# GENAI to assist in mental health- Negative

In this part we will create a large language model(LLM) QnA bot for medical issue. Since training an LLM is a cumbersome task which requires huge time and compute resource, we will use a technique called retrieval augmented generation. But you might be wondering that we already did this in the last chapter. Yes, you are right but this time we will showcase that it is not entirely safe to use LLMs and there are a few things that we should keep in mind before building anything using GenAI.

In this particular task we will use a medical record dataset with all the patient details, their history, basically everything. For this task we make use of the langchain library. This library is very useful since it helps in easy integration of LLMs, vector databases, embeddings and so on thus making it very easy to make whole pipelines of generative AI in a matter of few minutes.

First we load the csv file and then parse it to create a proper structured data including the metadata. We use the MiniLM L6 v2 from the sentence transformer library to generate the embeddings. Text embeddings is nothing but a vector representation of the text which is very popular in NLP and can be used to treat text as numbers thus enabling us the train the models on it and generate statistics.

Next up we create a ChromaDB vectorbase to store these embedding vectors. For this task we are going to use the Falcon LLM model from huggingface. After loading the Falcon model we create our LLM prompt. Prompt is just an initial set of instruction which is used the tell our LLM model how to process the user query, how to analyse it and in what format and how to give the answer required. Finally we integrate everything to create a chain which the user can use with their query to get the desired answer.

In the end we can clearly see that their is no safety or privacy setting in this LLM pipeline and it divulges any patient details or statistic which is asked. This can be a huge breach in privacy and can lead to data leaks.

## CODE:

## Installation

Pip install langchain chromadb sentence-transformers accelerate

## Imports

from langchain.document\_loaders import CSVLoader

from langchain.embeddings.sentence\_transformer import SentenceTransformerEmbeddings

from langchain.vectorstores import Chroma

import os

from huggingface\_hub import notebook\_login

notebook\_login()

from transformers import AutoTokenizer, AutoModelForCausalLM

from transformers import AutoTokenizer, pipeline

from langchain import HuggingFacePipeline

from langchain.chains import RetrievalQA

from langchain.prompts import PromptTemplate

import time

## Data Loading

loader = CSVLoader("./medical\_records.csv", encoding="windows-1252")

documents = loader.load()

## Embeddings and VectorDB

embedding\_function = SentenceTransformerEmbeddings(model\_name="all-MiniLM-L6-v2")

db = Chroma.from\_documents(documents, embedding\_function)

## LLM Model Loading

model\_id = "Rocketknight1/falcon-rw-1b"

tokenizer = AutoTokenizer.from\_pretrained(model\_id)

model = AutoModelForCausalLM.from\_pretrained(model\_id, device\_map='auto')

pipe = pipeline("text-generation", model=model, tokenizer=tokenizer, max\_new\_tokens=128)

llm = HuggingFacePipeline(

pipeline = pipe,

model\_kwargs={"temperature": 0.5, "max\_length": 512}

)

## Prompt Generation

PROMPT\_TEMPLATE = """Answer the question based only on the following context:

{context}

You are allowed to rephrase the answer based on the context.

Question: {question}

"""

PROMPT = PromptTemplate.from\_template(PROMPT\_TEMPLATE)

## Chain Building and prediction

qa\_chain = RetrievalQA.from\_chain\_type(

llm,

retriever=db.as\_retriever(k=2),

chain\_type\_kwargs={"prompt": PROMPT},

return\_source\_documents=True

)

query = "What are the allergies of Sandy Brown?"

result = qa\_chain({"query": query})

print(result['result'].strip())