What Is Prompt Engineering and Why Do We Care?

Introduction

In this lab, we'll introduce the concept of **prompt engineering** and how it can help you leverage what AI offers. Before jumping in, we should define Prompt Engineering, why it matters, and how we got to this point in the AI landscape.

If you've ever used a chatbot or virtual assistant online, you may have mixed feelings about the experience. Most chatbots are designed to respond to a predefined set of queries programmatically. While AI allows some flexibility in how we phrase our messages, these chatbots are essentially programmed with specific responses.

A well-designed chatbot can still provide a pleasant user experience, but often lacks a 'wow' or 'magic' factor. A user interacting with a good chatbot is unlikely to be amazed by its AI capabilities. On the other hand, poor chatbots—of which there are many—can be frustrating, often leading users to angrily type 'AGENT!!!' after just a few exchanges.

A first taste of AI

It's no surprise then that the world took serious notice when OpenAI's ChatGPT, Google Bard, watsonx, and similar AI chatbots became available. ChatGPT, for example, gained a million users within just five days of its launch in November 2022. For comparison, some of the fastest-growing apps in the world took several weeks to reach that milestone.

For the first time, conversational generative AI chatbots truly amazed us. The conversations felt natural, and the answers were both plausible and valuable.

Behind the scenes, complex mathematics and vast amounts of data are used to train large language models (LLMs), which power conversational AI. Fortunately, you don't need to worry about the intricate implementation details to take full advantage of their capabilities.

The experience is remarkably similar to engaging with a brilliant individual possessing vast knowledge equivalent to terabytes of data.

People have started using such chatbots to boost productivity, communicate more clearly, quickly summarize large amounts of data, and assist with decision-making.

LLMs are powerful technologies that have the potential to transform industries and the world at large.

So, we have this incredible tool—how can we make the most of it?

It all starts with user input, our **prompt**. A simple text box is all we have to control this powerful tool. The quality of our prompt significantly impacts the results (the output). As the saying goes, 'garbage in, garbage out.' If we provide a vague or poorly structured prompt, the AI won't have much to work with, limiting its ability to deliver valuable responses and showcase its full potential.

What is prompt engineering?

For example, let's ask one of the aforementioned LLM-based AIs, 'What is Prompt Engineering?'

This is the result I got:

> Prompt engineering refers to the process of designing and refining prompts or instructions given to a language model, such as GPT-3.5, to generate desired outputs. It involves carefully crafting the input text provided to the model to elicit the desired response or behavior.
>
> Prompt engineering can be used to fine-tune the output of language models by specifying explicit instructions, adding context, or using specific formatting techniques to guide the model's generation. By tweaking the prompts, developers and users can influence the output to be more accurate, coherent, or aligned with a specific goal.
>
> Effective prompt engineering requires understanding the underlying language model and its capabilities. It involves experimenting with different prompts, iterating on them, and analyzing the model's responses to achieve the desired results. Prompt engineering is particularly important Prompt engineering is crucial when working with large language models like GPT-3.5, as their responses can occasionally be unpredictable or need extra context to produce the desired results.
>
> Overall, prompt engineering is a crucial aspect of working with language models to ensure they produce accurate and meaningful responses that align with the user's intentions or requirements.

Not bad, eh? This AI technology might be quite useful. In this course, you'll learn everything you need to know to optimize the results you get from conversational AI, regardless of which specific LLM you choose to use.

Zero-shot, one-shot, and few-shot prompting

In the realm of large language models (LLMs) like GPT-3.5, you may come across terms such as **zero-shot**, **one-shot**, and **few-shot** prompting. These terms describe different ways of instructing or prompting the model.

In a **zero-shot** scenario, the model is given a task without any prior examples. It must determine what to do based solely on the prompt and its training.

In the next couple of labs, we'll refer to naive or standard prompting a few times. This refers to the type of prompt used by someone unfamiliar with LLM-based AI. They'd ask their question without providing any examples for the model.

For example, when we asked, 'What is Prompt Engineering?', we used a zero-shot prompt.

In contrast, a **one-shot** prompt provides the model with a single example, while a **few-shot** prompt includes multiple examples to guide its output. These techniques help shape the model's responses, especially when it hasn't been explicitly trained for a particular task. Users can steer the model toward a specific output or format by offering examples.

Chain-of-thought (CoT), a more advanced prompting technique explored later in the course, is an example of **one-shot** prompting (or **few-shot** prompting, depending on how we phrase our prompt).

Author

Antonio Cangiano