ChatMaJA
From Data to Answers: Developing a Medical Q&A System with LLMs and Transformers

BY : AGATA KACZMAREK, PRANJAL SHARMA, JAN SMOLEŃ, MATEUSZ STĄCZEK

ChatMaJA

CHATMAJA IS A QUESTION & ANSWERING MACHINE BASED ON LLMS AND TRANSFORMER.

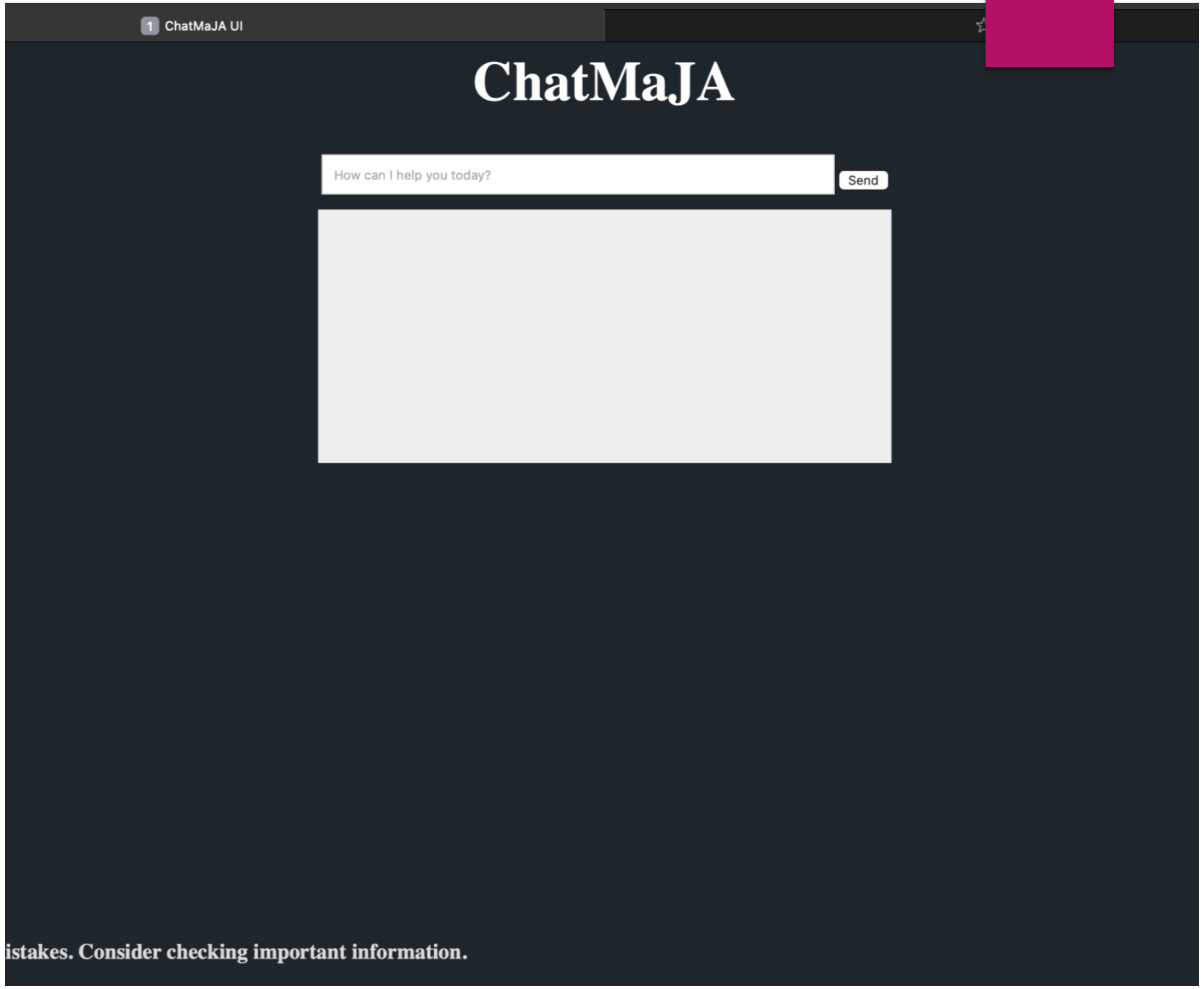
IT IS BASED ON PAST 10 YEARS MEDICAL DATA

Data Collection

- We simply extracted data in multiple excel files containing PMIDs from PubMed
- These PMIDs were fed into PubMed2XL.com to retrieve title and abstracts.
- Finally, all the files were merged into single master sheet.

