Prompt Engineering for Language Models

A step-by-step guide to tailoring the use of language models (LMs)

A common use case for LMs such as with GPT is question answering. For this purpose, it is often used for asking a question with one prompt. However, in this guide, we will show how LMs can be better tailored to more complex tasks by using multiple prompts.

As an example of a complex task, we will show how an LM can be used to control a chatbot. As an exercise, you will then implement such a chatbot.

Part 1: Manual Use of a LM

Using ChatGPT, you will try to use multiple prompts to obtain a specific behaviour of your choice. The following table lists the main components of a conversation between you and ChatGPT in the left column, with an example in the context of a health coach in the right column. The resulting health coach will be used to generate messages to be sent to patients. Patient responses will then be processed using ChatGPT, and to generate follow-up messages to the patient.

The overall structure of your conversation with ChatGPT could be as follows.

Element of Conversation with ChatGPT	Example
1. Role of GPT	You are a health coach supporting long-term treatments of …
2. Additional role information	Health coaching is the use of evidence-based skillful conversation …
3. Context of subsequent use	Daniel is 50 years old with a BMI of 27. His therapy plan includes …
4. Request for message to end-user	Create a message to be sent to Daniel. The goal of the message is to find out about …

1. Role Definition

As a first prompt, you specify the role of GPT. For example, you may specify the role of a health coach or an HR recruiter. As a starting point, you may find interesting prompts here:

ChatGPT Prompt Templates[1]

2. Additional Role Information

You may add multiple prompts further describing the role or methods to be applied. For example, in the case of a health coach you may provide a description of a coaching method. In the case of HR recruiting, you may provide a job description. You may start looking for such additional information on the web using Google or WikiPedia. With this part, you can increase the expertise and specificity of your GPT application.

3. Context of Subsequent Use

Now, you specify the context of the interaction with a specific end-user. For example, if the idea is to use GPT for health coaching, you may provide EMR data about the patient. For HR recruiting, you

[1] https://ignacio-velasquez.notion.site/2-500-ChatGPT-Prompt-Templates-d9541e901b2b4e8f800e819bdc0256da

may specify the CV of the candidate. This is the part where you can personalise the end-user experience.

4. Request for Messages to End-User

Tell GPT to generate a message to be sent to the end-user, e.g., a message to welcome the user. You may additionally specify the goal of this welcome message. For example, the goal could be to welcome the user, present the chatbot and inform the user about the goal of the conversation.

Part 1: Exercise

- Register at https://platform.openai.com/
- Navigate to https://platform.openai.com/playground and set the mode to "Chat"
- Follow the guide above to implement your own use case
- Use the conversation managed with GPT to interact with an end-user

Note that you will send the messages to the end-user and receive the messages from the end-user using a separate communication channel such as WhatsApp or Microsoft Teams. As a result, you will obtain two conversations: one between you and GPT, another one between you and the end-user.

Output

The set of prompts (e.g., role prompt, additional role information, context of subsequent use) that worked best.

Part 2: Create Your Custom GPT

Using ChatGPT+ (the paid subscription), you will create your own custom GPT for a specific use case of your choice. We provide you with a list of example use cases, but feel free to create a custom GPT for one of your own use cases:

- Project management assistant for reports and risk assessment
- Persona creator for user stories
- Text simplifier to support understanding complex texts
- Event planner

1: Define Goal of your Assistant

Define which tasks your assistant should focus on. You are creating a specialist hat is better than GPT for this task.

2: Identify persona

Identify the key elements and characteristics of your Assistant's persona. Is it going to be pro-active and ask for additional information? What education level do they have? What gender? Do they have interests?

3: Test and Refine

Talk to your assistant and check whether it exhibits the expected behaviour. You can edit the prompts if you feel like it needs more tweaking. The advantage of your own GPT is that you can refine the prompt anytime.

Part 2 Exercise

- Login at https://chat.openai.com/ (you need a paid subscription)
- Navigate to Explore GPTs
- Click on + Create
- Start answering the questions of the GPT builder. Use your previously identified goal and persona too answer the questions. Answer the questions of the GPT builder as precisely as possible.

Note: You do not need to create an English-speaking Assistant. You can also create a German-speaking assistant or any other language you are proficient in.

Output

The link to your refined custom GPT.

Part 3: Programmatic Use of GPT

We will now begin the first steps of implementing a fully functional chatbot available online. There are many tutorials and documentations available online, such as the openai api reference[2]. We strongly encourage you to use ChatGPT as well!

Preparation

You need to set up the openai library in your Python environment. You also need to create and obtain an API key.

Python

The following code shows how to use the API to obtain a response to a request.

```
import openai
openai.api_key = "..."

chat = openai.ChatCompletion.create(
    model="gpt-4",
    messages={
        "role": "user",
        "content": "Hello GPT."
    }
)
print(chat.choices[0].message.content)
```

Part 3: Exercise

- Use "pip install openai" to setup the openai library in your Python environment
- Create a Python Notebook and copy the code above
- Navigate to https://platform.openai.com/account/api-keys, create a new secret key and insert it in your code (openai.api_key = "...")
- Possibly adapt the user prompt from "Hello GPT." to anything you like
- Execute the code and make sure you obtain a response

Output

The notebook.

[2] https://platform.openai.com/docs/api-reference