Brance Research InternTask

Name:Harshul Surana
Linkedin Profile: linkedin
Date Challenge Received:06-07-23
Date Solution Delivered:14-07-23

1. Problem Statement

- To build (Langchain-based) models that follow the RAG architecture to answer questions based upon the given knowledge document (information about PAN cards). The goal is to maximise the evaluation metric which is semantic similarity between the model output and the desired output (gold standard or eval data).
- Compare the accuracies of different QA chains (qa_chain, RetrievalQA) using **Semantic Similarity**
- Build a chatbot based Langchain(ConversationalRetrievalChain) for facilitating easier and human-like interaction

2. Approach

The approach is to build Langchain models. Different Chains are built and the performance is evaluated using Semantic Similarity.
The chatbot model is also built for facilitating human-like interactions.

3. Solution

a) QA_Chain

1. The document is loaded and divided into 200 tokens long chunks(texts) using CharacterTextSplitter.
2. The texts are embedded.
3. Index is created using FAISS
4. k=3 top most similar documents is the context

b) RetrievalQA_Chain (Prompt included)

1. The document is loaded and divided into 200 tokens long chunks(texts) using CharacterTextSplitter.
2. The texts are embedded.
3. Index is created using Chroma
4. GPT-3.5 Turbo model is used
5. k=4 top most similar documents is the context
6. Prompt is passed along with the queries.

c) ConversationalRetrievalChain

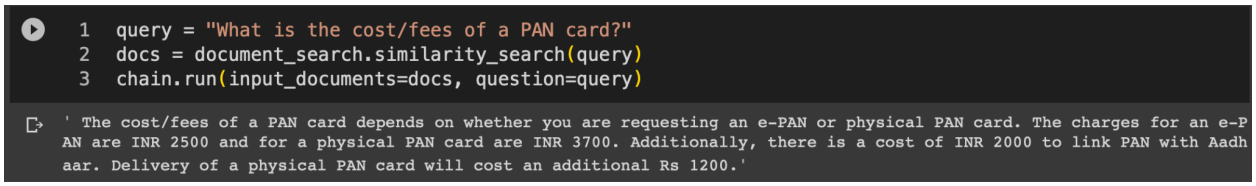
Same as RetrievalQA

4. Results and Evaluation

Cosine Similarity is used as the evaluation metric, using the SBERT

The idea is that the more accurate generated answer will be more semantically similar to to the gold answer(y), and therefore have a higher cosine similarity score. The generated and the gold answer are passed to SBERT to get the respective sentence embeddings, and the cosine similarity is calculated.

While traditional evaluation metrics like exact match is used (SQuAD), it will not be useful in this case because the answers are generative in nature. Instead of penalizing the model for not outputting the exact words, the semantic similarity to the gold answer should give a good indication of the accuracy of the generated answer.



QA_chain example output

RetrievalQA sample output

```
[244]  1  # Build prompt
       2  from langchain.prompts import PromptTemplate
       3  template = """Use the following pieces of context to answer the question at the end.
       4  If you don't know the answer, just say that you don't know, don't try to make up an answer.
       5  Use three sentences maximum and keep the answer as concise as possible.
       6  Always say "thanks for asking!" at the end of the answer.
       7  {context}
       8  Question: {question}
       9  Helpful Answer:"""
      10  QA_CHAIN_PROMPT = PromptTemplate(input_variables=["context", "question"],template=template,)
      11
      12  # Run chain
      13  from langchain.chains import RetrievalQA
      14  #llm = ChatOpenAI(model_name="gpt-3.5-turbo", temperature=0)
      15  qa_chain_prompt = RetrievalQA.from_chain_type(llm,
      16                                                retriever=retriever,
      17                                                chain_type_kwargs={"prompt": QA_CHAIN_PROMPT})
      18
      19  result = qa_chain_prompt({"query": question})
      20  result["result"]
```

RetrievalQA with prompt

```
'The documents required to apply for a new PAN card are as follows:\n\n1. Passport (any country) or OCI card\n2. Passport
size photograph\n3. Overseas address proof with zip code (supporting documents can include an Indian NRO/NRE account stat
ement, overseas bank statement, or utility bill)'
```

```
7s  1  chat_history = [(query, result["answer"])]
    2  query = "what can the supporting documents be?"
    3  result = qa({"question": query, "chat_history": chat_history})
```

```
WARNING:chromadb.db.index.hnswlib:Number of requested results 4 is greater than number of elements in index 1, updating r
```

```
0s  1  result
```

```
{'question': 'what can the supporting documents be?',
 'chat_history': [('What is the cost/fees of a PAN card?',
   'The documents required to apply for a new PAN card are as follows:\n\n1. Passport (any country) or OCI card\n2.
Passport size photograph\n3. Overseas address proof with zip code (supporting documents can include an Indian NRO/NRE
account statement, overseas bank statement, or utility bill)')],
 'answer': 'The following documents can be used as supporting documents for a PAN card application:\n\n1. Passport (Any
Country) / OCI Card\n2. Passport Size Photograph\n3. Overseas address proof with zip code (Supporting documents - Indian
NRO/NRE Account statement or Overseas bank statement or Utility bill)\n4. Copy of Existing PAN card (for PAN card
correction/updation)\n5. Bank Account Statement in the country of residence\n6. NRE Bank Account Statement in India\n7.
Residential Permit\n8. Visa granted and Copy of appointment letter/contract from Indian Company & Certificate (in
original'}
```

ConversationalRetrievalChain example

QA_Chain gets an average cosine similarity score of **76.4%** over 33 samples.

RetrievalChain gets an average cosine similarity score of **81.5%** over 20 samples. (20 chosen because later instances run into problem of exhaustion of openAI API usage rate)

This shows that adding a prompt gives better results.

The Conversational (Chatbot) model is able to answer a user question emanating from an answer to the given query. This is possible because the model keeps track of the chat history. Therefore the user can chat with the system and clarify the doubts that may arise from the generated answer to the question.

4. Future Scope

1. Comparison between different evaluation metrics.
2. Trying out other Indexing techniques like Pinecode.