Prompt Engineering Exercises

Alex van Vorstenbosch

2026-02-01

Google Colab

You can make these exercises using the general google colab notebook. Please take a few minutes to read through the notebook to familiarize yourself with the environment.

Prompting Cheatsheet

Check out the prompting cheatsheet: **Cheatsheet.pdf**!
It can be found in the course material under **./slides/**.

It contains some example prompts and inspiration for all kinds of prompting tasks. Read it before starting on the assingments below.

Prompting assignments

Below you will find some assingments to get some experience with prompting:

1 Exercise: Create a Basic Prompt

Using the general prompting template provided in the slides, create a prompt that asks the language model to act a personal tutor for you. Make the model explain to you what a transformer is. Specify the:

- ROLE,

- TASK_DESCRIPTION,
- TASK_SPECIFICATION,
- and FORMAT_OUTPUT.

1.1 Exercise: System prompt vs User prompt

It is good practice to put the general part of the prompt in the `system prompt` (aslo know as `developer prompt` for some models), and the specific part in the `user prompt`. `System prompts` explain the current context within which the model should generate its response, before any interaction with the user has occured. This makes most sense when you are hosting a model with external users who interact with a chatbot, but it is also good practice for your own use. Most LLMs, but not all, are trained in such a way that they strongly adhere to the system prompt, and it is not easy for users to override this. This is in contrast to specifying the RTF framework in the `user prompt`, where it is easy to override by the user.

This is also usefull when you are not dealing with external interaction, as it makes your systems prompts highly reusable.

```
messages = [
    {"role": "system", "content": """
    ROLE: You are a professional chef in a restaurant
    TASK: Based on the ingredients provided, give suggestions for dinner dishes that can be made.
    SPECIFICATION: Only suggest healhty meals that can be made in under 30 minutes. Suggest 3 distinct dishes.
                    Don't include any dishes that contain lactose.
    OUTPUT_FORMAT: Provide the name of the dish, the ingredients, and a short summary of the preparation.
    """},
    {"role": "user", "content": """
    Hi! I have the following ingredients:
    - 4 chicken breast
    - 2 bell peppers
    - 3 onions
    - Garlic cloves
    - 1 can of chickpeas
    - 1 can of diced tomatoes
    - 2 can of coconut milk
    - 1 pack of brown rice
    - a cupboard full of many different types of spices
    """}
]
```

1.2 Try some variations

Try experimenting with some variations of this yourself. Change the role, task, and output format. How does the model respond to these changes?

Answer:
Example chat

1.3 Experiment some more

Pick 2 of the prompts from the cheatsheet and experiment with them. Adjust the wording if needed and use them for a task of your own interest. If you have no inspiration, you can try these:

- Adjust the **Formatting** template, and have your LLM turn this information into a table: *The Ipsos survey explored the impact respondents believe AI will have on various aspects of their lives, such as health and entertainment. On topics like time management and entertainment, the majority viewed AI positively. For instance, 54% of global respondents agree that AI will improve the efficiency of their tasks, and 51% believe AI will enhance entertainment options like TV, movies, music, and books. However, skepticism was more prominent in other areas. Only 39% feel AI will benefit their health, and 37% think it will improve their job. Only 34% anticipate AI will boost the economy, and just 32% believe it will enhance the job market.*
- use the **Brainstorming** assistant template and adjust it: You want to organize an event in your company to promote the possibilities of LLMs. Come up with a list of possibilities for your event!

2 Translation

Below you'll find an e-mail in Italian.

> Ciao,
>
> Sono estremamente arrabbiato! Dove diamine è il tuo contributo per il progetto? La scadenza era venerdì scorso e non ho ancora visto niente da parte tua. Grazie mille per il disordine che hai creato! Adesso dobbiamo estendere la scadenza e il cliente sarà furioso con noi. Questo ci costerà un sacco di soldi, magari non per questo progetto, ma sicuramente per quelli futuri. Cosa hai da dire in tua difesa?! Ho già avvisato il capo di questa situazione, quindi voglio che tu informi il cliente al posto nostro.
>
> X

2.1 Translate the language

Have your LLM translate this to English. What does it say?

2.2 'Translate' the tone of the email

This mail does not read very professional. Please make it more professional sounding, while keeping the original message clear.

2.3 'Translate' the context of the email in canvas

Apperently there was a misscommunication, and the deadline is next Friday. Have your LLM change the email to one making sure we make next Friday's deadline. Make sure the email is firm yet constructive.

