RAG_Chat_Bot

December 16, 2025

[43]:
```
from langchain_community.document_loaders import WebBaseLoader
from langchain_text_splitters import RecursiveCharacterTextSplitter
from langchain_huggingface import HuggingFaceEmbeddings
from langchain_community.vectorstores import Chroma

from langchain_groq import ChatGroq
from langchain_core.runnables import RunnablePassthrough
from langchain_core.output_parsers import StrOutputParser
from langchain import hub
```

[44]:
```
from langchain_community.document_loaders import WebBaseLoader

loader = WebBaseLoader(
    web_paths=["https://artelus.com/"]
)

docs = loader.load()
print(docs[0].page_content[:500])
```

```
Artelus - Empowering Healthcare with Artificial
IntelligenceHomeAboutProductsAwardsPublicationsMediaCareersContactUsing
Artificial Intelligence to prevent Preventable BlindnessBook a Call Now
WHAT IS DRISTiDRISTiDiabetic Retinopathy Screening (DRISTi) (CE Class 1) is an
AI product designed to detect the early presence of Diabetic Retinopathy (DR) in
patients during eye check up screening process instantaneously. By cutting the
cord, and creating a first of its kind, AI on a Chip we have
```

[45]:
```
from langchain.text_splitter import RecursiveCharacterTextSplitter

text_splitter = RecursiveCharacterTextSplitter(
    chunk_size=1000,
    chunk_overlap=200
)

splits = text_splitter.split_documents(docs)
print(len(splits))
```

```
3
```

[46]:
```
from langchain_huggingface import HuggingFaceEmbeddings
from langchain_community.vectorstores import Chroma


embeddings = HuggingFaceEmbeddings(
    model_name="sentence-transformers/all-MiniLM-L6-v2"
)

vectorstore = Chroma.from_documents(
    documents=splits,
    embedding=embeddings,
    persist_directory="./chromaDB",
    collection_name="mental_health"
)

vectorstore.persist()

print("Stored successfully in Chroma")
```

```
Stored successfully in Chroma
```

[47]:
```
retriever = vectorstore.as_retriever()
```

[48]:
```
from langchain_core.prompts import ChatPromptTemplate

prompt1 = ChatPromptTemplate.from_template(
    """
You are an AI assistant answering questions using the provided context.

Use ONLY the information from the context to answer.
If the answer is not present in the context, say "I don't know".

Context:
{context}

Question:
{question}

Instructions:
- Provide a clear, structured, and detailed response.
- Use headings and subheadings where appropriate.
- If multiple products or components are mentioned, explain each one separately.
- Keep the tone professional and informative.

Answer:
"""
)
```

[49]:
```
from langchain import hub

prompt = hub.pull("rlm/rag-prompt")
```

[50]:
```
from langchain_groq import ChatGroq
#from dotenv import load_dotenv
#import os

#load_dotenv()

llm = ChatGroq(
    groq_api_key="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",
    model_name="llama-3.1-8b-instant"
)
print("Testing LLM")
response = llm.invoke("What is Retrieval Augmented Generation?")
print(response.content)
```

```
Testing LLM
Retrieval Augmented Generation (RAG) is a type of artificial intelligence (AI)
model that combines the strengths of two different approaches: retrieval-based
models and generation-based models.

1. **Retrieval-based models**: These models rely on searching a pre-existing
database or knowledge graph to find relevant information. They use techniques
such as information retrieval to retrieve information that matches the input
query or prompt.

2. **Generation-based models**: These models, on the other hand, use machine
learning algorithms to generate new text based on the input prompt or query.
They learn from large datasets and can produce novel text that is coherent and
fluent.

RAG models combine the strengths of both approaches by using the retrieval-based
model to find relevant information and then using the generation-based model to
create new text based on that information. This allows RAG models to produce
high-quality text that is both informed by existing knowledge and creative in
its expression.

RAG models typically work as follows:

- **Retrieval step**: The input query or prompt is used to search a database or
knowledge graph to retrieve relevant information.
- **Ranking step**: The retrieved information is then ranked based on relevance
to the input query or prompt.
- **Generation step**: The top-ranked information is used as input to a
generation-based model, which produces new text based on that information.
```

RAG models have been shown to be effective in a variety of applications, including question answering, text summarization, and conversational dialogue systems. They have the potential to revolutionize the way we interact with language models and access information.

[51]:
```
from langchain_core.runnables import RunnablePassthrough
from langchain_core.output_parsers import StrOutputParser

def format_docs(docs):
    return "\n\n".join(doc.page_content for doc in docs)

rag_chain = (
    {
        "context": retriever | format_docs,
        "question": RunnablePassthrough()
    }
    | prompt
    | llm
    | StrOutputParser()
)
```

[52]:
```
rag_chain.invoke("about Artelus")
```

[52]:
```
'Artelus is a company that empowers healthcare with artificial intelligence.
They offer an AI product called DRISTi, which is a Diabetic Retinopathy
Screening (DRISTi) designed to detect Diabetic Retinopathy in patients. DRISTi
is an offline solution that does not depend on the internet or cloud.'
```

[53]:
```
rag_chain.invoke("Capital of India")
```

[53]:
```
"I don't know the capital of India."
```

[54]:
```
rag_chain.invoke("Mention all the products developed by Artelus")
```

[54]:
```
'The products developed by Artelus include DRISTi, a Diabetic Retinopathy
Screening (DRISTi) AI product, and two AI systems: Fundus AI (DRISTi) and OCT AI
for retinal disease detection and diagnosis. Fundus AI can capture 4 levels of
Diabetic Retinopathy and Diabetic Macular Edema, while OCT AI can identify 7
retinal diseases. Both systems utilize explainable AI for transparency and
informed clinical decisions.'
```