Prompt templates

Transcript

1. Prompt templates

00:00 - 00:09

Welcome back! In this video, we'll learn how to prompt LLMs in a more modular and reusable way using prompt templates!

2. Prompt templates

00:09 - 00:27

Prompt templates are a fundamental LangChain component that act as reusable recipes for defining prompts for LLMs. Prompt templates can include instructions or questions, examples for the model to draw on, and any additional context that might help the model complete the task.

3. Prompt templates

00:27 - 01:17

Prompt templates are created using LangChain's PromptTemplate class. We start by creating an example template string, which, in this case, prompts the model to explain a particular concept. The curly braces indicate that we'll use dynamic insertion to input any concept we like into the string later in the code. To convert this string into a prompt template, we pass it to the .from_template() method of the PromptTemplate class. To show how a variable will be inserted, we can call the .invoke() method on the prompt template and pass it a dictionary to insert values where there are input variables. We can see in the output how the concept placeholder has been replaced with the concept we provided in the dictionary. Let's integrate this with an LLM.

4. Integrating PromptTemplate with LLMs

01:17 - 02:00

We start by defining an LLM, as we have before. Then, to integrate the prompt template and model, we'll use LangChain Expression Language, or LCEL. The pipe operator, from LCEL, creates a chain, which is another fundamental LangChain component. Chains connect a series of calls to different components into a sequence. In this case, the user input will be passed into the prompt template to populate it, then the prompt will be inserted into the LLM. To pass an input to this chain, we call the .invoke() method again with the same dictionary as before. We can see that the model responded to the user-inputted question.

5. Chat models

02:00 - 02:59

Chat models support prompting with roles, which allow us to specify a series of messages from these roles. To create a prompt template including chat message roles, we'll use the ChatPromptTemplate class. This allows us to specify a list of tuples containing the roles and messages to pass to the chat model. This list is then passed to the .from_messages() method to create the template. In this example, we can see three roles being used: system, human, and ai. The system role is used to define the model behavior, here indicating that the model should behave like a calculator. The human role is used for providing user inputs, and the ai role is used for defining model responses, which is often used to provide additional examples for the model to learn from. Like with the template strings we created, curly brackets are used to indicate input variables, in this case, math.

6. Integrating ChatPromptTemplate

02:59 - 03:08

We define our model, as before, create our chain again using an LCEL pipe, define a user input, and invoke the chain as before.

7. Let's practice!

03:08 - 03:11

Now it's your turn!