Brance Founding AI Applied Researcher Hiring Task

Name: Raviraj Prajapat

Linkedin Profile: <https://www.linkedin.com/in/ravirajprajapat/>

Date Challenge Received: 7th July 2023

Date Solution Delivered: 8th July 2023

1. Problem Statement

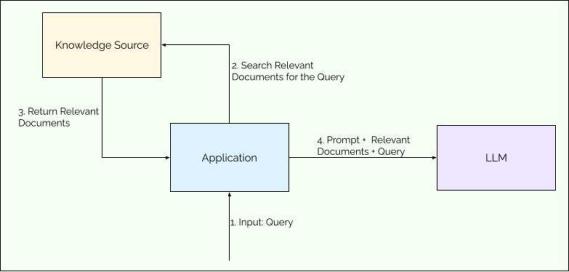
The Idea is to build a working production grade chatbot application using Retrieval Augmented Generation (RAG) method. User should be capable of talk/query to knowledge base in natural language. You are free to your LLMs and orchestrators like langchain or any other standard libraries from github. It should be powered by LLM hence I can use OpenAI [Not open source].

2. Approach

RAG is standard go to approach for efficient utilization of LLM capabilities over large amount of data which LLMs might not have knowledge about. It is explained neatly in the [original paper](https://arxiv.org/abs/2005.11401). Paper finds that RAG models generate more specific, diverse, and factual language than a state-of-the-art parametric-only seq2seq baseline.

Each User query will through query encoder, will use retriever using max inner product search and pull out candidate document with highest probability of having query’s result in it. Then These candidates are sent to LLM along with query to get the actual answer.

3. Solution



(Image courtesy : [https://miro.medium.com/v2/resize:fit:4800/format:webp/1\*FdJyBbb6AQswpoojuJgGEw.jpeg](https://miro.medium.com/v2/resize:fit:4800/format:webp/1*FdJyBbb6AQswpoojuJgGEw.jpeg) )

I have built the app modular and flexible. You can upload the database in real time (Any txt file) and can query the data. If that is not required, one can hardcode it and pull from S3 bucket on a prod system.

1. I have used following components & Modules:
   1. LLM : OpenAI - gpt-3.5-turbo
      1. Why: GPT-4 is overkill for these kind of problem. GPT3 has enough capability to engage with content of 5-6 paragraph. Also, it is faster. Other good options are light weight open-source models (Falcon, MPT or all in <7B range). Claude-instant is also great choice is speed is really the concern.
   2. Embeddings : OpenAI text-curie-001
      1. Why : For shear purpose of convenience I used open-ai embedding. In production I would always go for MPNET model given the quality-speed-cost reasons.
   3. Voice Model : Whisper-tiny
      1. Why : Given the size and small machine for deployment on huggingface. Just for Demo purpose. Suggest to for bigger model.
   4. Database : Chroma
      1. Why : Again becuase of easy to implement in a POC. For production, my choice would be weaviate or milvus. They offer advantage of stability and latancy.
   5. Langchain:
      1. TokenTextSplitter
      2. RetrievalQA
      3. Chain Type “stuff”

Why : Token Text splitter is easy to interpret and provide cohesive cut-off content to the LLM for interpretation. I did lot of experiments earlier with different splitters and I have found it easy to comprehend and easy to reproduce results.

Retrieval QA with chaining of “stuff” make single hit for each query to LLM, making the whole process cost efficient and fast.

1. How to control Hallucination:
   1. Better quality control over retrieved result would be first step for this. I used hard limit of 5 retrivals to not to feed misleading and off the topic in context to the LLM.
   2. I Let standard prompting “Please answer the user's question about document’ do the job as it was already giving very consistent result. A tuning in prompt would control hallucination if found any. Also, forcing the model to output in specific format like bullets or JSON would provide more control over hallucination.
2. Multi-Linguality :
   1. Already been taken care of by gpt-3.5 model
   2. If we change the LLM, we can opt for a middle layer of translation using commercial or open source (Optus, marianmt or libre-translate)
3. Edge Cases covered :
   1. Nothing will work without OPEN-AI key.
   2. Key is saved safe as ENV Variable, hence secrecy is maintained.
   3. Error cases are flagged with Gradio Error control.
4. **Evaluation of the answer** (Approach only, given the time constraints):
   1. We can generate synthetic data using knowledge base and LLM (Open AI).
   2. A Entailment model (nli) using mpnet or debarta model can be trained with Siamese network having both positive and negative pairs. This model would tell whether Answer entail the question or not.

4. Future Scope

* Current solution is good enough for small dataset and low volume of queries. If database is small and query volume is high, I would go for open source LLMs to save the cost. Paying per query is going to be a real headache in such scenarios. We can fine tune Open models using LoRA in cost efficient and can deploy on A10G machines.
* If database is going to be big, then I would go for different vector DB. Preferred choice is weaviate, because it let us do the hybrid search with BM25 using RRF. Means, we can query not just with vector inner product but also word frequencies (just like tf-idf ngrams). For large volume data, vector do a poor job in retrieval.
* If query volume is to remain very high, using small open source LLM like falcon or MPT model, finetuned with SFT using OASST data with give us equally good results, saving a lot per query money. (0.002$ for 1k openai gpt3.5).
* A bit more work with prompt template can provide much more control and customization over the resulted ouput.
* Given my busy schedule I can’t spend more then few hours on this, I wish I could make it production grade.