Langchain

Akila Peiris

We will cover

- Introduction
- Langchain enables
- Value props
- Usecases
- Installation

Introduction

- LLMs are very powerful generative tools
- But insufficient as stand-alone
- Needs computation sources
- Needs knowledge sources

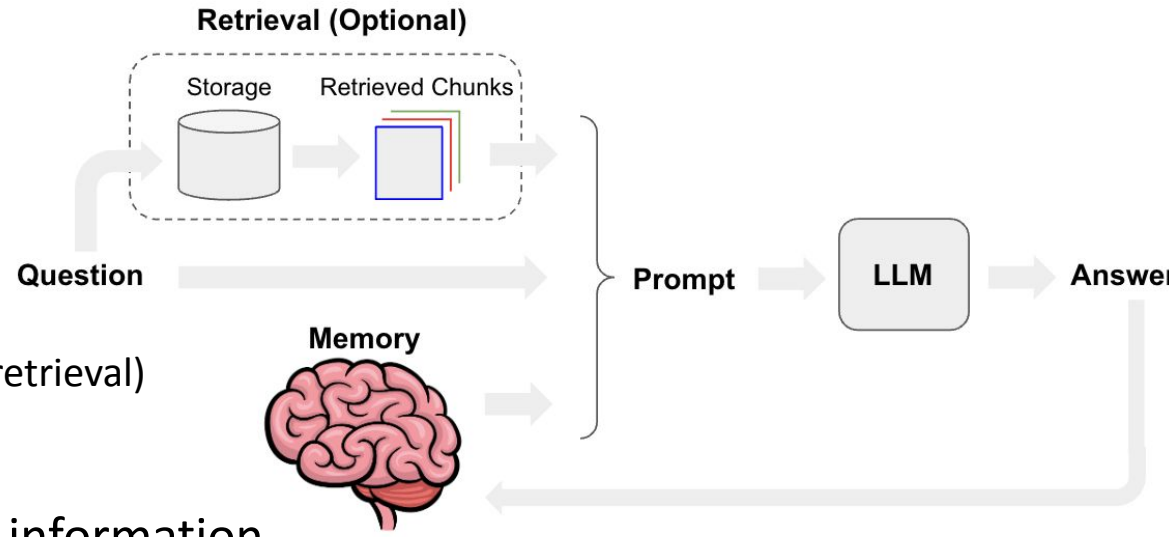
Langchain Enables

- Context-aware applications
  - prompt instructions
  - few shot examples
  - content to ground its response (such as documents)
- Reason
  - answer based on provided context
  - what actions to take

Value Props

- Components
  - abstractions for working with language models
  - + a collection of implementations for each abstraction
  - modular and easy-to-use
- Off-the-shelf chains
  - a structured assembly of components for accomplishing specific higher-level tasks
  - make it easy to get started
- For complex applications, components make it easy to customize existing chains and build new ones.

