# **Medical Drug and News Search Tool**

INDEX-  
  
1. Executive Summary

2. Project Overview

3. Prototype Description

4. System Architecture

5. Implementation Details

6. Future Development Plans

7. Challenges and Solutions

8. Conclusion

9. Appendices

## Project Overview

Objective

The healthcare and pharmaceutical industries continually evolve with the development of new medical drugs and treatments. Professionals in these fields require up-to-date, accurate information to make informed decisions regarding drug development, competitive analysis, and regulatory compliance. However, accessing and synthesizing the vast amounts of medical data available can be challenging due to the sheer volume and complexity of the information.

The objective of this tool is to streamline the process of querying and summarizing extensive medical literature and news concerning pharmaceutical drugs. By integrating advanced search capabilities with a Large Language Model (LLM) for summarization, the tool aims to provide quick, accurate, and succinct summaries and analyses of medical drugs and their competitive landscape. This solution caters primarily to pharmaceutical researchers, healthcare professionals, and regulatory affairs specialists who need to make strategic decisions based on the latest medical research and market developments.

**Requirements**

The tool was developed to meet the following specific requirements outlined in the task description:

1. User Interface (UI) Design and Implementation:

* Develop a user-friendly interface where users can input natural language queries.
* Ensure the interface includes an input field for text queries and a submission button to initiate searches.
* Display results in a manner that is easy to understand, incorporating elements of data visualization and interactive feedback for enhanced user experience.

2. API Integration for News Search:

* Integrate with a reliable medical research study database or news API.
* Implement effective query transformation logic to convert natural language queries into API-compatible search queries.
* Justify the choice of API based on factors such as data comprehensiveness, update frequency, and ease of use.

3. Information Indexing and Relevance Filtering:

* Design an indexing system that organizes search results based on their relevance to the user’s queries.
* Develop mechanisms to filter out irrelevant data, ensuring the user receives the most pertinent information.

4. Summarization with a Large Language Model (LLM):

* Utilize an LLM to process and condense the indexed information.
* Ensure the summaries are pertinent and directly address the user queries.
* Explore innovative summarization techniques that enhance user understanding and engagement.

The tool is designed as a proof-of-concept prototype that demonstrates core functionalities, with the potential for further development into a comprehensive platform based on feedback and additional requirements.

## Prototype Description

**Current Capabilities**

The prototype developed functions as a sophisticated tool designed for the efficient querying and summarization of information pertinent to medical drugs, sourced from research papers and news articles. Below is a detailed description of its principal functionalities:

* Query Interface: The interface allows users to submit natural language inquiries concerning various aspects of medical drugs, including competitor analysis, recent research developments, and regulatory status updates. This feature supports a broad range of medical and pharmaceutical research queries, facilitating detailed investigative tasks by end-users.
* Search Execution: Upon the submission of a query, the system employs advanced natural language processing techniques to identify and extract key terms and phrases. These terms are utilized to conduct a structured search through the PubMed API, which has been seamlessly integrated to retrieve relevant medical literature, ensuring that the most pertinent articles are sourced in response to user queries.
* Data Summarization: After the relevant articles are fetched, their abstracts are succinctly summarized using a state-of-the-art Large Language Model (LLM), specifically OpenAI's GPT model. These summaries are crafted to directly address the user's questions, providing focused and actionable insights without the need for further distillation.
* Result Presentation: Summarized data, along with hyperlinks to the complete articles, are presented to the user in a clear and succinct manner. For queries that benefit from visual representation, the system generates and displays graphical data visualizations to better illustrate trends, comparisons, and statistical data, thereby enhancing the analytical value of the presented information.
* Interactive Feedback: Throughout the query processing phase, the system provides real-time visual feedback to users. This feature is crucial for enhancing the user experience by setting appropriate expectations for data retrieval and processing durations, thus maintaining user engagement during wait times.

Technologies Used

The prototype leverages a combination of web technologies and AI tools to provide a robust solution:

* Flask: A lightweight WSGI web application framework used to serve the web interface and handle backend operations such as routing and request handling. Flask was chosen for its simplicity and flexibility in building web applications.
* HTML/CSS/JavaScript: These technologies form the backbone of the frontend, creating a responsive and intuitive user interface. JavaScript, along with libraries like jQuery, is used to manage user interactions and AJAX requests for seamless user experience.
* OpenAI's GPT-3.5: This LLM is employed to perform complex natural language processing tasks, including keyword extraction and text summarization, ensuring high accuracy and relevance in the information provided.
* PubMed API: Integrated for its extensive database of medical literature, allowing the system to retrieve current and relevant medical articles based on user queries.
* Chart.js: Used for data visualization, enabling the presentation of search results in graphical formats to enhance the analytical utility of the information presented.

