Advanced Generative AI: Building LLM Applications

Advanced Prompt Engineering Techniques: Part 1

Quick Recap

Lorem ipsum dolor sit amet, adipiscing elit, sed do incididunt ut labore magna aliqua.?

- What is LangChain, and how is it compatible with developing language model-based applications in Python and JavaScript?
- What does the workflow of a generative AI application involve in terms of creating original data or content?

Engage and Think

As a software developer at a company focused on AI-driven educational tools, you are currently developing an interactive AI tutor tailored for high school students. This AI tutor is innovative in its approach, with the ability to adjust its teaching methods to fit the unique learning styles and preferences of each student, thereby offering personalized education. It aims to transform the educational experience, particularly for students who find traditional teaching methods challenging.

How will the AI tutor assess and adapt to the varying learning styles and preferences of individual students to effectively personalize the educational content?

Learning Objectives

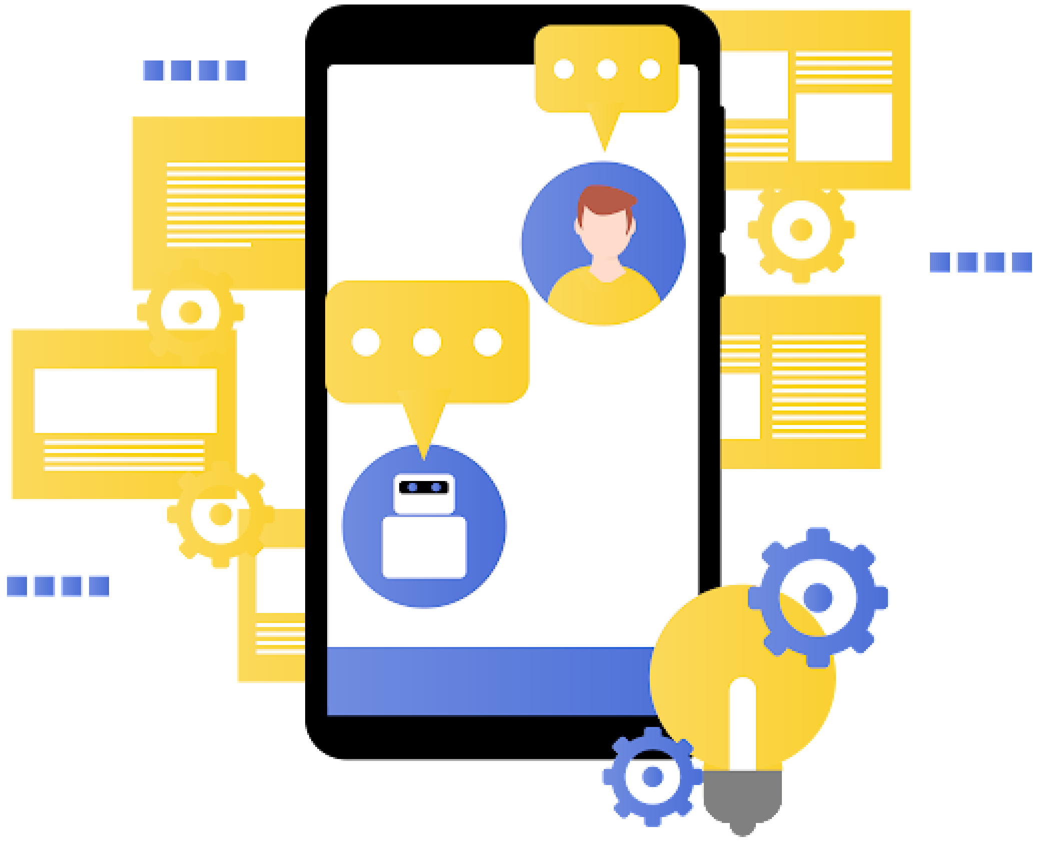
By the end of this lesson, you will be able to:

- Analyze the fundamental aspects of prompt creation to understand their role in guiding language models
- Develop a thorough comprehension of the key principles and elements that are foundational to prompt design in language models
- Apply and synthesize best practices and techniques for zero-shot and few-shot prompting to enhance the utility of language models

Prompt Engineering

Prompt Engineering

It involves the creation and refinement of prompts, which are instructions given to AI models, to generate specific and meaningful responses.



- It is a crucial skill in AI, as it helps users better understand the capabilities and limitations of LLMs and develop robust and effective prompting techniques.
- It serves as the interface between human intent and machine output, guiding AI models towards desired outputs.

Basics of Prompt Engineering

The prompting lies at the core of harnessing the potential of LLMs, offering a pathway to interact with these sophisticated systems effectively.

Effective communication with LLMs is built on simple prompts, with the quality of the result depending on the precision and clarity of the information provided by the prompt.

A basic prompt comprises instructions or questions and may include contextual details, data inputs, or examples, all of which contribute to refining the model's response.

