The notebook aims to showcase the effectiveness of the RAG method by comparing it to pure generative model(the hugging face pipeline from Langchain using the Llama-2-13b model)

First, the notebook explored the use of Pinecone as a vector embedding database for building vector index.

import os

import pinecone

# get API key from app.pinecone.io and environment from console

pinecone.init(

    api\_key=os.environ.get('PINECONE\_API\_KEY') or 'PINECONE\_API\_KEY',

    environment=os.environ.get('PINECONE\_ENVIRONMENT') or 'PINECONE\_ENV'

)

Notice: you need to replace the “PINECONE\_API\_KEY” with an access key that you create in your pinecone account.

Then, we explored this dataset:

'jamescalam/llama-2-arxiv-papers-chunked'

And we modified its contain to make it adaptable to our Pinecone database before upserting(insert or updates if exists).

data = data.to\_pandas()

batch\_size = 32

for i in range(0, len(data), batch\_size):

    i\_end = min(len(data), i+batch\_size)

    batch = data.iloc[i:i\_end]

    ids = [f"{x['doi']}-{x['chunk-id']}" for i, x in batch.iterrows()]

    texts = [x['chunk'] for i, x in batch.iterrows()]

    embeds = embed\_model.embed\_documents(texts)

    # get metadata to store in Pinecone

    metadata = [

        {'text': x['chunk'],

         'source': x['source'],

         'title': x['title']} for i, x in batch.iterrows()

    ]

    # add to Pinecone

    index.upsert(vectors=zip(ids, embeds, metadata))

For the last part, we compared the output/answers of different queries using both RAG pipeline(which uses a retrievalQA chain that extracts relevant contexts related to the query from the Pinecone database already initialized as seen below)

from langchain.vectorstores import Pinecone

text\_field = 'text'  # field in metadata that contains text content

vectorstore = Pinecone(

    index, embed\_model.embed\_query, text\_field

)

query = 'what makes llama 2 special?'

vectorstore.similarity\_search(

    query,  # the search query

    k=3  # returns top 3 most relevant chunks of text

)

Reminder: the similarity metric is the cosine.

After comparing the two methods(only Llama-2 model and Llama-2+RetrievalQA) some queries were not treated conveniently by the generative Llama-2 model, but we got more contextually adequate answer using the Llama-2+RetrievalQA RAG pipeline.

Link to a video related to the work:

<https://www.youtube.com/watch?v=ypzmPwLH_Q4&t=861s>

Embedding transformer: all-MiniLM-L6-v2 (384 dimension embedding vector)

Similarity metric: cosine

Pinecone(vector embedding dataset): creating and querying vector indexes for similarity search.

Datasets: llama-2-arxiv-papers-chunked ( 4838 instances)