This technology stack was selected to optimize the development process, ensure scalability, and maintain reliability in handling various types of user queries and data processing requirements.

System Architecture

This section provides a detailed exposition of the architecture and operational mechanisms of the medical drug search tool. It outlines the structural design, data processing, and interaction dynamics between the system components, ensuring a robust understanding of the prototype's functionality.

**Current Architecture**

The prototype's architecture is meticulously crafted to manage complex data interactions and deliver precise information to users. Here's an overview of the key components:

* Frontend: Constructed using HTML, CSS, and JavaScript, the frontend serves as the primary interaction layer for users. It captures user inputs and displays results. AJAX is employed for asynchronous communication with the backend, enhancing user experience by eliminating the need for page reloads.
* Backend: Powered by Flask, this lightweight yet powerful web application framework handles all server-side logic, including API requests, data processing, and session management. Flask’s flexibility makes it ideal for rapid development and effective scaling of web applications.
* API Integration: The system integrates with the PubMed API, a crucial component for accessing a comprehensive database of medical literature. This integration ensures that the system retrieves the most relevant and recent articles based on user queries.
* Large Language Model (LLM): Initially using BioMistral, the system transitioned to OpenAI's GPT-3.5 model due to its superior performance in natural language processing tasks. These tasks include extracting key terms from user queries and summarizing complex medical texts, ensuring that the responses are accurate and contextually relevant.

A screenshot of a computer

Description automatically generated

**Enhanced Data Processing and Filtering Techniques**

To refine the accuracy and relevance of the search results returned by the PubMed API, the system employs specialized filtering techniques. These techniques utilize niche filtering terms tailored to the domain of medical drug research. The backend dynamically constructs these filters based on the extracted keywords from user queries, focusing on terms that are particularly significant in pharmaceutical studies, such as "clinical trial results," "FDA approval," and "market analysis." This targeted approach ensures that the search results are not only relevant but also of high importance and utility to the user's specific inquiries.

**Transition to OpenAI's GPT Model**

Initially, the prototype employed BioMistral, a custom LLM, for processing user queries and summarizing medical texts. While BioMistral provided adequate initial results, challenges were observed in terms of response accuracy and the adaptability of the model to the wide range of medical terminology encountered.

To enhance the system’s performance, a strategic decision was made to transition to OpenAI's GPT-3.5 model. This shift was motivated by several factors:

* Improved Accuracy: GPT-3.5 offers state-of-the-art accuracy in natural language processing, significantly enhancing the relevance and precision of keyword extraction and text summarization.
* Better Adaptability: OpenAI’s model demonstrates superior capabilities in handling diverse and complex medical datasets, adapting more effectively to the specialized language used in medical research and drug development.
* Scalability: The GPT model supports a scalable architecture, which is crucial for expanding the prototype's capacity to handle increasing query volumes and data complexity.
* Ongoing Support and Updates: OpenAI continuously updates its models with the latest advancements in AI research, ensuring that the system remains at the cutting edge of technology.

Implementation Details

This section delineates the practical aspects of the prototype’s implementation, focusing on the frontend design, backend logic, and the integration of the Large Language Model. Each component is tailored to optimize the tool's performance and user experience.

Frontend Implementation

The user interface of the prototype is designed to be intuitive and user-friendly, facilitating effortless interactions for users seeking information on medical drugs. Here’s how the frontend has been implemented:

* User Interface Design: The design utilizes a clean and modern aesthetic to ensure ease of use. HTML5 and CSS3 are employed to create a responsive layout that adapts to various devices and screen sizes, ensuring accessibility and usability. JavaScript, particularly with AJAX, is used to handle user interactions without page reloads, enhancing the dynamic nature of the application.
* Interaction Flow: The main interface includes a query input box where users can type their natural language questions. A submit button initiates the search process, during which an animated loader appears to indicate that processing is underway. Results, once ready, are displayed below the query input area, allowing users to easily review the summaries and access links to full articles.
* Accessibility and Usability: Special attention is given to accessibility standards, including keyboard navigability and screen reader support, ensuring that the tool is accessible to a broad range of users, including those with disabilities.