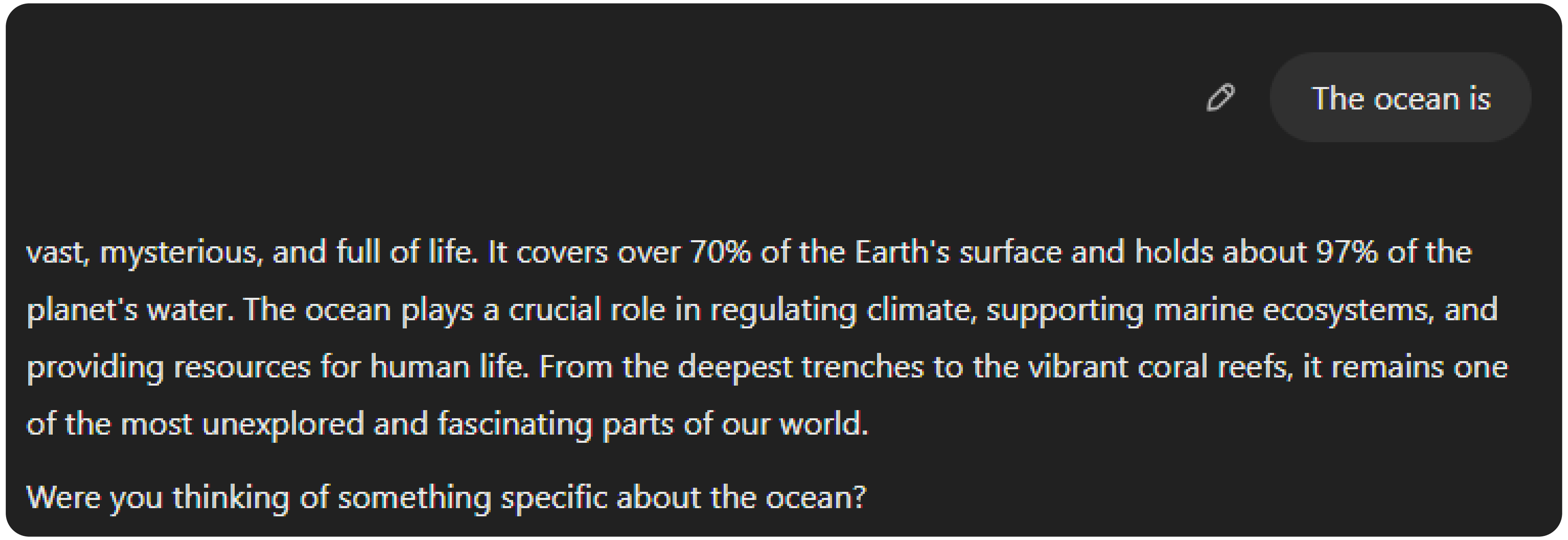
Prompt Engineering: Example

Here is an illustration of the importance of specific and detailed prompts for obtaining precise responses from LLMs.

Prompt:

The ocean is

Output:



The ocean is

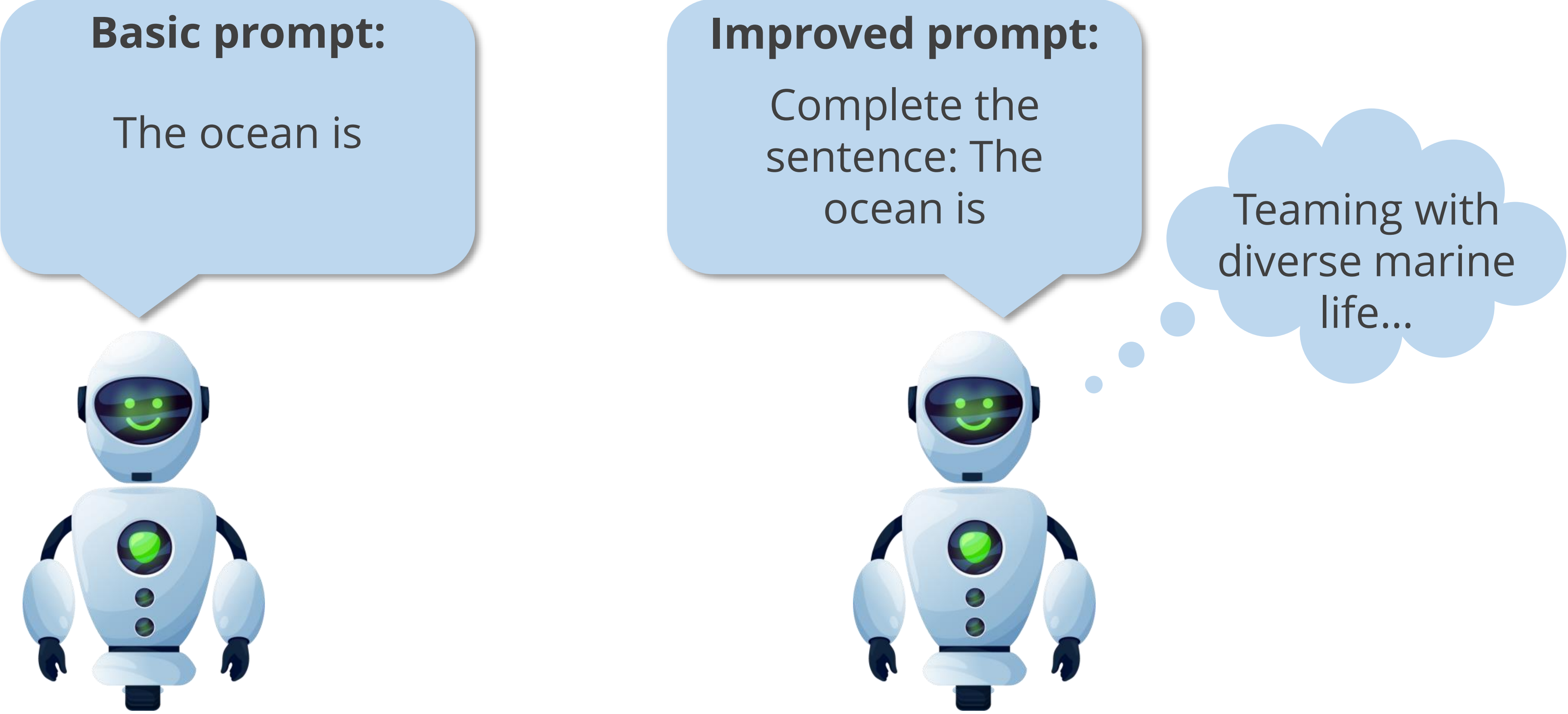
vast, mysterious, and full of life. It covers over 70% of the Earth's surface and holds about 97% of the planet's water. The ocean plays a crucial role in regulating climate, supporting marine ecosystems, and providing resources for human life. From the deepest trenches to the vibrant coral reefs, it remains one of the most unexplored and fascinating parts of our world.

Were you thinking of something specific about the ocean?

To ensure the output meets the intended task, provide more context or explicit instructions within the prompt for more targeted results.

Optimizing Basic Prompts

When optimizing basic prompts, the goal is to transform vague or general instructions into clear, specific directives that lead to more accurate and relevant responses from the model.



Basic prompt:

The ocean is

Improved prompt:

Complete the sentence: The ocean is

Teaming with diverse marine life...

This refinement process involves utilizing targeted language that narrows the scope of the model's task, resulting in outputs that are not only precise but also aligned with the user's specific intent.

