MedGPT: Advancing Conversational Medical Diagnostics

Yifan Bian, Chuyuan Zhong, Donglin Jiang

Georgetown University

ANLY-5800 Advanced Natural Language Processing

Fall 2023

Introduction

Medical diagnosis and documentation are time-consuming, yet vital, tasks in healthcare. To help improve this, we developed MedGPT, an AI-powered conversational chatbot to assist with both medical diagnosis and automating clinical documentation. MedGPT leverages recent advances in natural language processing (NLP) from HuggingFace and OpenAI to understand patient symptoms and provide preliminary diagnostic suggestions to healthcare professionals.

The core function of MedGPT is a conversational interface where patients can explain their symptoms. Using advanced NLP pre-trained on medical data, MedGPT analyzes these symptom descriptions to provide possible diagnostic suggestions and pertinent questions to gather further information. Healthcare professionals can then use these preliminary diagnoses to interview patients and determine an accurate diagnosis.

Additionally, MedGPT automates medical documentation for more efficient clinical workflows. As the provider interacts with the patient, MedGPT takes note of the input information. Key details are extracted and structured, which can be populated into electronic health record (EHR) fields. This structured data allows the patient-provider conversation to directly feed medical records in the future, saving providers time on documentation.

Motivation

Healthcare resources are often overburdened, leaving patients unsure of how to interpret symptoms or access help. MedGPT directly addresses this challenge through an accessible, conversational interface, providing everyone equal access to preliminary medical guidance. By design, MedGPT prioritizes understanding patient concerns and quickly suggesting potential diagnoses or appropriate next steps when symptoms are urgent.And by capturing conversations

as structured data, MedGPT reduces time-consuming record-keeping, giving back more time for patient care. Our ultimate goal is quality healthcare available to all. MedGPT serves both patients and providers to move towards more efficient, equitable, and collaborative healthcare experiences.

Data

To train the capability of our MedGPT, we used two datasets collected from Kaggle: MedDialogue and GenMedGPT. Both datasets are composed of JSON-formatted dialogues between patients and doctors, which allow us to train the model. Each dialogue contains an instruction specifying how to answer medical questions as a doctor, the patient's symptom input text, and the doctor's diagnostic response text. GenMedGPT and MedDialogue contain 5452 and 1024 medical conversations, respectively. The symptoms cover a wide range of conditions across medical domains such as gastrointestinal, neurological, respiratory, and more. By training MedGPT on these real patient-doctor conversations, our chatbot learns to analyze inputted symptoms, gather further details, and provide preliminary diagnoses or next steps, thus mirroring real clinical evaluation workflows.

Method

In our study, we implemented the MedGPT chatbot, designed to provide patients with conversational answers about their symptoms using a retrieve-and-generate architecture. This architecture consists of two primary stages, ensuring that the chatbot's outputs are both specialized and relevant. In the first stage, data ingestion and indexing, we used the MedDialogue and GenMedGPT datasets as our foundational input. This phase involves

extracting dialogues by parsing raw data files, a process facilitated by JSON loaders that consistently organize data according to predefined schemas.

The FAISS package plays an important role in this stage, as it is employed to construct a dense vector index, which is crucial for later semantic search functionalities. Each dialogue is converted into a high-dimensional embedding vector using the HuggingFaceEmbeddings module, which is optimized for biomedical text. This transformation is represented by the equation:

