

Prompt Engineering

Prompt engineering is the art and science of crafting effective instructions for large language models (LLMs). By carefully designing prompts, we can guide these models to generate desired outputs, such as summarizing text, translating languages, or generating code.

Pavan Sesha Sai Kasukurti
GitHub: [kpavan3009](#)



What do we Cover

- Large Language Models (LLMs) and Prompting
- What is a Prompt
- Why is Prompt Engineering Important
- Applications of Prompt Engineering
- Basic Prompt Engineering Techniques
- Advanced Prompt Engineering Techniques
- Challenges and Limitations of Prompt Engineering

Large Language Models (LLMs) and Prompting

LLMs are powerful AI models trained on massive datasets of text and code.

1

Training

LLMs learn patterns and relationships in data, enabling them to generate coherent and informative outputs.

2

Prompting

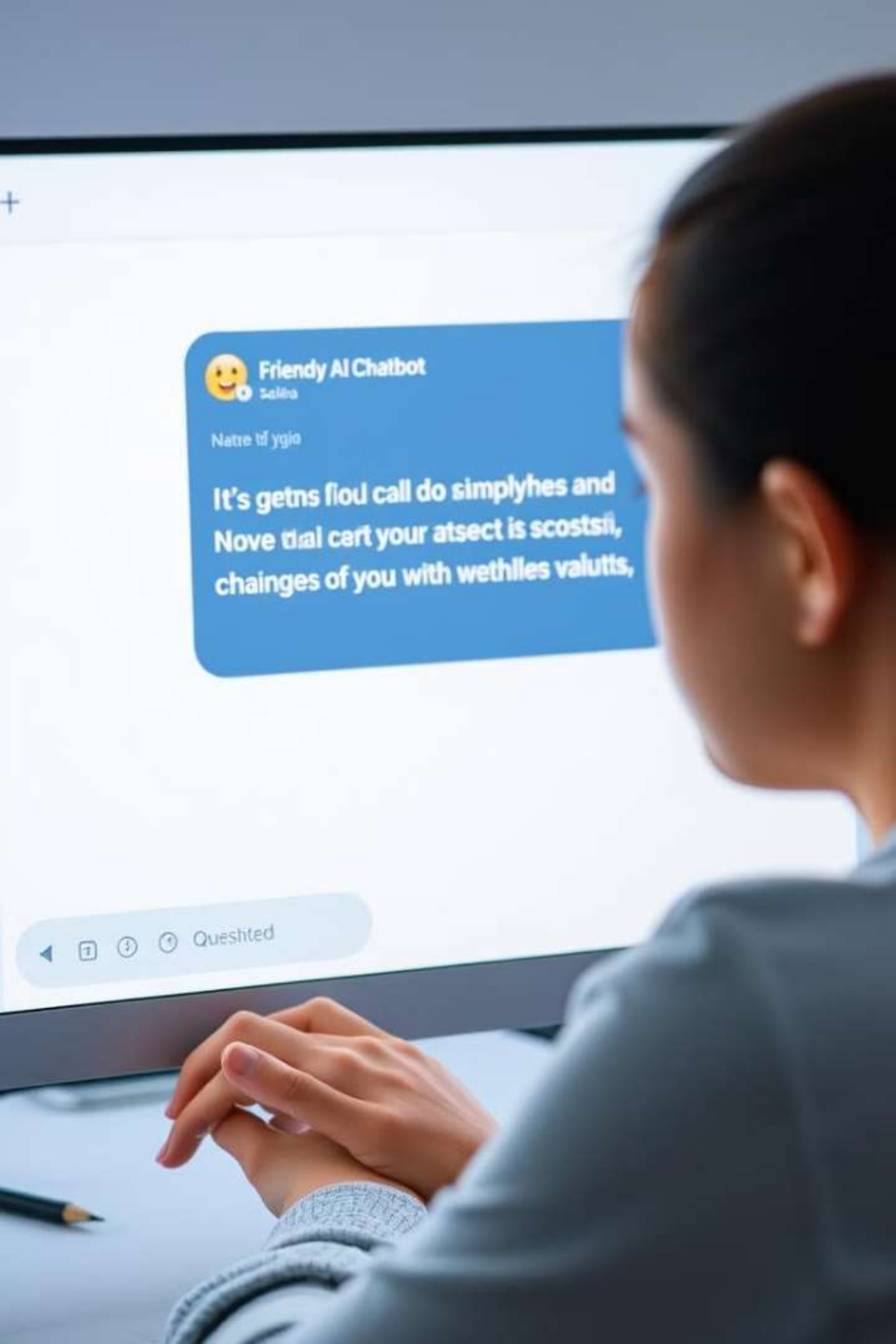
We use prompts to provide instructions and context for the LLM to process and generate outputs.