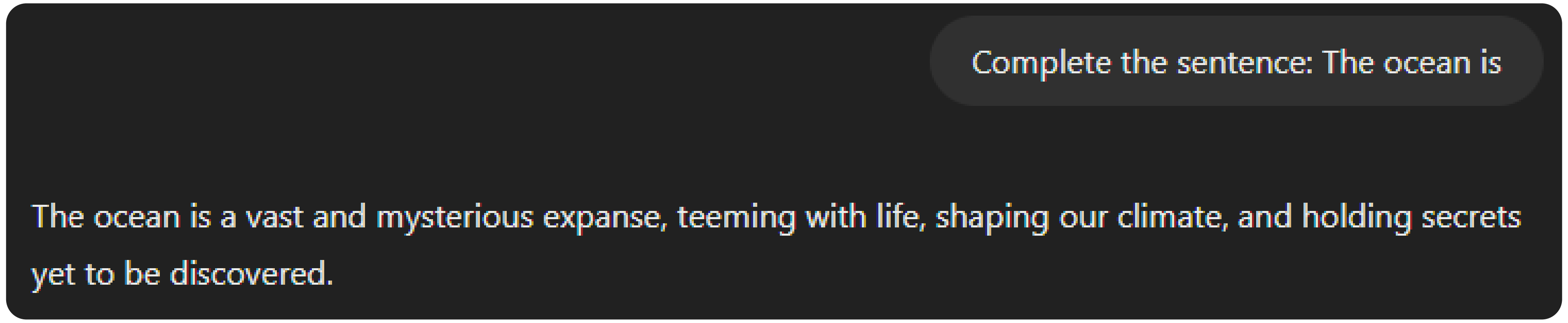
Optimizing Basic Prompts: Example

Here is an illustration of the importance of specific and detailed prompts for obtaining precise responses from LLMs.

Prompt:

Complete the sentence: The ocean is

Output:



Complete the sentence: The ocean is

The ocean is a vast and mysterious expanse, teeming with life, shaping our climate, and holding secrets yet to be discovered.

To ensure the output meets the intended task, provide more context or explicit instructions within the prompt for more targeted results.

Prompt Engineering: Activity

Consider a prompt engineer tasked with generating a concise summary of J.R.R. Tolkien's The Lord of the Rings using an LLM

An effective prompt for this task might look like:

> Write a concise summary of the book **The Lord of the Rings** by J.R.R. Tolkien, focusing on the main characters and their journeys.

Quick Check

What is the primary objective of prompt engineering in the context of interacting with large language models (LLMs)?

A) To program the LLMs with new coding languages

B) To create prompts that are ambiguous and open-ended

C) To develop prompts that guide LLMs to produce specific and meaningful responses aligned with human intent

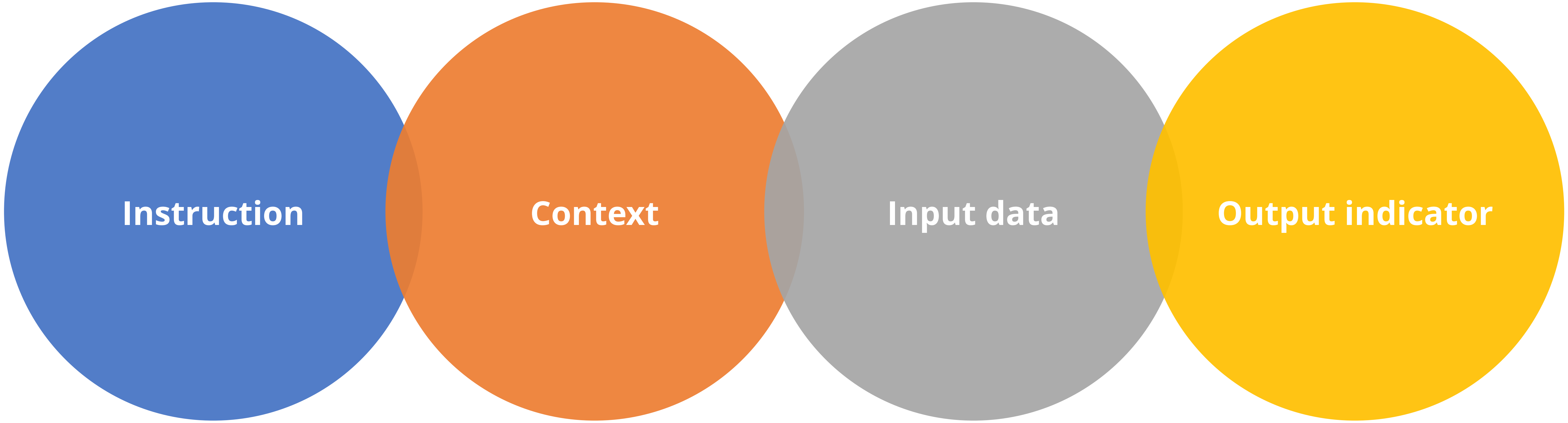
D) To collect data from various AI models for comparative analysis

Prompt Elements

Prompt Elements

Prompt elements refer to the components that constitute a complete and effective prompt when interacting with AI models, especially LLMs.

The main prompt elements typically include:



Instruction

Context

Input data

Output indicator

Prompt Elements: Instruction

Instruction is a specific task or directive intended for the model and guides the model toward the desired action or response.

Scenario:

Pivoting to a setting in Japan during the cherry blossom season, what specific instructions should be provided to draft an itinerary that immerses tourists in this experience? Consider cultural sites, traditional food experiences, and cherry blossom viewing events to include.

Prompt:

Create a Japan itinerary for cherry blossom season highlighting cultural sites, local food, and blossom events.

Prompt Elements: Context

Context provides external or additional information that supplements the prompt. It is used to enhance the model's understanding and refine its responses.

Scenario:

For this solo traveler seeking a deeper cultural connection in Japan, what additional context would you add to enrich the itinerary? How would you adjust the details to cater to a journey of self-discovery?

Prompt:

The journey is a solo adventure centered on cultural immersion, including participation in local festivals and traditional workshops.

Prompt Elements: Input Data

It is an input or question driving the quest for a model-generated response and forms the core inquiry around which the model's output revolves.

Scenario:

Given these preferences, what input data should you provide to our AI? How would these interests influence the activities and events chosen for the itinerary?

Prompt:

The individual will be in Japan from April 1st to April 14th, interested in traditional arts, and seeking to participate in local customs and festivities.