$$V = \text{HuggingFaceEmbeddings (Text)}$$

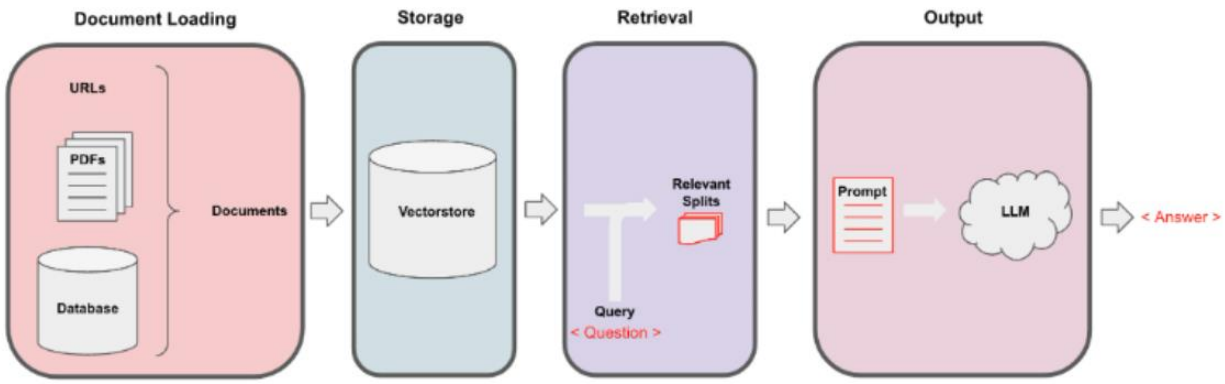

Figure 1: The Workflow of the Dialogue part

In the second stage, semantic retrieval, the system encodes a symptom inquiry into an embedding vector using the same language model. This encoded inquiry is then matched against the pre-constructed index to retrieve dialogues with the highest semantic similarity. Following this, the contextual generation process begins. The retrieved dialogues provide the context for our generative model, MedLLaMa, a 7 billion parameter large language model (LLM) specifically trained on medical conversations. This model utilizes the context and query to continue the dialogue, with the generative process formalized as

$$\text{Response}=\text{MedLLaMa}(\text{Context}, \text{Query}),$$

where response is the generated text, context is the set of retrieved dialogues, and query is the patient's original input.

The model's parameters, such as temperature, maximum tokens, and repetition penalty, are meticulously fine-tuned to produce outputs that are coherent and non-repetitive. Specifically, the temperature is set to 1×10^{-5} to minimize variance, the maximum tokens are limited to 512, and the repetition penalty is set at 1.1 to discourage redundancy.

Then, we incorporated the BAAI BGE-small model from the Sentence Transformer library by Huggingface for improved embedding techniques, specifically tailored for biomedical texts. This allowed us to create a more effective dense vector index using the FAISS library, significantly enhancing our algorithm's capability to identify and extract the most comparable questions from the dataset.

Finally, we employed 'llSourcell/medllama2_7b', a transformer-based model with 7 billion parameters, pre-trained on scientific text. This model underwent specialized fine-tuning with the MedQA dataset, enhancing its proficiency in providing detailed and accurate responses to medical inquiries. The diverse training program, which included a wide range of medical dialogues and journals, equipped the model with the ability to interpret complex health-related questions effectively. Its ability to draw from various sources ensures that it delivers informed responses, making it an invaluable tool for guiding users to pertinent medical information.