Backend Implementation

The backend serves as the backbone of the application, handling data processing, API integration, and response management. Here’s an overview of the backend implementation:

* Server-Side Logic: Built using Flask, the backend manages routing, requests, and session data efficiently. Flask's ability to handle multiple requests simultaneously makes it ideal for an application requiring high availability and quick response times.
* API Integration: The integration with the PubMed API is critical for fetching relevant medical articles. The backend dynamically constructs API queries based on the keywords extracted from user inputs. It then processes the API responses, fetching the necessary article IDs and their corresponding abstracts for further processing.
* Error Handling and Security: Robust error handling mechanisms are in place to manage potential failures in API responses or processing hiccups.

LLM Integration

The integration of OpenAI’s GPT-3.5 model is pivotal to the tool's ability to provide accurate and concise summaries of medical literature. Here's how the LLM is integrated and utilized:

* Model Integration: The LLM is integrated via OpenAI’s API, which the backend accesses with secure API keys stored in environment variables. This setup ensures that the model can be updated or maintained without affecting the rest of the system.
* Data Processing: When the backend receives abstracts from the PubMed API, it sends them to the LLM for summarization. The model processes the text, focusing on extracting and condensing information relevant to the user's query, leveraging its advanced natural language understanding capabilities.
* Response Optimization: The summaries generated by the LLM are formatted and optimized for readability before being sent to the frontend for display. This step ensures that the information is not only accurate but also presented in an easily digestible format.

Future Development Plans

This section outlines the strategic enhancements and innovations planned for future iterations of the medical drug search tool. These developments are designed to enhance functionality, scalability, performance, and security, ensuring that the tool remains at the forefront of technological advancements and continues to meet the evolving needs of its users.

Planned Features

1. Advanced API Integration and Information Indexing:
   1. Multiple API Integration: To broaden the scope and depth of information retrieval, I plan to integrate additional APIs such as ClinicalTrials.gov, which provides access to global clinical trial data. This integration will enable the tool to offer more comprehensive coverage across different types of medical research.
   2. Advanced Query Transformation: Leveraging more sophisticated NLP techniques will enhance the conversion of user-input natural language into more precise search queries. This will improve the accuracy and relevance of the search results.
   3. Relevance Filtering: Implementation of advanced algorithms to rank results based not only on keyword matching but also on contextual relevance and user intent. This will ensure that the most pertinent information is prioritized in the response to the user.
2. Summarization with LLM Enhancements:
   1. Synthesis of Information: Rather than summarizing articles individually, I aim to synthesize information from multiple sources into a coherent response. This approach will provide users with more integrated and contextually comprehensive answers.
3. Enhancing User Interface (UI):
   1. Multimedia Integration: To make the information more accessible and engaging, I will include multimedia elements such as images, diagrams of drug mechanisms, and video explanations from medical experts where relevant.
4. Utilization of Domain-Specific LLMs:
   1. Medical Data-Trained LLMs: Implement LLMs that have been specifically trained on medical datasets to improve the accuracy and relevance of responses, particularly in handling complex medical jargon and data.
5. Interactive Chatbot Interface:
   1. Conversational UI: Develop a chatbot-like interface to allow users to interact with the system in a conversational manner. This feature will make the application more intuitive and user-friendly, mirroring natural human dialogue.
6. Retrieval-Based QA System:
   1. Context-Aware Summarization: Enhance the summarization process with a retrieval-based QA system. This system will identify key documents and use LLMs to generate concise, context-aware summaries or answers directly related to user queries.
7. Enhanced Data Retrieval:
   1. Qdrant for Vector Search: Implement Qdrant to enhance the data retrieval mechanism, enabling fast and relevant searching through embeddings.
   2. Vector Embeddings for Information Retrieval: Utilize SentenceTransformerEmbeddings to generate vector embeddings of texts. This semantic search approach will improve the relevance of search results beyond simple keyword searches, using models such as "NeuML/pubmedbert-base-embeddings".
8. Multilingual Support:
   1. Translated Search Feature: Introduce a feature to handle queries in different languages by implementing real-time translation, expanding the tool’s usability across diverse linguistic demographics.

Scalability and Performance Improvements

* Cloud-Based Infrastructure: Migrate to a scalable cloud infrastructure to handle increased traffic and data loads seamlessly.
* Load Balancing: Implement load balancing techniques to distribute user requests efficiently across servers, improving responsiveness and uptime.
* Caching Mechanisms: Introduce caching of frequently accessed data to reduce latency and enhance the speed of query processing.

Challenges and Solutions

Throughout the development of the medical drug search tool, several technical challenges were encountered. These challenges provided valuable learning opportunities that not only improved the current system but also imparted critical insights for future developments.

