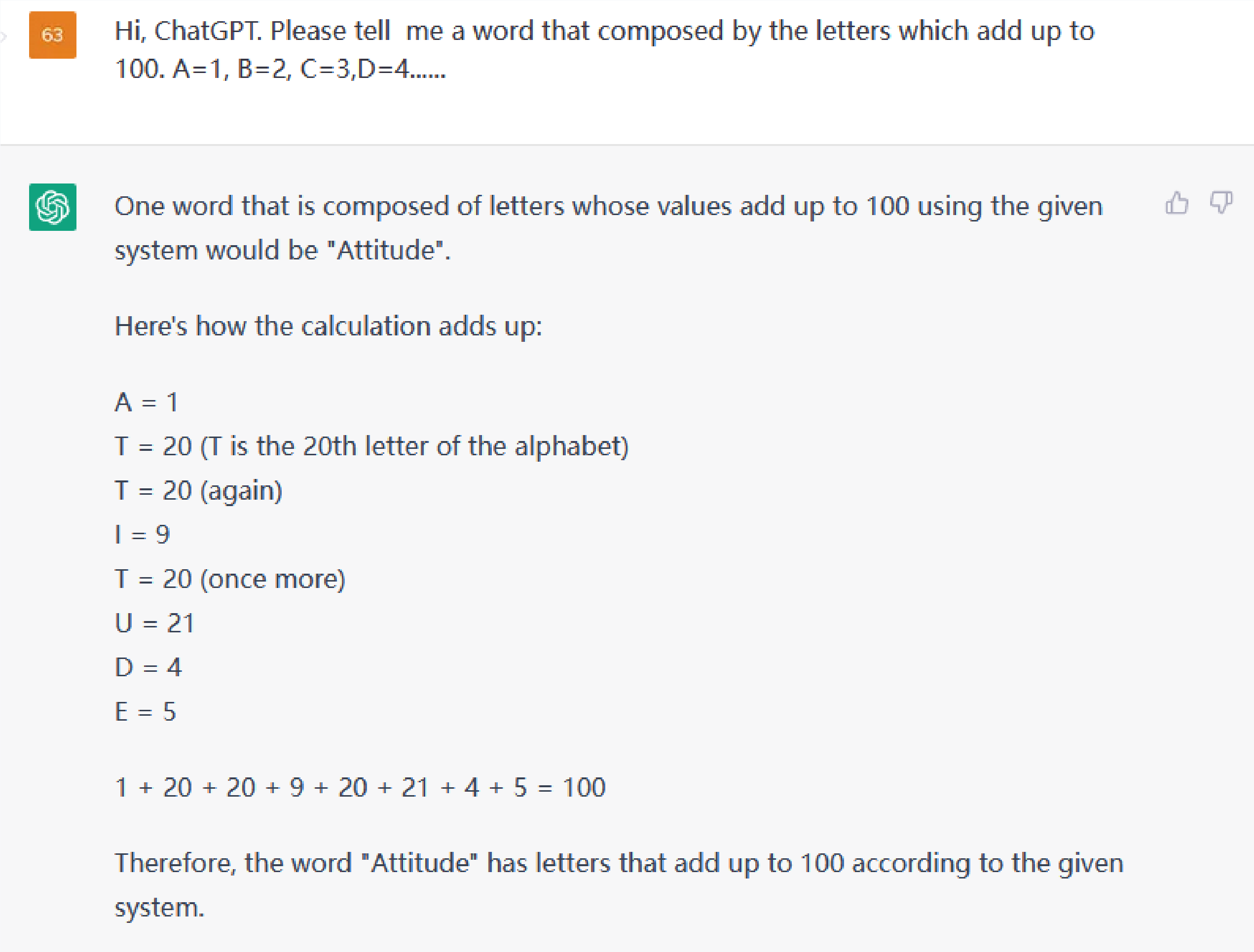
UNLEASHING THE FULL POTENTIAL OF AI LANGUAGE MODELS WITH PROMPT ENGINEERING:
INSIGHTS, TECHNIQUES, AND BEST PRACTICE

FAN YANG
PHD, LGSW

DR. AN'S AI GROUP

APR 1ST, 2023



Hi, ChatGPT. Please tell me a word that composed by the letters which add up to 100. A=1, B=2, C=3,D=4......

One word that is composed of letters whose values add up to 100 using the given system would be "Attitude".

