Ram, O., Levine, Y., Dalmedigos, I., Muhlgay, D., Shashua, A., Leyton-Brown, K., & Shoham, Y. (2023, August 1). In-context retrieval-augmented language models. arXiv.org.
https://arxiv.org/abs/2302.00083

3. Agastya, A. (2023, August 10). Harnessing retrieval augmented generation with Langchain. Medium.
https://betterprogramming.pub/harnessing-retrieval-augmented-generation-with-langchain-2eae65926e82

4. Patange, N. (2023, August 10). Nikhil Patange on linkedin: Navigating Langchain: Overcoming Azure Openai's embedding batching... Nikhil Patange on LinkedIn: Navigating LangChain: Overcoming Azure OpenAI's Embedding Batching...
https://www.linkedin.com/posts/nikhil-patange-81823858_navigating-langchain-overcoming-azure-openais-activity-7095544547211976704-d6UF

5. N/A. (n.d.) Vector databases (Part 4): Analyzing the trade-offs
https://thedataquarry.com/posts/vector-db-4/