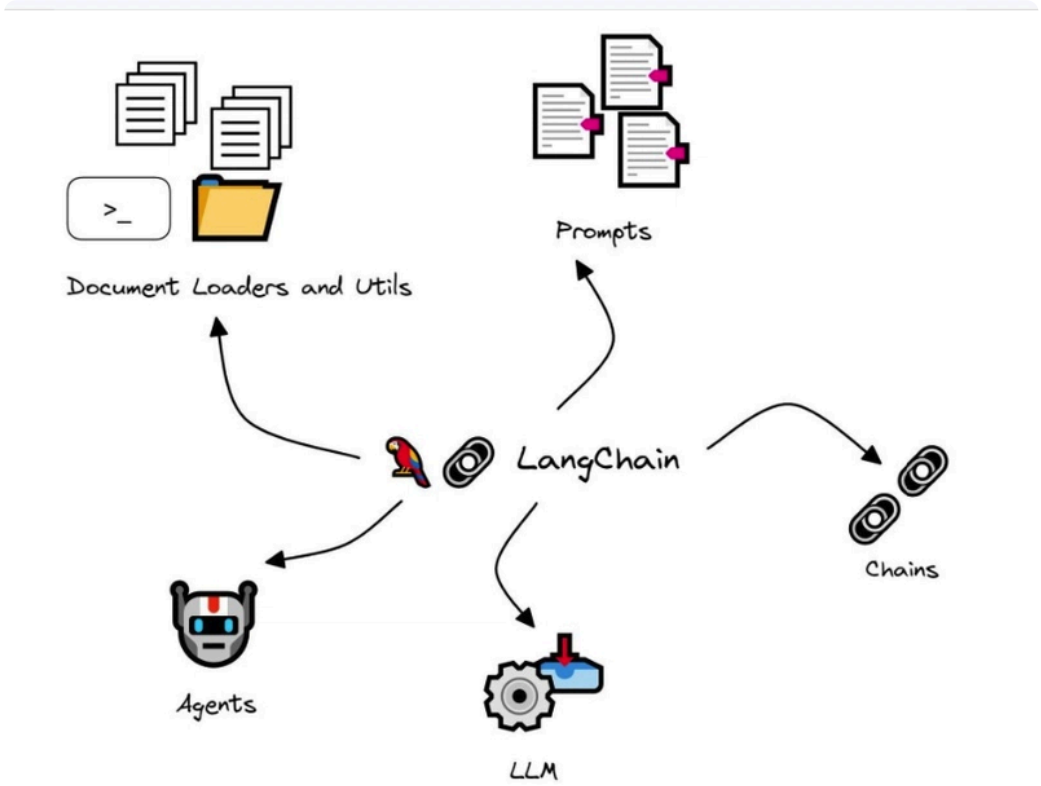
Thiago da Silva Adriano
Posted on Jul 13, 2024

Getting to Know LangChain

Currently, a library has been standing out a lot in the world of Artificial Intelligence. This library is called LangChain.

For those who are not yet familiar with it, this library offers a wide range of functionalities for creating and managing natural language processing pipelines.

Below you have an image demonstrating this library along with its toolkit that it provides us:



Let's get to know each of them below:

- Document Loaders and Utils: This part of LangChain refers to document loaders and utilities. They are responsible for reading, processing and loading documents from various sources (text files, PDFs, databases, etc.) so that they can be used in natural language processing operations.
- Prompts: Prompts are instructions or questions given to the language model to generate responses or perform a specific task. They are fundamental to directing the behavior of the language model.
- Chains: These represent a sequence of steps or operations that are performed together to achieve a specific goal. While using a single LLM may be sufficient

Thiago da Silva Adriano

Microsoft MVP

JOINED
Apr 9, 2018

More from Thiago da Silva Adriano

Imperative Programming

Setting up GraalVM on macOS

Introduction to Event-driven Architecture
#eventdriven

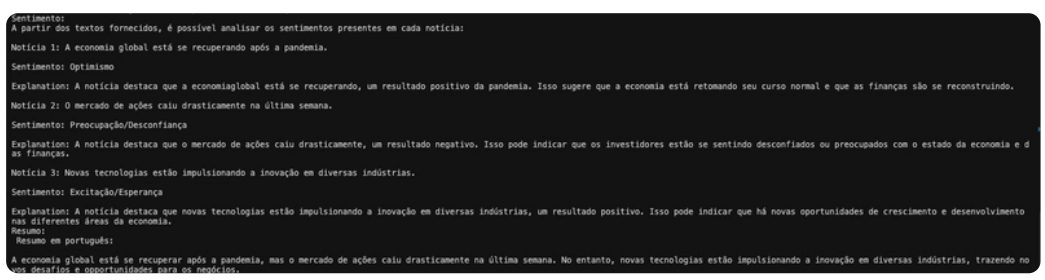
for simpler tasks, LangChain provides a standard interface and some commonly used implementations to chain LLMs together for more complex applications, either with each other or with other specialized modules.
- LLM (Large Language Models): LLMs are large-scale language models, such as GPT-4, BERT, and others, that are used to perform various natural language processing tasks, such as translation, summarization, and text generation.
- Agents: These are components that act autonomously to perform specific tasks based on inputs and language models. They can make decisions and perform actions based on rules or learning.

To make it clearer how this library works, let's look at a practical example of it using Python + LangChain + .txt file where we will analyze the sentiment of some news titles.

```
from langchain_community.llms import Ollama
from langchain.callbacks.manager import CallbackM
from langchain.callbacks.streaming_stdout import
from langchain.prompts import PromptTemplate
from langchain.chains import LLMChain
from langchain_community.document_loaders import
import os

def carregar_documentos(caminho_arquivo):
    loader = TextLoader(caminho_arquivo)
    documentos = loader.load()
    return documentos

def limpar_texto(texto):
    return texto.strip()

llm = Ollama(
    model="llama2",
    num_gpu=0,
    callback_manager=CallbackManager([StreamingSt
)

prompt_sentimento = "Analise o sentimento do segu
prompt_resumo = "Gere um resumo em português para

template_sentimento = PromptTemplate(input_variab
template_resumo = PromptTemplate(input_variables=

chain_sentimento = LLMChain(llm=llm, prompt=templ
chain_resumo = LLMChain(llm=llm, prompt=template_

caminho_arquivo = os.path.join(os.path.dirname(__

if not os.path.exists(caminho_arquivo):
    raise FileNotFoundError(f"O arquivo {caminho_

documentos = carregar_documentos(caminho_arquivo)
```

```
for doc in documentos:
    texto_limpo = limpar_texto(doc.page_content)

    resultado_sentimento = chain_sentimento.run({
    resultado_resumo = chain_resumo.run({"text":

    print(f"Notícia: {texto_limpo}")
    print(f"Sentimento: {resultado_sentimento}")
    print(f"Resumo: {resultado_resumo}")
    print("-" * 50)
```

Running the code we have the following result:



Well, with that I finish this article, I hope you liked it and see you next time guys :)

Top comments (0)