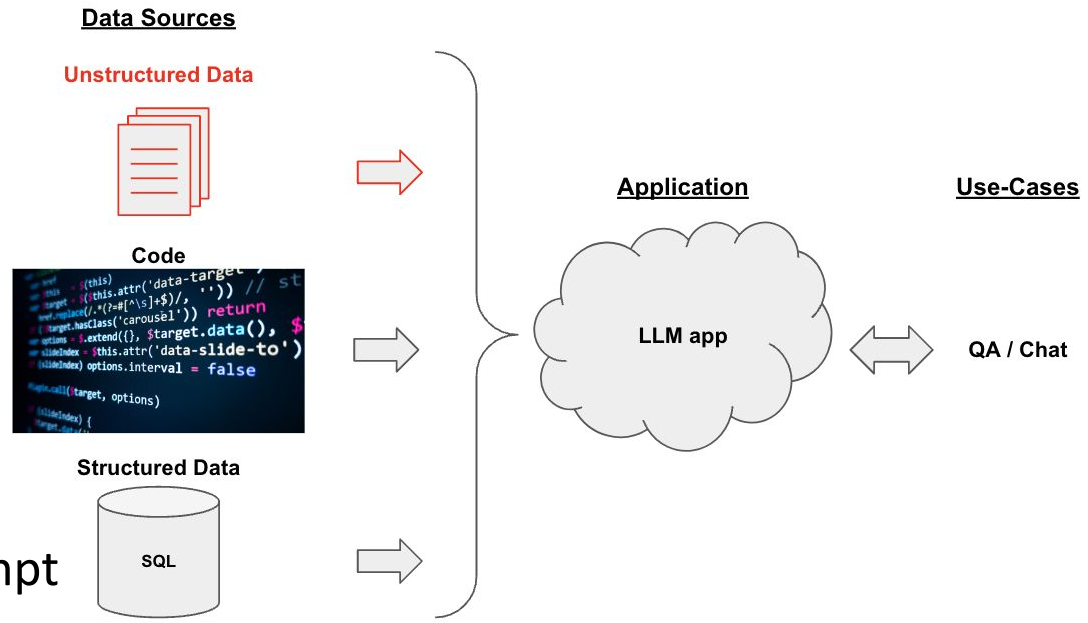
Usecases - Chatbots



- 2 key features
  - long-running conversations (memory)
  - access to information that users want to know about (retrieval)
- Memory – remember past interactions
- Retrieval – provides up-to-date, domain-specific information
- Chat model – the LLM (may be specifically trained for conversations)
- Prompt template – combines default messages, user input, chat history, retrieved context

Usecases - Q&A over specific documents

- ”Read” documents and answer questions
  - Tables, CSVs, Codes
- Load – load the documents
- Split – split document into specific sizes
- Storage – house and embed the splits
- Retrieval – retrieve splits from storage
- Generation – LLM produces an answer using a prompt
- Conversation – multi-turn conversations with memory



Usecases - Agents

- Use an LLM to choose a sequence of actions to take
- LLM is used as a reasoning engine for the actions and their order
- AgentAction dataclass – the action an agent should take
  - tool – name of the tool
  - tool_input – input for the tool
- AgentFinish dataclass – signifies that the agent has finished and should return to the user
- intermediate_steps – previous agent actions and corresponding outputs

Installation

pip install langchain

Or

conda install langchain -c conda-forge

- To install modules needed for the common LLM providers

  pip install langchain[llms]

- To install all modules needed for all integrations

  pip install langchain[all]

Natively supported Chat models

Model	Invoke	Async invoke	Stream	Async stream
AzureChatOpenAI	✅	✅	✅	✅
BedrockChat	✅	❌	✅	❌
ChatAnthropic	✅	✅	✅	✅
ChatAnyscale	✅	✅	✅	✅
ChatGooglePalm	✅	✅	❌	❌
ChatJavelinAIGateway	✅	✅	❌	❌
ChatKonko	✅	❌	❌	❌
ChatLiteLLM	✅	✅	✅	✅
ChatMLflowAIGateway	✅	❌	❌	❌
ChatOllama	✅	❌	✅	❌
ChatOpenAI	✅	✅	✅	✅
ChatVertexAI	✅	✅	✅	❌
ErnieBotChat	✅	❌	❌	❌
JinaChat	✅	✅	✅	✅
MiniMaxChat	✅	✅	❌	❌
PromptLayerChatOpenAI	✅	❌	❌	❌
QianfanChatEndpoint	✅	✅	✅	✅

Natively supported LLMs

Model	Invoke	Async invoke	Stream	Async stream	Batch	Async batch
AI21	✅	❌	❌	❌	❌	❌
AlephAlpha	✅	❌	❌	❌	❌	❌
AmazonAPIGateway	✅	❌	❌	❌	❌	❌
Anthropic	✅	✅	✅	✅	❌	❌
Anyscale	✅	❌	❌	❌	❌	❌
Aviary	✅	❌	❌	❌	❌	❌
AzureMLOnlineEndpoint	✅	❌	❌	❌	❌	❌
AzureOpenAI	✅	✅	✅	✅	✅	✅
Banana	✅	❌	❌	❌	❌	❌
Baseten	✅	❌	❌	❌	❌	❌
Beam	✅	❌	❌	❌	❌	❌
Bedrock	✅	❌	✅	❌	❌	❌
CTransformers	✅	✅	❌	❌	❌	❌
CTranslate2	✅	❌	❌	❌	✅	❌
CerebriumAI	✅	❌	❌	❌	❌	❌

Find the complete list at https://python.langchain.com/docs/integrations/llms/

Resources

- https://python.langchain.com/
- https://github.com/langchain-ai/langchain