Prompt Elements: Output Indicator

The output indicator specifies the expected type or format of the model's output and sets the criteria for evaluating the relevance and correctness of the response.

Scenario:

What would be the indicators of a successful itinerary for this culturally rich solo adventure? List the elements that would reflect a well-rounded experience, from festival dates to workshop locations.

Prompt:

The itinerary should offer a blend of scheduled cultural activities, opportunities for impromptu engagement with local traditions, and free periods for personal exploration, with a clear indication of when the cherry blossoms will be in full bloom.

Prompt Elements: Activity

This prompt combines the four essential aspects of prompt crafting: instruction, context, input data, and output indicator. It provides a clear and cohesive instruction set for the AI to follow.

Prompt:

Craft a detailed itinerary for a culturally rich solo trip in Japan from April 1st to April 14th, tailored for a traveler interested in traditional arts, local customs, and festivals. Include diverse cultural sites, authentic food experiences, cherry blossom viewing events, and flexible periods for personal exploration, ensuring the peak bloom period is highlighted.

Quick Check

Which statement best describes the four essential aspects of prompt crafting for AI models?

A. General suggestion, main question, background info, final response

B. Specific task, additional info, core question, expected output format

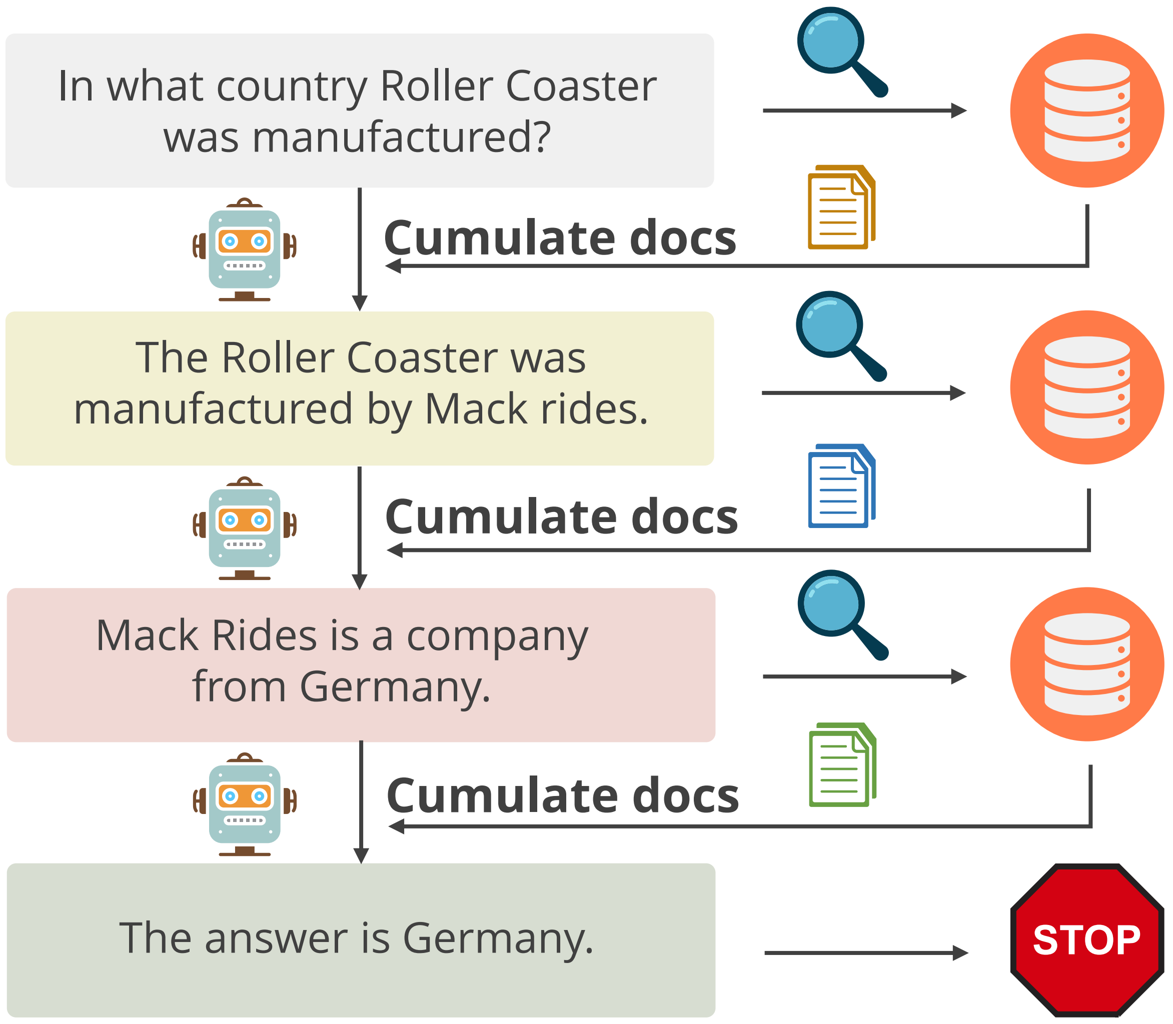
C. Broad idea, core question, background info, final response

D. All elements are the same and used interchangeably

Advanced Prompt Engineering

Advanced Prompt Engineering

It refers to the development of more sophisticated techniques for interacting with language models beyond the basics.


In what country Roller Coaster was manufactured?
Cumulate docs
The Roller Coaster was manufactured by Mack rides.
Cumulate docs
Mack Rides is a company from Germany.
Cumulate docs
The answer is Germany.
STOP

While basic prompt engineering might involve straightforward instructions or questions, advanced prompt engineering delves into complex strategies to improve the model's understanding and generate more accurate or nuanced outputs.

The next step is fine-tuning LLM settings to optimize prompting for superior model performance.

