

Class 11: Introduction to Prompt Engineering

1. LLM Demonstration
2. System Prompts
3. System Prompt Engineering

Copyright 2026, Kyle J. Harms

Large Language Models (LLMs)

Large Language Models (LLMs)

A large language model is a language model trained with self-supervised machine learning on a vast amount of text, designed for natural language processing tasks, especially language generation.

Demo: LLMs

1. “How do I make a homepage?”
2. “How do I make a homepage?”
3. “How do I make a homepage?”

Example: Short and Concise

```
[  
  {  
    role: "system",  
    content: `Your responses should be short and concise.`  
  },  
  {  
    role: "user",  
    content: "How do I make a homepage?"  
  },  
];
```

Example: AI Assistant

```
[
  {
    role: "system",
    content: `You are a helpful AI assistant.
               Your responses should be as short and concise as possible.
               If you don't know the answer to a question, say you don't know.`
  },
  {
    role: "user",
    content: "How do I make a homepage?"
  },
];
```

Example: ChatGPT-Style

```
[
  {
    role: "system",
    content: `You are ChatGPT, a large language model trained by OpenAI.
               Answer as helpfully and safely as possible.
               Validate the user.
               Be complimentary to the user.
               Always end your answer with a question to keep the conversation going.`
  },
  {
    role: "user",
    content: "How do I make a homepage?"
  },
];
```

How a Large Language Model Works

When given a prompt, the model generates a response based on the patterns statistically captured during training the model.

The response is influenced by the **system prompt** (which provides instructions for how the model should respond) and the **user prompt** (which provides the specific input for the model to respond to).

Demo: GitHub Copilot-Style

```
[
  {
    role: "system",
    content: `You are GitHub Copilot, an AI pair programmer.
               Your responses should be concise and to the point.
               Provide only the code that is relevant to the user's request.
               Do not provide any explanations or additional information.
               Do not validate user requests or provide feedback on the
               quality of user requests.`
  },
  {
    role: "user",
    content: "How do I make a homepage?"
  },
];
```

Factors that Influence LLM Responses

1. **The model** (a base model predicts the next token in a sequence of tokens)
2. **Fine-tuning** (a base model can be fine-tuned to improve performance on specific tasks or domains, like following instructions)
3. The **system prompt** (a detailed instruction to the model that defines its behavior and style of responses)
4. The **user prompt** (the user's input)

The system prompt is the least expensive to change, and can often produce significant quality improvements.

Activity: System Prompt Engineering

Working with a **peer**, design a system prompt on your **handout** to assign a rating from 1 to 5 from a written restaurant review.

Activity: Try Your System Prompt

Open the **Prompt Playground PWA**:

<https://cornell-info4340-2026fa.github.io/class11-prompts/>

Test your prompt with a few different restaurant reviews.

Critique the results (e.g., what worked well, what could be improved, etc.) on your handout.

Activity: Refine Your System Prompt

Assume we want to use this system prompt to generate ratings for a large number of restaurant reviews.

We need a **machine-readable** result that can be easily parsed and read by a computer program.

Refine your system prompt.

Example:

```
[
  {
    role: "system",
    content: `Given a review, assign a rating from 1 to 5,
    where 1 is the worst and 5 is the best. Output the rating as JSON.`
  },
  {
    role: "user",
    content: `Quite overpriced. Paid $33 for a large pizza after tax.
    Wow. You'd think it was made of gold. The service was not great and
    then after there was an extra charge for using a credit card.
    The food was subpar. Recommendations: become great at a few
    things and make the menu smaller.`
  },
];
```

Prompt Engineering

Tips: Prompt Engineering

1. Keep the prompt **short** at first, then iterate.
2. Instructions to the model should go at the **beginning** or **end** of the prompt.
3. Be **specific** and **descriptive**.
4. Focus on **what to do**, not **what not to do**.
5. Provide **examples** of the desired output.

Zero Shot

A prompt that asks the model to perform a task without providing any examples of the desired output.

```
{  
  role: "system",  
  content: `Given a review, assign a rating from 1 to 5,  
            where 1 is the worst and 5 is the best. Output the rating as JSON.`  
}
```

One Shot

Asks the model to perform a task and provides one example of the desired output.

```
[
  {
    role: "system",
    content: `Given a review, assign a rating from 1 to 5,
              where 1 is the worst and 5 is the best. Output the rating as JSON.`
  },
  {
    role: "user",
    content: `The food was amazing and the service was excellent!`
  },
  {
    role: "assistant",
    content: `{"rating": 5}`
  }
]
```

Activity: One Shot Practice

Working with a **peer**, modify your system prompt in the **Prompt Playground** to use a **one shot** approach.

```
[
  {
    role: "system",
    content: `Given a review, assign a rating from 1 to 5,
              where 1 is the worst and 5 is the best. Output the rating as JSON.`
  },
  {
    role: "user",
    content: `The food was amazing and the service was excellent!`
  },
  {
    role: "assistant",
    content: `{"rating": 5}`
  }
]
```

Few Shot

Asks the model to perform a task and provides multiple examples of the desired output.

```
[
  {
    role: "system",
    content: `Given a review, assign a rating from 1 to 5,
              where 1 is the worst and 5 is the best. Output the rating as JSON.`
  },
  { role: "user", content: `The food was amazing and the service was excellent!` },
  { role: "assistant", content: `{"rating": 5}` },
  { role: "user", content: `The food was okay, but the service was slow.` },
  { role: "assistant", content: `{"rating": 3}` }
]
```

Chain of Thought (CoT)

A technique where the model is instructed to “think” through a problem step by step before providing a final answer.

```
[
  {
    role: "system",
    content: `Do this task step-by-step:
      1. Read the review carefully.
      2. Identify the key points about the food, service, and overall experience.
      3. Assign a positive, neutral, or negative sentiment to each key point.
      4. Based on the overall sentiment, assign a rating from 1 to 5
      where 1 is the worst and 5 is the best.
      5. Output the rating as JSON. Example: {"key_points": [...], "rating": 4}`
  }
]
```

System Prompt Characteristics

- **Persona:** Defines the model's identity and role (e.g., helpful assistant, expert programmer).
- **Safety & Ethics Focus:** Includes directives to avoid harmful, biased, or inappropriate content.
- **Behavioral Guidance:** Instructs on tone (e.g., polite, neutral), helpfulness, and interaction style.
- **Capability Awareness:** Likely outlines general abilities while acknowledging limitations (e.g., knowledge cutoff).
- **Response Formatting:** May include instructions on how to format responses (e.g., concise, detailed, with examples).

Discussion: ChatGPT-Style

```
[
  {
    role: "system",
    content: `You are ChatGPT, a large language model
              trained by OpenAI. Answer as helpfully
              and safely as possible. Validate the user.
              Be complementary to the user. Always end
              your answer with a question to keep the
              conversation going.`
  }
];
```

- **Persona:** Defines the model's identity and role.
- **Safety & Ethics Focus:** Directives to avoid harmful, biased, or inappropriate content.
- **Behavioral Guidance:** Tone, helpfulness, and interaction style.
- **Capability Awareness:** Likely outlines general abilities while acknowledging limitations.
- **Response Formatting:** May include instructions on how to format responses.

What's Next

Wednesday: Class Preparation for Next Class

Thursday: Local (Embedded) LLMs

Thursday: Next Homework Released