EECS E6893 Big Data Analytics
Generative AI Part 2

Qingcheng Yu, qy2281@columbia.edu
Agenda

• Prompt
• LangChain
• Chatbot
What is Prompt?

• interact with LLMs to accomplish a task
• a user-provided input to which the model is meant to respond
• include specific constraints or requirements like tone, style, or even desired length of the response
Why Prompts are important?

• The quality and relevance of the response generated by the LLM is heavily dependent on the quality of the prompt
• Prompt can solve “hallucination” problem

Was King Renoit real?

King Renoit is not a real historical figure. He is a fictional character from the French epic poem The Song of Roland, written in the 11th century.

Is King Renoit mentioned in the Song of Roland, yes or no?

No. King Renoit is not mentioned in the Song of Roland.

If you ask AI an open-ended question, you received a hallucination.

In fact, King Renoit was never mentioned in the Song of Roland.

But when you asked it to respond with only "yes" or "no," it corrected itself.

https://zapier.com/blog/ai-hallucinations/
Three ways of prompting

• Zero-Shot Prompting
• Few-Shot Prompting
• Chain-of-thought Prompting
Zero-Shot Prompting

• zero-shot prompting means providing a prompt that is not part of the training data to the model
• the model can generate a result that you desire

https://www.promptingguide.ai/techniques/fewshot
Few-Shot Prompting

- give some examples in the prompt
- few-shot prompt enables the model to learn without training
- model generates response based on format of given examples
Chain-of-thought Prompting

• enable complex reasoning capabilities through intermediate reasoning steps

---

**Standard Prompting**

**Model Input**

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

**Model Output**

A: The answer is 27. ×

---

**Chain-of-Thought Prompting**

**Model Input**

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

**Model Output**

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The answer is 9. ✓
What is LangChain?

- a framework for developing applications powered by language models
- connect a language model to sources of context (prompt instructions, few shot examples)
- simplifies the process of creating generative AI application interfaces
LangChain prompt templates

• LangChain provides a set of default prompt templates
• Create custom prompt is also available
• To create a custom string prompt template, there are two requirements:
  • It has an input variables attribute
  • It defines a format method that takes in keyword arguments corresponding to the expected input variables and returns the formatted prompt
Custom Prompt

```python
from langchain.prompts import StringPromptTemplate
from pydantic import BaseModel, validator

PRIMP = """
Given the function name and source code, generate an English language explanation of the function.
Function Name: (function_name)
Source Code:
(source_code)
Explanation:
"""

class FunctionExplainerPromptTemplate(StringPromptTemplate, BaseModel):
    """A custom prompt template that takes in the function name as input, and formats the prompt template to provide
""
    @validator("input_variables")
    def validate_input_variables(cls, v):
        """Validate that the input variables are correct."
        if len(v) != 1 or "function_name" not in v:
            raise ValueError("function_name must be the only input variable.")
        return v

def _prompt_type(self):
    return "function-explainer"
```

Template Prompt

```python
from langchain.prompts import PromptTemplate

prompt_template = PromptTemplate.from_template(
    "Tell me a {adjective} joke about {content}.
"
)

prompt_template.format(adjective="funny", content="chickens")
```
Chatbot

Retrieval (Optional)

Storage → Retrieved Chunks

Question → Prompt

LLM → Answer

Memory
Libraries for building chatbot

- `pip install -Uqqq pip --progress-bar off`
- `pip install -qqq torch==2.0.1 --progress-bar off`
- `pip install -qqq transformers==4.33.2 --progress-bar off`
- `pip install -qqq langchain==0.0.299 --progress-bar off`
- `pip install -qqq chromadb==0.4.10 --progress-bar off`
- `pip install -qqq xformers==0.0.21 --progress-bar off`
- `pip install -qqq sentence_transformers==2.2.2 --progress-bar off`
- `pip install -qqq tokenizers==0.14.0 --progress-bar off`
- `pip install -qqq optimum==1.13.1 --progress-bar off`
- `pip install -qqq auto-gptq==0.4.2 --extra-index-url https://huggingface.github.io/autogptq-index/whl/cu118/ --progress-bar off`
- `pip install -qqq unstructured==0.10.16 --progress-bar off`
• Download model from your HuggingFace hub

```python
import torch
from langchain import HuggingFacePipeline
from transformers import AutoModelForCausalLM, AutoTokenizer, GenerationConfig, pipeline

MODEL_NAME = "JackyYu/llama-2-7b-chat-finetune"

tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, use_fast=True)

model = AutoModelForCausalLM.from_pretrained(
    MODEL_NAME, torch_dtype=torch.float16, trust_remote_code=True, device_map="cuda"
)

# Loading checkpoint shards: 100% 2/2 [01:01:00.00, 28.29s/it]

# Next line contains warnings. 

pipe = pipeline(
    "text-generation",
    model=model,
    tokenizer=tokenizer,
    generation_config=GenerationConfig(
        max_new_tokens=1024,
        temperature=0.0001,
        top_p=0.95,
        do_sample=True,
        repetition_penalty=1.15,
    ),
)

text_pipeline = pipepline(
    "text-generation",
    model=model,
    tokenizer=tokenizer,
    generation_config=GenerationConfig(
)

llm = HuggingFacePipeline(pipeline=text_pipeline, model_kwargs={"temperature": 0})
```

• Change the configuration of model and build the llm pipeline
• **Test Ilm pipeline**

```python
result = Ilm(
    "Explain the difference between ChatGPT and open source LLMs in a couple of lines."
)
print(result)
```

ChatGPT is an AI-powered chatbot developed by Meta that uses natural language processing (NLP) to generate human-like responses to user queries, while open source LLMs are machine learn

• **Example of prompt template**

```python
from langchain import PromptTemplate

template = """"";
<><[INST]><SYS>
Act as a Machine Learning engineer who is teaching high school students.
</SYS>

{text} [/INST]
"""

prompt = PromptTemplate(
    input_variables=['text'],
    template=template,
)```
Functions you need for chatbot

```python
from langchain.prompts.chat import (ChatPromptTemplate,
                                        HumanMessagePromptTemplate,
                                        SystemMessagePromptTemplate,
                                        )
from langchain.schema import AIMessage, HumanMessage
```

• Chatbot looks like:

AI: The most powerful AI model is the Transformer, which was introduced in 2017 by Google researchers. It has been used for a wide range of applications such as language translation, t
Human: Can you explain it to me like I’m five years old?
AI: Sure thing! Imagine you have a big box full of toys. You want to find one specific toy inside the box, but you don’t know where it is or what color it is. That’s kinda like how con
Human: Wow, thank you for explaining it to me like that!
AI: Of course! I am always here to help you learn about artificial intelligence. Do you have any questions about anything else related to this topic?
References

- https://www.promptingguide.ai/techniques/fewshot
- https://lh6.googleusercontent.com/aKjm1ycrNyqvETuDGvbNS0M1aiNN-m3ZHs3RVz--lNY-zxwbycf6MuB-P4JDb8yDx-urlJ1LbUkJpZQrLH4LGHeN0cy2o0IDsI9qqPMMYQi5Wg7tRYXVhGdZ-z1xRh-Ln1R9fVDHxsEmqMUfa4eSjCo
- https://zapier.com/blog/ai-hallucinations/
- https://www.promptingguide.ai/techniques/cot
- https://www.langchain.com/