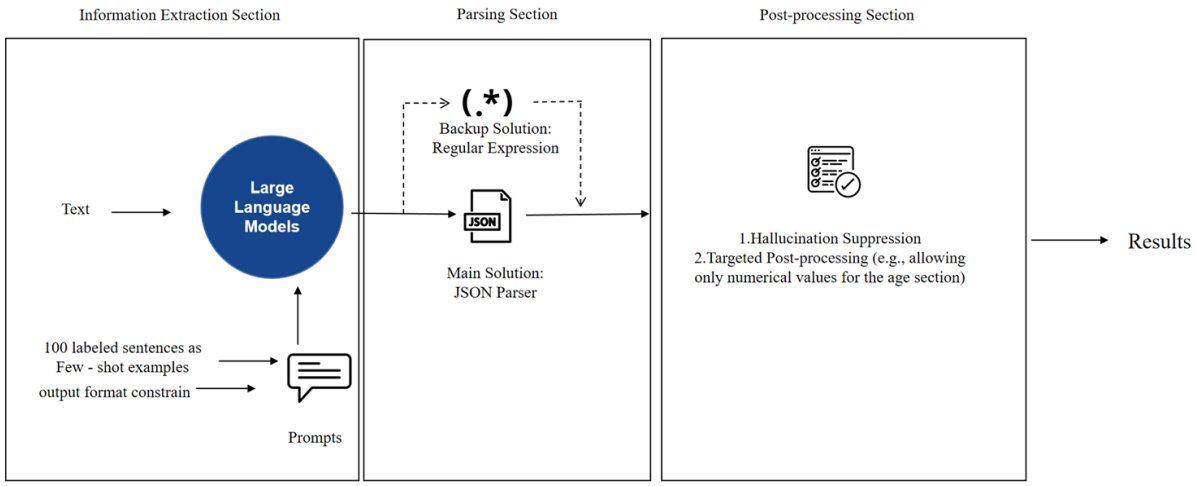

Figure 2: The Workflow of Data Extraction with a Single Large Language Model

Information Extraction

In this section, we focused on the development and optimization of prompt templates for effective data extraction. A prompt template is a predefined structural framework designed to guide the delivery of prompts, typically comprising components like instructions, contexts, and output indicators. Moreover, we have introduced constraints on the output format to enhance stability and readability, essential attributes considering the use of LLMs as an intermediary process in our system.

We used "decode only" LLMs: Llama2, MedLlama2, and GPT-3.5. These models combine self-attention, position encoding, and multi-task training, enhanced by extensive training data and numerous parameters. A crucial aspect of their effectiveness is Reinforcement Learning from Human Feedback (RLHF), which markedly improves their prompt-based learning abilities.

Building upon existing research that highlights the utility of JSON outputs, we have refined our approach to include more detailed prompts. These prompts encompass specific value types, formatting rules, and annotations, thereby improving the quality of the output. Additionally, the integration of Markdown formatting, as utilized by OpenAI, has been found to significantly bolster performance. Our research indicates that the combination of Markdown and JSON formatting effectively reduces instability in LLM responses, particularly when faced with slightly varying inputs. This improvement is consistent across different model types, demonstrating a remarkable degree of generalization.

We also explored the Chain of Thought (CoT) approach, aiming to dissect complex extraction tasks into simpler, more manageable components. This method involves initially directing the model to search for potential norms, followed by the identification of valid entities. Furthermore, the concept of a system role prompt is employed to direct specific tasks. This technique involves referencing professional roles, personality traits, or behaviors to guide the model's responses. Our research underscores the benefit of assigning a unique identity or nickname to the bot. In line with this finding, we have named our bot ‘Watson’.

Parsing

In this section, our primary solution focuses on integrating a JSON parser. The primary role of this parser is to transform the LLM output into a JSON-like format. This conversion is crucial for the seamless integration of the output with subsequent stages of the processing pipeline. The JSON parser serves as a bridge, facilitating the transition from the raw, textual output of the LLM to a structured format that can be easily analyzed in subsequent processes.

Considering the variability and complexity of outputs from some LLMs, we designed a backup parsing solution to address potential shortcomings in the primary parsing method. It is recognized that without meticulously defined constraints, approximately 14% of the pipeline is at risk of failure. A simple oversight, such as omitting a comma in constraint writing, can lead to parsing errors. To mitigate these risks, the backup solution employs a robust approach to handle even non-standard JSON formats which may occasionally arise from model responses. By implementing specific constraints, the responses are rendered JSON-like, simplifying the extraction and analysis of relevant information. We used regular expressions to isolate content between two entities and remove unnecessary punctuation marks.

Enhanced Entity Extraction

In this section, we addressed a specific challenge in entity extraction tasks: the occurrence of hallucinations, where LLMs generate results not present in the input data. To solve this, we've implemented a comprehensive sanity check process to validate the responses. This process begins with language detection, using the architecture's capability to handle multilingual inputs. Once the language is identified, the input is concurrently processed by three distinct models, each with unique proficiencies in extracting different types of information. The models used are:

1. **MedLlama2**: Specializes in extracting information about age, conditions, symptoms, precautions, and drugs.
2. **Llama2**: Complements MedLlama2 by focusing on additional, varied data points.
3. **GPT-3.5**: Performs well in identifying names and provides augmented support in drug-related information extraction.

