

Assignment #2:

Understanding LLMs' Inner-workings and Prompt Engineering

Behnam Mo

September 20, 2026

Q1 [15 pt]. In the class, I used a tool to load a local language model and demonstrated the iterative process of token generation by the model.

- **A [5 pt].** What was the name of the tool again?
- **B [5 pt].** Which of the following better illustrates the iterative process we saw in the class?
- **C [5 pt].** What is the probability of the generated token trajectory?

Option 1:

Iteration 1: *How are you ...*

- doing → 40%
- holding → 30%
- going → 20%
- today → 5%
- ⟨other tokens⟩ → 5%

The model picks “holding” with 30% probability and appends it to the original prompt.

Iteration 2: *How are you holding ...*

- up → 80%
- this → 10%
- ⟨other tokens⟩ → 10%

The model picks “up” with 80% probability and appends it to the previous prompt.

Iteration 3: *How are you holding up ...*

- ? → 99%
- ⟨other tokens⟩ → 1%

The model picks “?” with 99% probability and appends it to the previous prompt.

Result: **How are you holding up?**

Option 2:

Iteration 1: *How are you ...*

- doing → 40%
- holding → 30%
- going → 20%
- today → 5%
- ⟨other tokens⟩ → 5%

The model picks “going” with 20% probability and keeps it in its “memory”.

Iteration 2: MEMORY: *going ...*

- to → 80%
- from → 10%
- ⟨other tokens⟩ → 10%

The model picks “to” with 80% probability and appends it to its memory.

Iteration 3: MEMORY: *going to ...*

- do → 50%
- find → 20%
- start → 10%
- ⟨other tokens⟩ → 20%

The model picks “start” with 10% probability and appends it to its memory.

Iteration 4: MEMORY: *going to start ...*

- ? → 99%

- $\langle \text{other tokens} \rangle \rightarrow 1\%$

The model picks “?” with 99% probability and appends it to its memory. Then the model appends its memory to the original prompt, resulting in:

Result: **How are you going to start?**

Q2 [20 pt]. Consider two models: “Gemma 4 – Pretrained” and “Gemini 3.8 Flash”.

- **A [5 pt].** Which one is more likely to generate harmful content?
- **B [5 pt].** Which one would you pick to power a customer support chatbot?
- **C [5 pt].** Which one is a more accurate reflection of the internet?
- **D [5 pt].** Which one does not need an internet connection to work?

Q3 [25 pt]. An AI model has a token dictionary of size 200,000.

- **A [5 pt].** If the temperature is not zero and the model is allowed to generate a response of length 128, how many unique responses can it generate?
- **B [5 pt].** Are all of them equally likely? (Yes/No)
- **C [5 pt].** Are all of them grammatically correct? (Yes/No)
- **D [5 pt].** If the temperature is set to zero, how does the answer to part A change?
- **E [5 pt].** Explain why the value you calculated in part (A) makes a marketing chatbot impossible to QA by enumerating outputs, which is precisely what motivates evaluation and guardrails.

Q4 [40 pt]. Use OpenRouter and create a customer support chatbot similar to what we built in the class. Use your preferred AI model. For each part, attach a screenshot to your final report. You can use the course material on prompt engineering for insights.

- **A [5 pt].** Add a role and context to the bot.
- **B [5 pt].** Define five tasks the agent should help customers with. Put them in its system prompt.
- **C [5 pt].** Interact with the bot and ask questions that illustrate that it follows instructions and covers said tasks. Then interact with the bot, push its boundaries on those tasks, and discover five edge cases that you had not thought of before hardening [1 pt per edge case].

- **D [10 pt]**. Let's make it formal: Write a fixed 8-case test set with an explicit pass/fail rule per case; run it against the bot before and after hardening; report the pass rate (pass@1¹) as a percentage each time [**1 pt per test**]. Report the percentage of improvement after prompt hardening [**2 pt**].
- **E [5 pt]**. The bot should be able to handle questions that require more reasoning. Use the right technique to add appropriate instructions to the bot. Then demonstrate the effectiveness of that (show three “before” and “after” responses from the model).
- **F [5 pt]**. Now try the bot on five tasks it was not designed to handle and add zero-shot instructions that guide the bot on how to respond to such requests.
- **G [5 pt]**. Repeat the previous part, except this time use few-shot prompting instead [**3 pt**]. Compare the effectiveness of this approach with zero-shot. Which one costs you more? [**2 pt**]

¹The pass@1 metric is the share of test cases the bot handles correctly on its first attempt, with no retries.