# 2.2 提示

新的大语言模型的编程方式是通过提示（prompt）进行的。“提示”是 Langchain 与大模型交互的最重要组件。Langchain 中的提示输入通常由多个部分构成。LangChain 提供了类和函数，以方便构造和操作提示。

#### 提示的定义

什么是提示？提示指的是模型的输入,　我们发给大语言模型的内容。

在 Langchain 中,提示是让模型在输入的提示文本**条件限制**下生成文本的过程。这些模型通常是继续预测下一个标记,通常是下一个单词,然后通过这种方式组装文本输出。

#### 你的第一个提示

在编程世界里，我们常说”Hello World”是每个程序员的第一步。现在，对于使用大语言模型编程的我们，“你好”, 一个emoj 表情或者任何一句话，也是我们的第一步。因为无论我们的任务多么复杂，一切都始于一个prompt。

还记得你第一次使用一个聊天机器人时，你敲打键盘的第一句话是什么吗？是不是 “你好” ？

|  |
| --- |
| 图 2-1 |

图 2-1

事实上，这个看似简单的“你好”，在编程世界里就像完成了“Hello World”的编程那样具有重要意义。这是因为在这个场景中，“你好”是一个提示。

你的第一个提示代表你已进入 AI 编程的美丽新世界。

提示对大语言模型的输出结果有很大的影响,以至于这导致了一个全新的领域,人们现在称之为提示工程。

#### 提示的演化

早期的提示非常简单,你可能会问一些简单的问题,比如乾隆是什么时候皇帝? 希望大型语言模型在它之前数据集学到足够的知识来回答这个问题,并生成一个输出结果。

随着模型的不断改进,提示从这些简单的形式转变为更加复杂的形式,现在我们看到人们给出了包含整个上下文和各种信息的提示。还有少数示例提示模板（Few-shot）,即我们不仅仅告诉提示一些内容,还会给它一些示例,然后让它生成一个新的答案或示例。

从GPT论文中,我们可以看到其中有不同类型的提示,论文中包含了许多不同的提示。

随着技术的不断进步和人工智能领域的深入研究，如今的提示已经越来越复杂和工程化。尤其是在参数化模型输入和样本示例这两个方面，都做出了巨大的努力来提高提示的质量，以实现更好地控制大型语言模型的目标。

#### 模型对提示的影响

我们现在进入了模型开发的白热阶段，也就是“千模大战”, 未来可能会有更多模型提供商加入,那其他的模型对提示会有什么影响？

LangChain 构造和操作的提示适用于其他模型，提示对于其他模型的影响，取决于模型的规模和质量。如果你想使用免费的Hugging Face模型中的一个，我们可以用LangChain 做相同的事情。

以下是使用 Google T5-Flan-XL 模型的代码：

from langchain.llms import HuggingFaceHub

在这里，我只是设置了一个Hugging Face版本的Google T5-Flan-XL模型。这不是最大的模型，最大的模型是XXL，但是这个模型的规模肯定没有 OpenAI 的text-davinci-003模型大，不过你会看到我们仍然可以得到一些连贯的回应。

llm\_hf = HuggingFaceHub(  
 repo\_id="google/flan-t5-xl",  
 model\_kwargs={"temperature":0.9 }  
)

你会注意到，这个回应不同，如果我们再次运行它，我们仍然得到相同的东西，但每个模型都给出了不同的回应，但基本上做的事情都是一样的。

text = "Why did the chicken cross the road?"  
  
print(llm\_hf(text))

It was hungry.

#### Langchain 里的提示

Langchain对于你开发应用的提示功能非常重要。通过它，你可以更有效地指导你的应用来实现预期的功能。而且，一旦模板构建好，你只需要提供关键的信息。让我们通过一个具体的实例来说明这一点。

假设你正在开发一个专门为电影爱好者设计的聊天机器人，该机器人的目标是提供详细的电影信息。此时，Langchain的提示功能就可以派上用场。你可以建立一个模板，像这样：

* 提示：“““请告诉我《{电影名}》的导演是谁？”“”
* 提示：“““《{电影名}》的主要演员有谁？”“”
* 提示：“““《{电影名}》的剧情简介是什么？”“”
* 提示：“““《{电影名}》的评分是多少？”“”