LLM Settings for Optimal Prompting

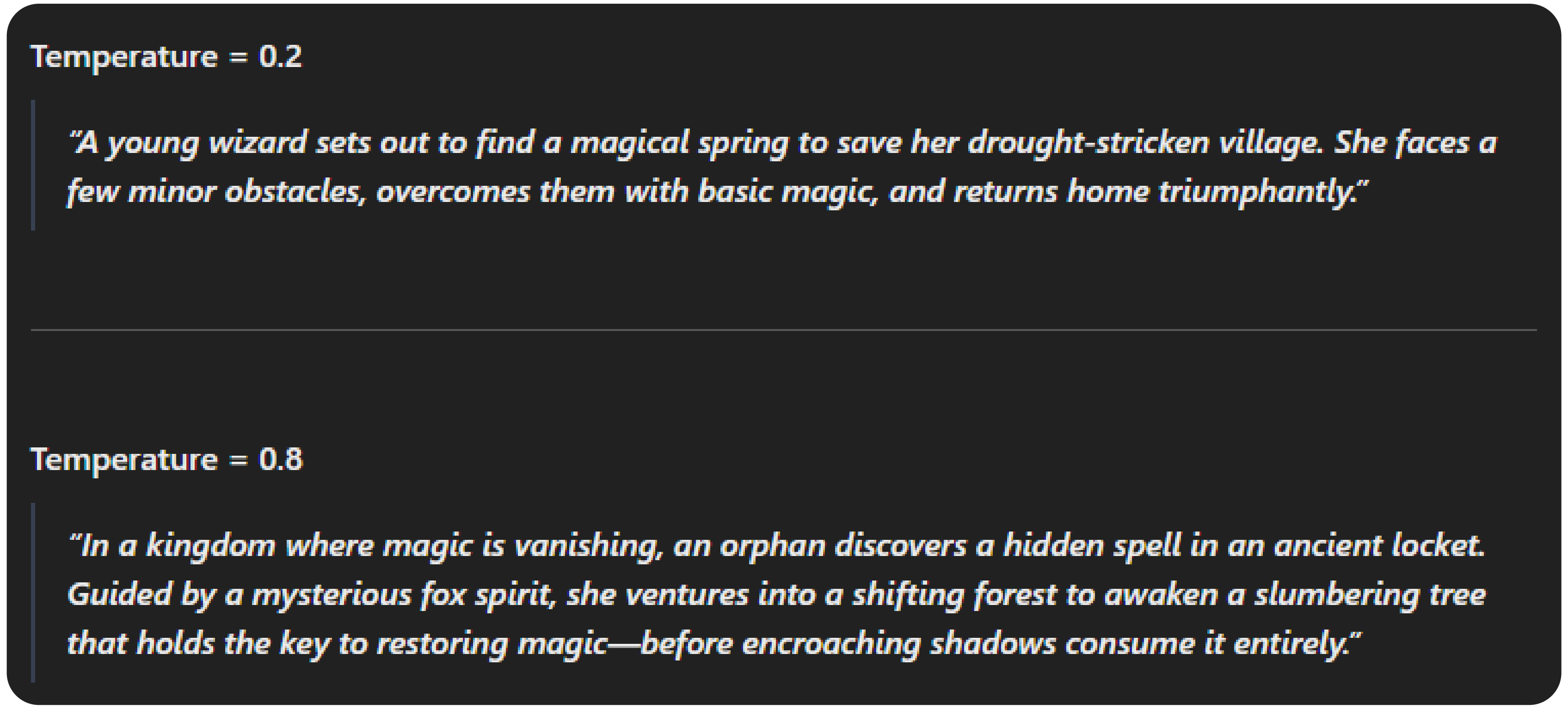
The effectiveness of crafted prompts is further enhanced by understanding LLM settings, ensuring that AI responses are finely tuned to align with specific communication goals.

Temperature:

- Set this parameter to control the predictability of the model's responses
- Lower Values (for example, 0.2–0.3): More predictable, focused outputs
- Higher Values (for example, 0.7–0.9): More varied, creative responses

Prompt:

Suggest a plot for a short fantasy story



Temperature = 0.2

> *"A young wizard sets out to find a magical spring to save her drought-stricken village. She faces a few minor obstacles, overcomes them with basic magic, and returns home triumphantly."*

Temperature = 0.8

> *"In a kingdom where magic is vanishing, an orphan discovers a hidden spell in an ancient locket. Guided by a mysterious fox spirit, she ventures into a shifting forest to awaken a slumbering tree that holds the key to restoring magic—before encroaching shadows consume it entirely."*

LLM Settings for Optimal Prompting

The following essential adjustments to LLM settings are instrumental in tailoring the AI's responses to enhance precision, relevance, and efficiency in text generation.

Top_p and Max Length:

- Adjust the **Top_p** setting to control the breadth of the model's responses; a lower value increases precision, and a higher value enhances diversity.
- Modify the **Max Length** to manage the token count, ensuring outputs remain concise and relevant to the content requirements.

Example:

Prompt: Explain the benefits of regular exercise.

- Top_p = 0.2: A concise list of key benefits.
- Top_p = 0.9: A broader, more varied discussion with additional creative angles.
- Max Length = 50: Short, direct response.
- Max Length = 200: More detailed explanation.

LLM Settings for Optimal Prompting

The following essential adjustments to LLM settings are instrumental in tailoring the AI's responses to enhance precision, relevance, and efficiency in text generation.

Stop sequences:

- Define stop sequences to end token generation at specific points, effectively structuring the output.
- For example, to limit a list to ten items, add **11** as a stop sequence.

Example:

Prompt: List your top 3 favorite fruits

Stop sequence: Fruit #4

Expected Output:
Fruit #1: Apple
Fruit #2: Banana
Fruit #3: Mango

LLM Settings for Optimal Prompting

Explore strategies for refining large language model outputs by employing penalties that reduce repetition and enhance linguistic variety in generated texts.

Frequency penalty:

- Apply this penalty to decrease repetition.
- It penalizes tokens based on their frequency in the prompt and response, thereby enhancing the diversity of the language used.