**Technical Challenges**

1. Model Evolution:

* Initial Model Selection: The project initially considered integrating MistralAI's Mixtral-8x7B-Instruct-v0.1. However, due to the high costs associated with the model and its substantial storage requirements, it was deemed impractical for local deployment.
* Transition to BioMistral and GPT-3.5 Turbo: Seeking a more feasible solution, the project transitioned to BioMistral, followed by a shift to OpenAI's GPT-3.5 Turbo. This evolution was driven by the need for a model that was not only more cost-effective but also less demanding in terms of storage. GPT-3.5 Turbo, in particular, offered superior performance, significantly enhancing accuracy and efficiency in natural language processing tasks.

2. Enhancing Prompts:

* Prompt Engineering: The system initially struggled with generating relevant search queries and summarizing results effectively. Iterative refinement of the LLM prompts led to more precise and actionable outputs, highlighting the critical role of prompt engineering in maximizing LLM performance.

3. API Integration Complexities:

* Search URL and Parameters Optimization: Integrating with the PubMed API presented challenges in effectively crafting search URLs and tuning parameters to fetch the most relevant articles. Through a process of trial and improvement, the system's ability to formulate optimized queries enhanced, leading to better data relevance and quality.
* PyMed library vs llama\_index’s PubmedReader vs. Direct API Access: The PyMed library, initially considered for simplifying API interactions, proved less efficient in terms of memory usage and response time compared to direct API calls. Opting for direct interactions with the API resulted in faster data retrieval and lower resource consumption.

**Learning Points**

1. Model Selection and Adaptation: This project underscored the importance of selecting the right AI model that aligns with the specific needs and constraints of the application. Continuous evaluation and adaptation of the chosen technology are crucial to ensure it meets the evolving demands effectively.
2. Prompt Engineering Proficiency: Developing expertise in prompt engineering is crucial as effective prompts significantly influence LLM performance, affecting the relevance and quality of outputs. This skill will remain a focus for future development.
3. Efficiency in API Utilization: Gaining experience in API integration taught the importance of understanding and optimizing API usage. Direct API interactions, while initially more complex, provided greater control and efficiency, highlighting the need for proficiency in API documentation and functionality.
4. Balancing Complexity and Performance: The experience highlighted the trade-offs between using third-party libraries and direct implementations. This stressed the importance of evaluating external dependencies for their impact on system performance and maintainability.

Conclusion

**Achievements**

The development of the medical drug search tool represents a significant advancement in the field of medical informatics. The current prototype successfully demonstrates the integration of advanced natural language processing technologies to facilitate efficient and accurate retrieval and summarization of medical information. Key achievements of the prototype include:

* User-Friendly Interface: The tool features an intuitive user interface that simplifies the process of querying complex medical data, making it accessible to a broad range of users, from medical professionals to researchers.
* Advanced Natural Language Processing: By leveraging OpenAI's GPT-3.5 Turbo, the system provides precise keyword extraction and generates coherent, concise summaries of extensive medical texts.
* Effective API Integration: The seamless integration with the PubMed API ensures that the tool has access to up-to-date and relevant medical literature, crucial for the accuracy of the information provided.
* Scalable Architecture: The backend, built on Flask, offers a robust and scalable framework that supports the tool's complex data processing needs and can accommodate future expansions.

**Future Vision**

Looking ahead, the vision for the tool is to establish it as a leading solution in the medical research field, known for its accuracy, reliability, and user-centric design. Future developments will focus on expanding the tool’s capabilities, including:

* Incorporating Additional Data Sources: By integrating more specialized medical databases and APIs, the tool will broaden its coverage of medical information, providing users with a more comprehensive research tool.
* Enhancing Interactivity and Accessibility: Future versions will aim to include more interactive elements such as multimedia content and a conversational interface, making the tool even more intuitive and engaging.
* Expanding Language Support: To increase its global usability, the tool will include features for multilingual support, allowing users to query and receive information in various languages.

The ongoing development and enhancement of this tool are expected to have a substantial impact on the medical research community, providing a powerful resource that enhances decision-making, accelerates research, and ultimately contributes to better health outcomes.

References

* <https://pubmed.ncbi.nlm.nih.gov/help/>
* <https://browse.welch.jhmi.edu/searching/pubmed-search-tips>
* <https://wiki.nested-knowledge.com/doku.php?id=wiki:autolit:search:pubmed>
* <https://pub.towardsai.net/using-large-language-models-to-build-a-biomedical-chatbot-and-deploying-it-af7818a0ab7f>
* <https://www.youtube.com/watch?v=A_m3tCqdts4>
* <https://www.nature.com/articles/s43856-023-00370-1>