import PyPDF2

from sentence\_transformers import SentenceTransformer

import faiss

from langchain.embeddings import FAISS

import openai

from langchain.chains import RetrievalQA

from langchain.vectorstores import FAISS

# 1. Extract text from PDF

def extract\_pdf\_text(pdf\_path):

text\_data = []

with open(pdf\_path, 'rb') as file:

pdf\_reader = PyPDF2.PdfReader(file)

for page in pdf\_reader.pages:

text\_data.append(page.extract\_text())

return text\_data

# 2. Chunking

def chunk\_text(text, chunk\_size=200):

chunks = []

for i in range(0, len(text), chunk\_size):

chunks.append(text[i:i + chunk\_size])

return chunks

# 3. Embedding and Storing

def store\_embeddings(chunks):

model = SentenceTransformer("all-MiniLM-L6-v2")

embeddings = model.encode(chunks)

dimension = embeddings.shape[1]

index = faiss.IndexFlatL2(dimension)

index.add(embeddings)

return index

# 4. Query Handling

def retrieve\_relevant\_chunks(query, index, chunks):

model = SentenceTransformer("all-MiniLM-L6-v2")

query\_embedding = model.encode([query])

D, I = index.search(query\_embedding, k=5) # Retrieve top 5

return [chunks[i] for i in I[0]]

# 5. Response Generation

def generate\_response(chunks, query):

openai.api\_key = "your-openai-api-key"

prompt = f"Using the following context: {chunks}, answer the question: {query}"

response = openai.Completion.create(

engine="text-davinci-003",

prompt=prompt,

max\_tokens=200

)

return response["choices"][0]["text"]

# Example Usage

pdf\_path = "path\_to\_your\_pdf.pdf"

text\_data = extract\_pdf\_text(pdf\_path)

chunks = chunk\_text(text\_data)

index = store\_embeddings(chunks)

query = "What is the unemployment rate for different degrees?"

relevant\_chunks = retrieve\_relevant\_chunks(query, index, chunks)

response = generate\_response(relevant\_chunks, query)

print(response)