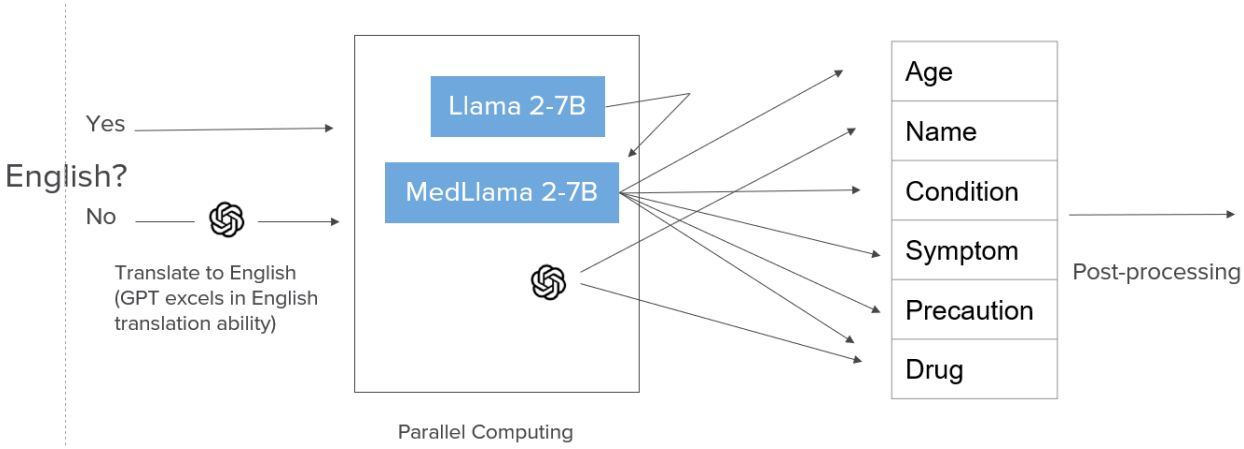

Figure 3: The Ensemble of Results of 3 Large Language Models

This not only enhances the accuracy of entity extraction but also ensures the validity of the extracted data by comparing it against the original raw text. In the end, we significantly reduce the risk of accepting erroneous data and improve the overall reliability of the extraction process.

Results and Evaluation

To evaluate the models' performance, we used the word error rate (WER) as a benchmark. The WER measures the number of edits needed to change the model-generated text into the correct reference text. This rate provides insight into the deviation of the models' outputs from the expected answers.

Our results indicate that MedLlama2, the fine-tuned version of the Llama2 model, outperforms its counterparts. This underscores the significance of model fine-tuning when tailoring LLMs to specific tasks. Most notably, by combining the strengths of Llama2, MedLlama2, and GPT-3.5 into an ensemble model, we observed a substantial reduction in WER, dropping by as much as 0.35. This ensemble approach yields a significant enhancement in performance,

	Llama2	MedLlama2	GPT-3.5	Ensemble
Word Error Rate	0.83	0.72	0.91	0.56

The performance of 4 different solutions

Table 1: The WER of 4 different solutions

Future Work

Our system currently faces a notable limitation due to the enforcement of strict output format constraints. This rigidity can impede the effectiveness of the CoT approach, especially in complex scenarios that require multiple input-output layers. The strict format requirement may hinder the model's contextual response capabilities, leading to suboptimal solutions in certain cases.

In terms of evaluation metrics, we currently rely on Word Error Rate (WER) as a primary indicator of performance. However, this metric has its drawbacks. WER's sensitivity to the format of the validation set can lead to overfitting, and it may not adequately reflect the model's proficiency in understanding and responding to intricate medical queries. This is because WER primarily measures textual accuracy without considering the contextual relevance and depth of the response.

Retrieval-augmented generation (RAG) can be a good solution to address these problems. It will historical medical dialogues as context, enriching the model's understanding and response accuracy. RAG can improve the relevance and depth of the model's responses, especially in the

domain of complex medical queries. This integration of RAG will serve as a significant step towards overcoming the current limitations and enhancing the overall efficacy of our system.

References

Taha, A. (2023). *GenMedGPT.json* [Data set]. Kaggle.

https://www.kaggle.com/datasets/amirataha/raw-medicalqa/data

Taha, A. (2023). *MedDialog.json* [Data set]. Kaggle.

https://www.kaggle.com/datasets/amirataha/raw-medicalqa/data

OpenAI. (2021). *gpt-3.5-turbo-16k*. https://platform.openai.com/docs/models/gpt-3-5

Hugging Face. (2023). *bge-small-en-v1.5*. https://huggingface.co/BAAI/bge-small-en-v1.5