在模板建立好之后，用户只输入电影名，比如《阿凡达》或者《泰坦尼克号》。此时，模型会接收到Langchain提供的结构化提示输入，如“请告诉我《阿凡达》的导演是谁？”或“《泰坦尼克号》的主要演员有谁？”等。

然后，模型会根据这些提示，生成相应的输出。由于Langchain的提示非常具体和结构化，模型生成的答案质量通常较高，符合用户的预期。你可以将更多的精力集中在用户和业务上，而不必担心如何指导模型生成满足用户需求的答案。因为一旦模板建立好，Langchain的提示功能就会帮你做这件事。

在 Langchain 中,所有这些都由提示模板来处理,因此了解提示的组成部分非常重要,对于你使用 Langchain 所做的一切至关重要。

Langchain 里的提示模板主要2个功能是：

#### 1. 参数化模型输入

参数化模型输入是通过将模型输入转化为可动态的参数形式，从而实现模型输入的灵活调整。具体来说，它允许我们根据特定的需求和环境条件，来动态地调整模型的输入。这样一来，我们可以根据特定的任务需求，通过调整输入参数，使模型生成满足我们需求的输出。例如，我们正在开发一个聊天机器人，并希望给它一个具有个性的名字，比如”小智”或”问答达人”等。我们可以将机器人的名字作为输入参数，通过Langchain 里的提示模板构建一个动态的提示。“小智”或”问答达人”就是我们输入的参数，Langchain 里的提示模板接受这个参数，构造成一个完整的提示输入给大语言模型,。

这是最基本的示例，但它使我们能够动态更改带有用户输入的提示。

这个代码中,{botname} 是要被替换的占位符。

# Import prompt and define PromptTemplate  
from langchain import OpenAI  
from langchain import PromptTemplate  
  
template = """  
从现在开始，你的名字是：{botname}。每次打招呼的时候，请回应：“你好，我是{botname}聊天机器人，有什么可以帮到你？”。  
"""  
  
prompt = PromptTemplate(  
 input\_variables=["botname"],  
 template=template,  
)  
# Run LLM with PromptTemplate  
llm = OpenAI(openai\_api\_key="YOUR\_OPENAI\_API\_KEY")  
llm(prompt.format(botname=""))

当我们传入 botname="小智" 时, PromptTemplate 方法,将 “小智” 注入到提示中，这个聊天机器人就以“小智”的身份回应你的问题。

#### 2. 动态选择包含在提示中的示例

“示例提示模板”（又称为 “少数示例学习” 或 “Few-shot learning”）已经成为当前提升大型语言模型性能的一个重要策略。让我们通过一个具体的例子来了解一下这种方法的工作原理。

假设我们正在开发一个聊天机器人，该机器人需要回答用户关于天气的各种问题。这时，我们可以使用示例提示模板或者少数示例学习的方法来训练和优化我们的机器人。

首先，我们可以为聊天机器人提供一些关于如何回答天气相关问题的示例。这些示例包括如下内容：

User：“今天北京的天气怎么样？” Chatbot：“我不确定，让我查一下。今天北京的天气预报是晴朗。”

User：“明天上海会下雨吗？” Chatbot：“让我为你查看明天上海的天气预报。不，明天上海不会下雨。”

这些示例将帮助机器人理解用户可能会提出的天气相关问题，以及如何回答这些问题。然后，当用户实际提问时，机器人 (Chatbot) 就可以参考这些已经学习过的示例，来生成合适的回答。这就是少数示例学习的工作原理。

通过使用这种方法，我们可以大大提高聊天机器人的性能。因为它允许我们以一种非常直接而灵活的方式，为机器人提供具体的指导。这使得我们的机器人不仅可以更准确地回答用户的问题，而且还可以更好地适应各种不同的环境和情境。因此，示例提示模板已经成为当前提升大型语言模型性能的一个重要工具。

在实际开发中，我们通过将这两种方法结合起来，我们可以更好地控制大型语言模型的表现。这是因为我们不仅可以根据需求调整模型的输入，还可以选择最适合的训练数据作为输入。这就提供了一种强大的工具，可以帮助我们实现更精细、更有目标性的模型控制。