Task 4: Context-Aware Chatbot Using LangChain or RAG

1. Problem Statement

The goal of this project is to build a context-aware chatbot capable of answering user queries by retrieving relevant knowledge from a large corpus (Wikipedia dataset) and generating responses using a free Hugging Face language model. Traditional chatbots without retrieval often hallucinate or provide inaccurate answers. By integrating a retrieval-augmented generation (RAG) approach, the chatbot can provide grounded, source-based answers while maintaining conversational context.

2. Objective

- Implement a chatbot that combines **document retrieval** and **language model generation**.
- Allow the chatbot to access a large textual dataset (Wikipedia) and find relevant information to answer user queries.
- Provide **sources** along with generated answers for transparency.
- Use **open-source/free tools** like Hugging Face models and LangChain.
- Demonstrate a scalable pipeline for ingesting large datasets and storing embeddings in a vector database (ChromaDB).

3. Dataset Loading & Preprocessing

Dataset

- **Name:** Wikimedia Wikipedia
- **Configuration:** 20231101.en (English dump)
- **Source:** Hugging Face Datasets (datasets library)

Steps

1. **Streaming Load**: Dataset loaded in streaming mode to handle large data efficiently.

```
# 6. Load dataset (streaming)
stream = load_dataset(DATASET_NAME, DATASET_CONFIG, split="train", streaming=True)

batch_docs = []
count = 0
batch_num = 0
```

2. **Sample Limiting**: Only the first 1000 articles are processed for quick experimentation.
3. **Document Construction**: Each article is transformed into a Document object containing:
    - title
    - text
    - source URL metadata
4. **Text Splitting**: Articles are split into smaller chunks (1000 characters with 100 overlap) using RecursiveCharacterTextSplitter to optimize embedding and retrieval.

```
# 5. Splitter
splitter = RecursiveCharacterTextSplitter(chunk_size=CHUNK_SIZE, chunk_overlap=CHUNK_OVERLAP)
```

4. Model Development & Training

Step 1: Embeddings

- Embeddings are used to convert textual chunks into dense vectors for semantic search.
- **Model:** sentence-transformers/all-MiniLM-L6-v2

```
# 3. Embeddings
embeddings = HuggingFaceEmbeddings(model_name="sentence-transformers/all-MiniLM-L6-v2")
```

Step 2: Vector Store (ChromaDB)

- Store document embeddings in **Chroma** vector database for efficient similarity search.

```
# 4. Chroma DB
db = Chroma(persist_directory=PERSIST_DIR, embedding_function=embeddings)
```

- Batched insertion is performed for efficiency (batch size 100, max batch 5000).

Step 3: Language Model

- Free LLM from Hugging Face:
  - **Option 1:** tiiuae/falcon-7b-instruct
  - **Option 2:** mistralai/Mistral-7B-Instruct-v0.2
- LLM is wrapped as a **LangChain pipeline**.

```
# 7. Load a free model from Hugging Face
model_name = "tiiuae/falcon-7b-instruct"  # or "mistralai/Mistral-7B-Instruct-v0.2"
hf_pipeline = pipeline(
    "text-generation",
    model=model_name,
    torch_dtype="auto",
    device_map="auto",
    max_new_tokens=512,
    temperature=0.3
)
```

Step 4: RAG Chain

- Combines **retriever** (Chroma) and **LLM** for context-aware generation.

```
# 8. Build RAG Chain
retriever = db.as_retriever(search_kwargs={"k": 3})
chat_chain = RetrievalQA.from_chain_type(llm=llm, retriever=retriever, return_source_documents=True)
```

5. Evaluation

```
config.json: 1.05k/? [00:00<00:00, 82.2kB/s]
model.safetensors.index.json: 17.7k/? [00:00<00:00, 1.68MB/s]
Fetching 2 files: 100% 2/2 [13:35<00:00, 815.83s/it]
model-00001-of-00002.safetensors: 100% 9.95G/9.95G [13:35<00:00, 57.0MB/s]
model-00002-of-00002.safetensors: 100% 4.48G/4.48G [10:11<00:00, 1.82MB/s]
Loading checkpoint shards: 100% 2/2 [01:01<00:00, 28.86s/it]
generation_config.json: 100% 117/117 [00:00<00:00, 10.7kB/s]
tokenizer_config.json: 1.13k/? [00:00<00:00, 94.7kB/s]
tokenizer.json: 2.73M/? [00:00<00:00, 58.1MB/s]
```

```
You: what is the capital of Pakistan?
Setting `pad_token_id` to `eos_token_id`:11 for open-end generation.

Bot:
 Use the following pieces of context to answer the question at the end. If you don't know the answer, just say that you don't know, don't try to ma

Autonomous Region of China; and the Gilgit-Baltistan territory, Khyber Pakhtunkhwa province and Balochistan province of Pakistan.

Autonomous Region of China; and the Gilgit-Baltistan territory, Khyber Pakhtunkhwa province and Balochistan province of Pakistan.

Afghanistan

Afghanistan, officially the Islamic Emirate of Afghanistan, is a landlocked country located at the crossroads of Central Asia and South Asia. Refer

Question: what is the capital of Pakistan?
Helpful Answer:
The capital of Pakistan is Islamabad.

Sources:
1 https://en.wikipedia.org/wiki/Afghanistan | title: Afghanistan
2 https://en.wikipedia.org/wiki/Afghanistan | title: Afghanistan
3 https://en.wikipedia.org/wiki/Afghanistan | title: Afghanistan

---

You: exit
```