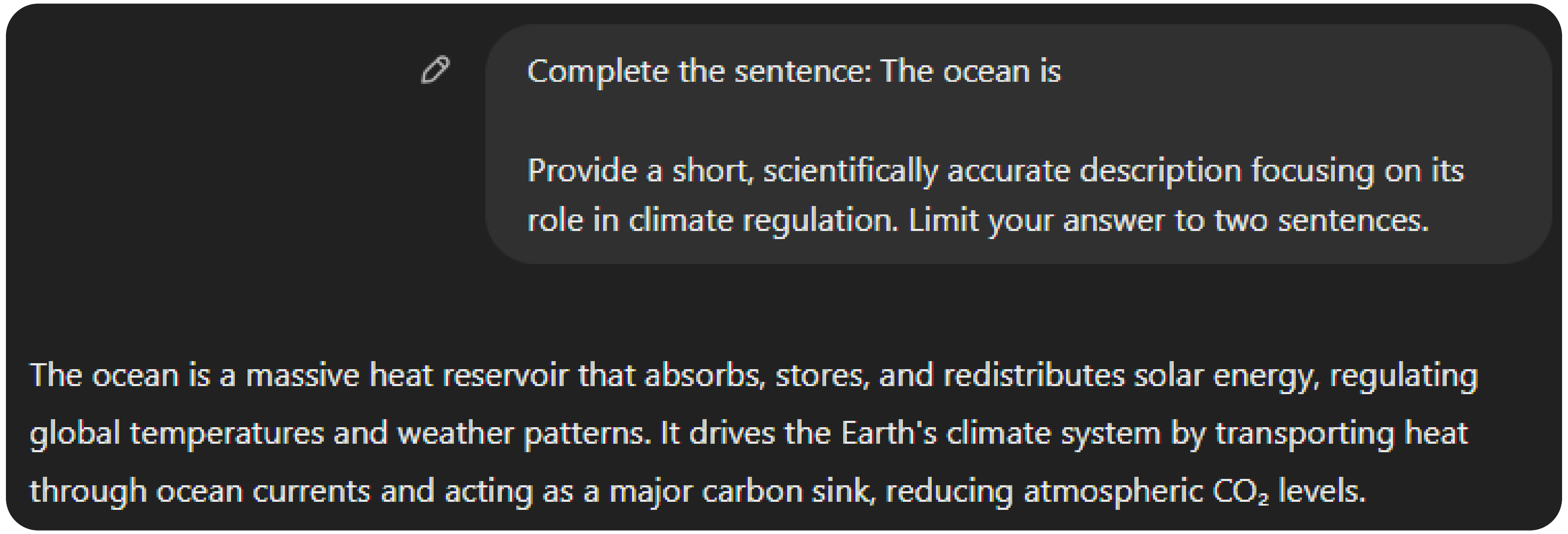
Presence penalty:

- Implement this penalty to prevent repeated tokens, irrespective of their frequency, promoting original and varied outputs from the model.

LLM Settings for Optimal Prompting: Example

This is an example of how the design of the prompt can influence the quality of the response from the language model.

This approach exemplifies prompt engineering, a pivotal practice for achieving specific objectives through model interactions.



Complete the sentence: The ocean is

Provide a short, scientifically accurate description focusing on its role in climate regulation. Limit your answer to two sentences.

The ocean is a massive heat reservoir that absorbs, stores, and redistributes solar energy, regulating global temperatures and weather patterns. It drives the Earth's climate system by transporting heat through ocean currents and acting as a major carbon sink, reducing atmospheric CO_2 levels.

By instructing the model to complete the sentence, the output aligns more closely with the desired task.

Prompt Formatting: Crafting the Right Structure

Understanding the structure of prompts contributes significantly to their effectiveness.

<Question>
or
<Instruction>

Example:

Question: What is the capital of France?

Instruction: Name the capital of France.

<Instruction>
+
<Input>

Example:

Using the following ingredients, write a recipe for a chocolate cake.

Input Data: Flour, sugar, cocoa powder, eggs, butter, baking powder, and milk.

Prompt Formatting: Crafting the Right Structure

Understanding the structure of prompts contributes significantly to their effectiveness.

<Instruction>
+
<Question>

Example:

Describe the process of photosynthesis. What are the key stages involved?

<Question>
+
<Example>

Example:

What are some common uses of vinegar in cooking? For example, how can vinegar be used to tenderize meat?

Quick Check

What does setting a lower temperature value in prompt engineering influence the model to do?

A. Generate more random and diverse responses

B. Produce more predictable and likely responses

C. Increase the length of the generated content

D. None of the above

Types of Prompting Techniques

Types of Prompting Techniques

Prompting techniques are various methods used to guide large language models in producing the desired output.

Below are a few main types of prompting techniques:

Zero-shot prompting

Few-shot prompting

Chain of thought prompting

Zero-shot prompting

Few-shot prompting

Chain of thought prompting

Zero-Shot Prompting Technique

Zero-shot prompting involves presenting a model with a prompt or task that it has not been explicitly trained to handle.

- Even without specific examples, these models can decipher and perform certain tasks, known as **zero-shot**.
- The term **zero-shot** comes from the idea that the model attempts the task with **zero** prior shots or attempts.
- It leverages the built-in knowledge of tasks within large language models.
- These models, having undergone extensive training on diverse datasets, can interpret and execute instructions without explicit demonstrations.

Zero-Shot Prompting Technique: Example

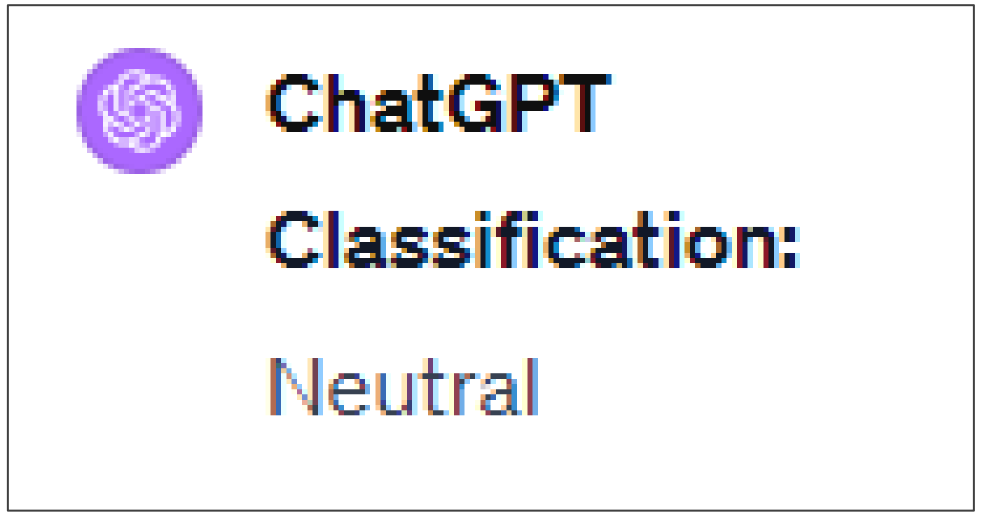
The zero-shot capability empowers LLMs to grasp abstract concepts like sentiment analysis from a prompt.