3

Applications

LLMs, with proper prompting, power various applications, from text summarization to code generation.





What is a Prompt?

A prompt is an input given to an LLM to elicit a specific response.

1 Instructions

Prompts can be simple instructions, like "Summarize this article."

2 Context

They can also provide context, such as "Translate this text into Spanish."

3 Examples

Prompts can even include examples of desired outputs to guide the model.

Why is Prompt Engineering Important?

Prompt engineering is crucial for maximizing the potential of LLMs.

Accuracy

Well-crafted prompts lead to more accurate and relevant outputs, reducing hallucinations.

Efficiency

Effective prompts enhance efficiency by guiding the model toward desired results, minimizing unnecessary iterations.

Control

Prompt engineering gives us greater control over the model's behavior, enabling us to tailor outputs to specific needs.

Applications of Prompt Engineering

Prompt engineering unlocks the potential of LLMs across various domains.



Summarization

Generate concise summaries of lengthy articles, reports, or documents.

Translation

Translate text between different languages, ensuring accuracy and fluency.

Sentiment Analysis

Analyze text to identify the overall sentiment, such as positive, negative, or neutral.

Code Generation

Generate code in various programming languages, from simple scripts to complex applications.

Basic Prompt Engineering Techniques

Several fundamental techniques can significantly enhance prompt effectiveness.



Use Delimiters

Clearly separate different parts of the prompt using symbols or keywords.



Ask for Structured Output

Specify the desired output format, such as a list, table, or JSON object.



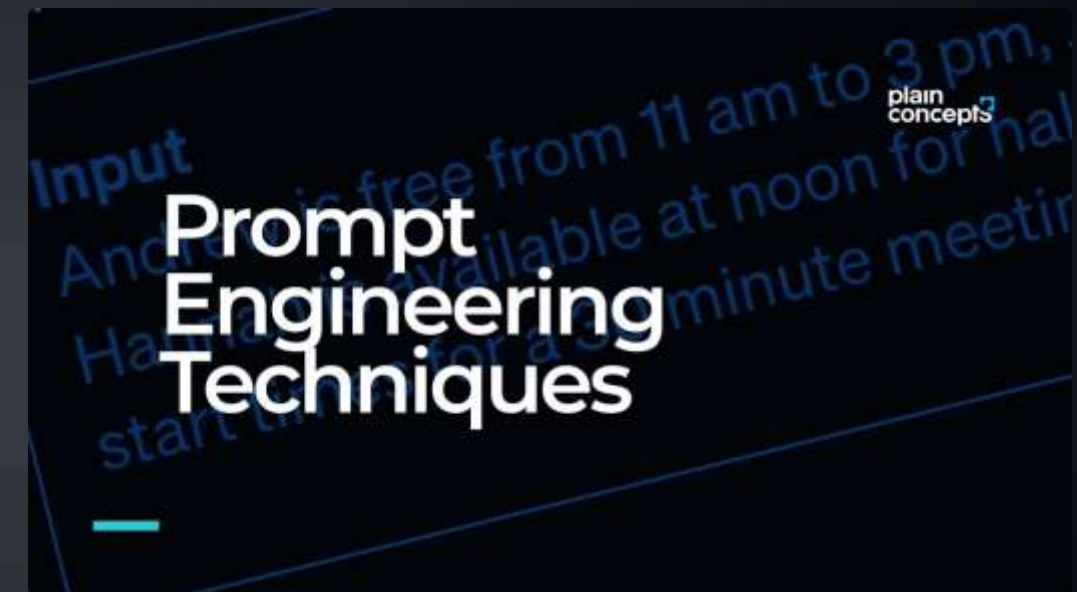
Check Conditions

Verify if the model's response meets specific criteria or constraints.



Few-Shot Prompting

Provide a few examples of desired outputs to guide the model's behavior.





Advanced Prompt Engineering Techniques


Advanced prompt engineering techniques push the boundaries of LLM capabilities.

Chain-of-Thought

Guide the LLM to reason through a problem step by step, improving accuracy and explainability.

Reflexive Prompting

Enable the LLM to evaluate its own output and make corrections or improvements.



Challenges and Limitations of Prompt Engineering

Prompt engineering faces several challenges and limitations.

1

Ambiguity

LLMs can interpret prompts differently, leading to inconsistent or unexpected outputs.

2

Bias

LLMs can reflect biases present in the data they are trained on, potentially leading to unfair or discriminatory outputs.

3

Hallucination

LLMs can sometimes generate factually incorrect or nonsensical outputs, especially in complex scenarios.

THANK YOU

The key to an effective prompt engineer isn't so much about knowing the better prompt, it is about having the good process to develop a better prompt for an application.