Web UI Designing

- We used Flask Framework to develop WebUI for this question-answering machine.
- It is developed locally over the docker



istakes. Consider checking important information.

Abstract embedding

- Abstract are split into chunks, each consisting of 3 sentences
- Each chunk is embedded seperately using BERT from HuggingFace. We use mean pooling to create vectors of even length
- Embedding are stores in .csv file, which will be then used to populate vector databse
- For research and demonstration purpose, we only processed first 100 articles

Vector Database - Milvus

- We are using a vector database Milvus run locally in a Docker container
- Our collection consits of fields:
  - Id – in the form of {PMID}{chunk_number}
  - title
  - chunk
  - chunk_embedded

Relevant document retrieval

- Search of relevant chunk is conducted by embedding the query and using cosine similarity
- Top k relevant chunks are returned. Id, Title of the article and string representation of chunk are also included.

Pros and coins of the approach

+ Milvus offers a very convenient docker image, paired with good quality Python interface and implemented vector similarity search

+ BERT contextual embeddings are relatively simple to compute and work quite well in many cases

- the system struggles with handling keywords

- so far, everything is only tested locally on a small portion of data

- The best way the data is divided into chunks needs to be further researched

Future work - basic

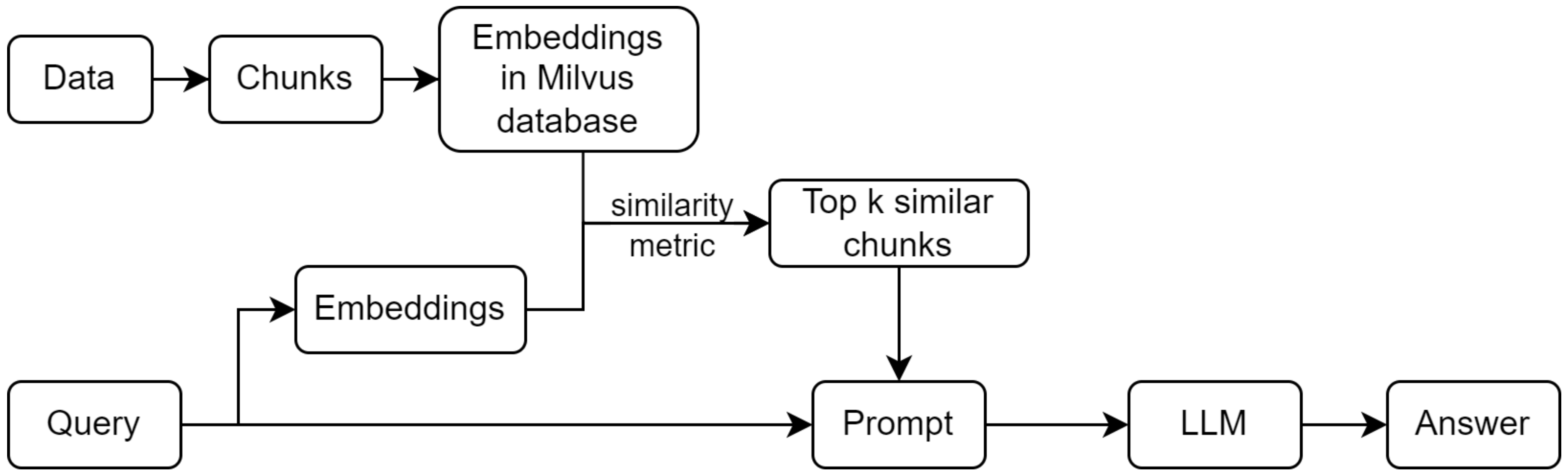
- Choose LLM – at the beginning we will try one of GPTs
- Add prompt generation to use LLM to generate easily readable answer
- Create Test Data – it will probably be based on PubMedQA dataset
- Integrate with our UI
- Evaluate

Improving our solution

We are not satisfied with the performance of the current solution, therefore:

- **Data acquisition – automate it** (now it's a *very manual process*)
- **Chunking** – investigate chunking methods from LangChain
  - **Current:** split after every 3 sentences
  - **New** (example): *RecursiveCharacterTextSplitter*, with overlap
- Similarity measure – check if changing it matters
- Questions types – might try more than just yes/no.
- **LangChain – integrate all components** of our solution **using Chains**

The end – thank you for your attention! Discussion time


Data
Chunks
Embeddings in Milvus database
similarity metric
Top k similar chunks
Embeddings
Query
Prompt
LLM
Answer