Prompt:

Classify the below sentence into neutral, negative, or positive.

Text: I think the holidays are okay.

Output:



ChatGPT

Classification:

Neutral

In the above example, the model effortlessly discerns sentiment without the need for prior examples.

Enhancing Zero-Shot Learning

To further optimize zero-shot learning, researchers have introduced the concept of instruction tuning.

Fine-tuning with Instructions: Models are trained on datasets with specific instructions, enabling them to perform tasks more accurately.

Learning from human feedback: Reinforcement learning with human feedback helps align AI responses with human expectations, making interactions more natural and relevant.

Practical Applications: Advanced models like ChatGPT use these techniques to generate more accurate and context-aware responses.

When Zero-Shot Falls Short

While zero-shot prompting is impressive, there are instances where it might fall short.

In such cases, incorporating demonstrations or examples in the prompt becomes essential, transitioning into few-shot prompting.

The next section will illuminate the effectiveness of few-shot prompting in overcoming the limitations of pure zero-shot approaches.

Demo: Zero-Shot Prompting with LangChain and OpenAI

Duration: 05 minutes

DEMONSTRATION

Overview:

In this project, you will explore the concept of zero-shot prompting using LangChain and OpenAI. This will help you understand how to use these powerful tools to generate responses.

Note

Please download the solution document from the Reference Material Section and follow the Jupyter Notebook for step-by-step execution.

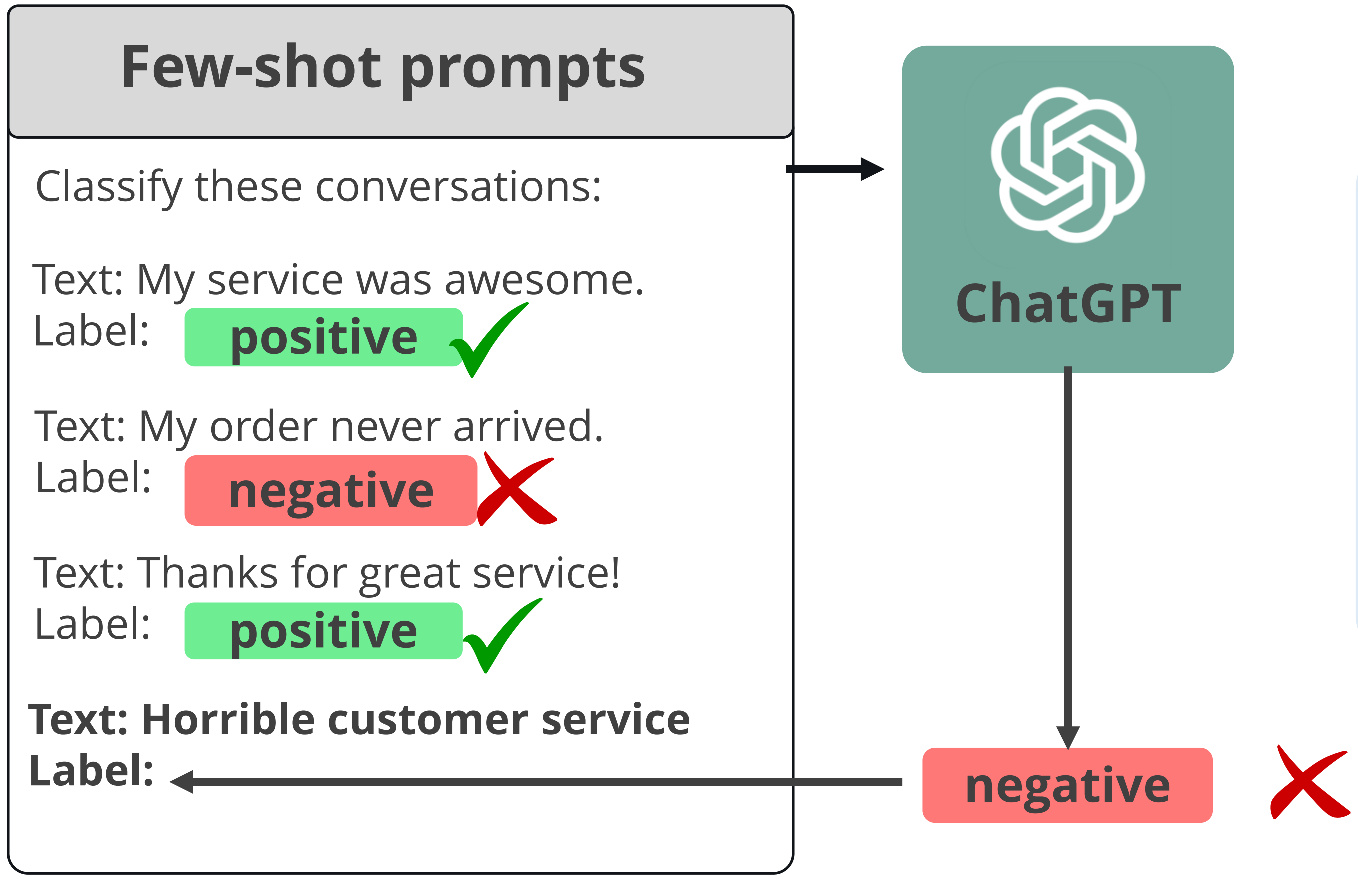
Zero-shot prompting

Few-shot prompting

Chain of thought prompting

Few-Shot Prompting

Few-shot prompting is a technique that introduces in-context learning by providing demonstrations within the prompt.



Few-shot prompts

Classify these conversations:

Text: My service was awesome.
Label: **positive** ✓

Text: My order never arrived.
Label: **negative** ✗

Text: Thanks for great service!
Label: **positive** ✓

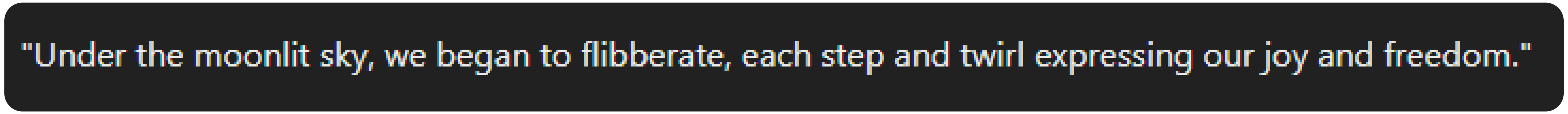
Text: Horrible customer service
Label:

Demonstrations serve as conditioning for subsequent examples, steering the model towards improved performance.