3 Exercise: Spelling-problems

1. Ask the model the reverse the word: “arbeidsongeschiktheidsverzekeringsformulier”.
2. Ask the model to reverse the word and spell it out letter-by-letter. What do you see? Why do you think this happens. Use the retry button to see if the model gives a different answer the second time. This feels like a trivial task, but it is far from it. Can you think of why?

3.1 A closer look

Use the functions below to look at the tokens of the word. What do you see?

```
# This returns a list of token IDs
tokens = llm.tokenize("arbeidsongeschiktheidsverzekeringsformulier".encode("UTF8"))
print("Token IDs:", tokens)

# Convert the token IDS back to the corresponding strings
decoded_tokens = [llm.detokenize([token]) for token in tokens]
print("Decoded tokens:", decoded_tokens)
```

3.2 Self-reflection

Ask the model whether the answer it gave was correct, what happens? Why do you think this is the case?

4 Comparing reasoning LLMs to regular LLMs

In this exercise we will compare the performance of LLMs developed for reasoning to regular ones. At the time of writing there are no true reasoning LLMs available open weights, that we can also run on colab. However, there are some distilled models that were trained with supervised finetuning instead of Reinforcement Learning, that can still demonstrate the difference in reasoning capabilities.

This demonstration is a little difficult as the models are stochastic: you cannot fully predict the answer you will get. The reasoning model is made specifically for reasoning task. You will see that it will work through the different problem solving steps more clearly, before giving an answer.

4.1 Riddle me this

Logic puzzles are one of the simples ways to show the improved reasoning skills of reasoning models. Look at the following riddle:

> George, William, John, Abe, and Millard have their birthdays on consecutive days, all between Monday and Friday.
>
> - George's birthday is as many days before Millard's as William's is after Abe's.
> - John is two days older than Abe.
> - Millard's birthday is on Thursday.
>
> Can you figure out whose birthday is on each day?

First try to solve it yourself! What is the answer?

Monday - John, Tuesday - George, Wednesday - Abe, Thursday - Millard, Friday - William

4.2 Compare the answers given by 2 different LLMs

Now pair up with a partner and use 2 different chats to ask this question.

- one of you will use `Mistral small-3.2-24B-2506 quantized`
- The other will use a smaller model such as `Microsoft Phi-4-mini quantized` model we were already using.
- Both use the same system and user prompts to ask the question.

Did both get it right? What did you think of the difference in the answers?

Now also try with the reasoning model: `Deepseek R1-0528-Qwen-8B quantized` Reasoning models are trained to first spend tokens 'thinking': reasoning about a problem, and planning which steps to solve next. This increases the amount of compute a model can spend on an answer, and creates usefull extra context tokens in de context-window. These models tend to easily one-shot this riddle. Why no use them all the time? I usually do, but they are more expensive and slower to run, and for many tasks the regular models are good enough.

5 Summarizing texts

One simple but time consuming task you might often do is to summarize meeting notes, or perhaps summarize presentation contents.

5.1 Use your LLM to summarize the contents of `Introduction to prompt engineering` slides

I've prepared a pdf version of the document:

- LLMs for Everybody - Introduction to Prompt Engineering.pdf

It can be found in the course material: exercises → documents

Write a prompt to summarize the content of the slides. Make sure to include the content of the slides as direct text: (*For this assignment you can just copy-paste from the pdf*)

What do you think of the quality of the summarization? Did it miss anything? Iterate over the summary untill you are satisfied.

6 Generating movie reviews

One of the areas where these models really shine is content generation. Or at least, getting you started with written content.

Have the model generate a review of a movie or television series of your own choice, based on your own opinions. Make use of the following rules:

- Make clear what the ‘reviewer’ liked about the movie, and what it didn’t like.
- Begin the movie review with some general information about the movie: such as the director, star actors/actresses, runtime and other such information.
- Don’t include a star rating or other numeric rating within this review.
- Give the model some reference material about the movie/series, for example from IMDB.
- The review should be around 500 words long.

7 Do you have time left? Perform the above exercises on gemini.google.com, ChatGPT.com, or huggingchat.co/chat/

You can just use the free account for these 3 options, and see how the state-of-the-art cloud models compare to, the state of the art open-weights models, compared to the the local open-weights models we have used so far.

- Gemini: ‘https://gemini.google.com‘ is currenctly the leading model above ChatGPT
- HuggingChat: ‘https://huggingchat.co/chat‘ is a collection of free open-weights models. We have been using huggingface so far to download the models for the exercises, but it also has a chat interface hosting some seriously impressive (and big) open-weights models