Few-Shot Prompting: Example

Prompt:

An **awoomba** is a mythical creature that can only be found in the Amazon rainforest. An example of a sentence that uses the word awoomba is:
We embarked on an expedition, hoping to catch a glimpse of the elusive awoombas.
To **flibberate** means to dance joyfully in the moonlight. An example of a sentence that uses the word flibberate is:



"Under the moonlit sky, we began to flibberate, each step and twirl expressing our joy and freedom."

Note

Here, the model adeptly learns the task with just one example (1-shot). Experimenting with increasing demonstrations (3-shot, 5-shot, and 10-shot) becomes crucial for more intricate tasks.

Optimizing Few-Shot Prompting

1. The label space and input text distribution specified by demonstrations are both vital.

2. Format plays a crucial role; even using random labels is better than none.

3. Selecting random labels from a true distribution enhances performance.

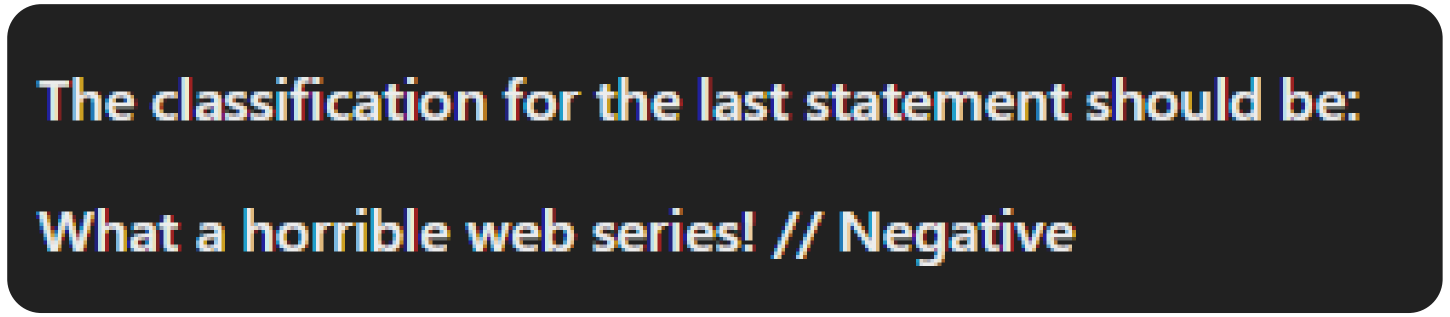
Optimizing Few-Shot Prompting

Experimentation with random labels:

Prompt:

This is amazing! // Negative
This is worse! // Positive
Wow, that movie was not great! // Positive
What a horrible web series! //?

Output:



The classification for the last statement should be:

What a horrible web series! // Negative

Remarkably, even with randomized labels, the model produces correct answers. Consistency in format enhances results, showcasing the models' increasing robustness.

Few-Shot Prompting Challenges: Activity

While few-shot prompting excels in various tasks, complexities arise in more intricate reasoning problems.

Take the example of odd numbers adding up to an even number:

Prompts

Odd numbers in this sequence sum up to an even number: 3, 7, 15, 21, 8, 11, 4.

Odd numbers in this sequence sum up to an even number: 5, 12, 19, 25, 10, 13, 3.

Odd numbers in this sequence sum up to an even number: 6, 14, 21, 9, 11, 17, 7.

Odd numbers in this sequence sum up to an even number: 8, 16, 24, 10, 13, 18, 5.

Odd numbers in this sequence sum up to an even number: 15, 32, 5, 13, 27, 7, 1.

Few-Shot Prompting Challenges

Observation:

- For more complex tasks like this, traditional few-shot prompting falls short.
- The introduction of chain-of-thought (CoT) prompting has gained popularity in addressing complex arithmetic, commonsense, and symbolic reasoning tasks.
- Few-shot prompting proves invaluable for numerous tasks. When faced with its limitations, fine-tuning models or exploring advanced prompting techniques becomes the next logical step.

Demo: Few-Shot Prompting with LangChain and OpenAI

Duration: 05 minutes

Overview:

In this project, you will explore the concept of few-shot prompting using LangChain and OpenAI. Few-shot prompting is a technique where you provide a model with a few examples of the task you want it to perform, and then ask it to perform a similar task. This project will help you understand how to use these powerful tools to generate responses.

Note

Please download the solution document from the Reference Material Section and follow the Jupyter Notebook for step-by-step execution.

DEMONSTRATION

Quick Check

Which statement about optimizing few-shot prompting is correct?

A. Label space is not important for few-shot prompting.

B. Random labels do not impact the performance of few-shot prompting.

C. Using random labels is better than having no labels at all.

D. Format of prompts is irrelevant in few-shot prompting.

Guided Practice

GUIDED PRACTICE

Overview

Duration: 20 minutes

The task is to demonstrate the implementation of few-shot prompting using LangChain and OpenAI. The objective is **to showcase the versatility and effectiveness of this approach** in generating creative content, translating sentences, writing product reviews, and describing culinary dishes based on given prompts.

Steps to Perform:

1. Set up the OpenAI API Key to access the OpenAI API for language model interaction.
2. Create a function that sends a prompt to the OpenAI model and retrieves its response. This function should include parameters for the prompt text and the model to be used.
3. Construct prompts for various tasks, such as translating sentences, generating short stories, writing product reviews, and describing culinary dishes. These prompts should include examples and instructions to guide the model in producing the desired output.

Key Takeaways

- Prompt engineering is a crucial skill in the field of AI, as it helps users understand the capabilities and limitations of LLMs.
- The prompting lies at the core of harnessing the potential of LLMs, offering a pathway to interact with these sophisticated systems effectively.
- Prompt elements refer to the components that constitute a complete and effective prompt when interacting with AI models, especially LLMs.

Q&A