Here's how the calculation adds up:

A = 1
T = 20 (T is the 20th letter of the alphabet)
T = 20 (again)
I = 9
T = 20 (once more)
U = 21
D = 4
E = 5

1 + 20 + 20 + 9 + 20 + 21 + 4 + 5 = 100

Therefore, the word "Attitude" has letters that add up to 100 according to the given system.

AGENDA

LEARNING GOALS

- ➢**Introduction**
- ➢**Basic Principles**
- ➢**Techniques and Applications**
- ➢**Future Directions**
- ➢**Recourses and References**

01

INTRODUCTION

1.1 Concept

CHATGPT

- a state-of-the-art language model developed by Open AI.
- Capable of generating human-like text in multiple languages and has been trained on a massive corpus of text data.

1.1 Concept

Prompt involve instructions and context passed to a language model to achieve a desired task

Prompt Engineering

- ➢ A process of creating a set of prompts, or questions, that are used to guide the user toward a desired outcome.
- ➢ is the practice of developing and optimizing prompts to efficiently use language models (LMs) for a variety of applications
- ➢ Prompt engineering is a useful skill for AI engineers and researchers to improve and efficiently use language models

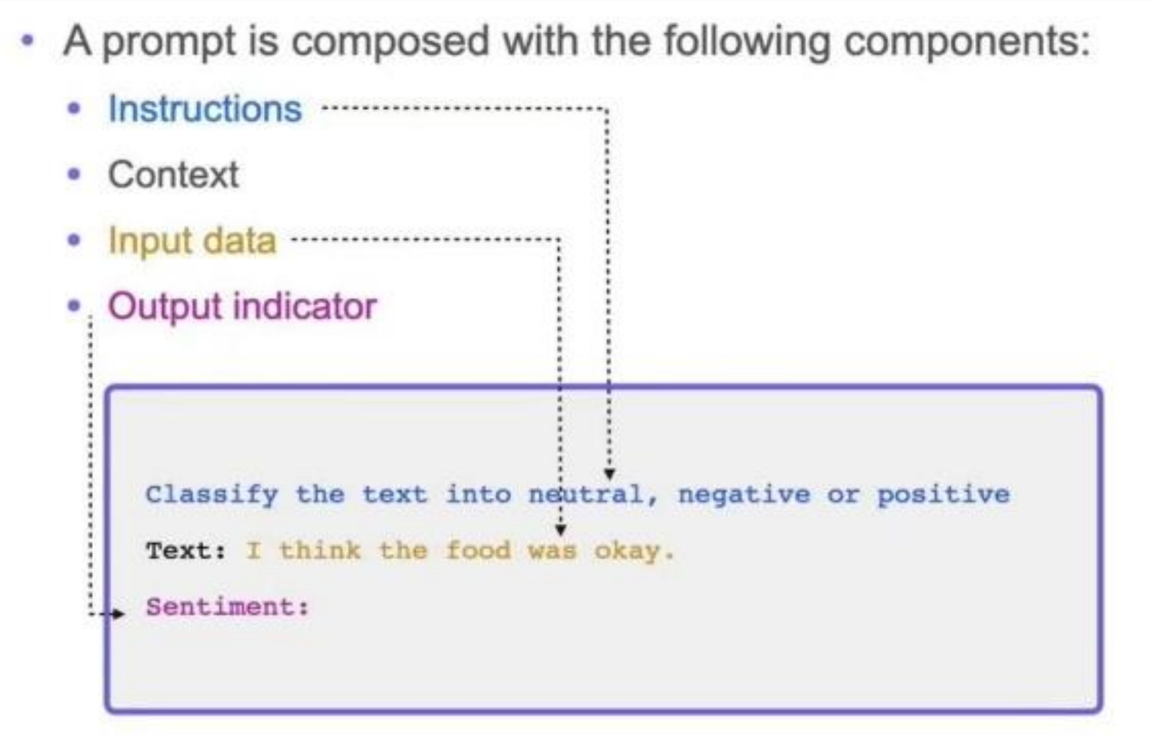
Why Prompt Engineering important

- ➢ determines the quality and relevance of the output generated by the model.
- ➢ unlock the full potential of LLM；
- ➢ achieve better results with less effort and time.

1.2 Elements of A Prompt

Prompt Structure

A prompt is composed with instructions, context, input data, and output indicator.



- A prompt is composed with the following components:
  - Instructions
  - Context
  - Input data
  - Output indicator

```
Classify the text into neutral, negative or positive

Text: I think the food was okay.

Sentiment:
```

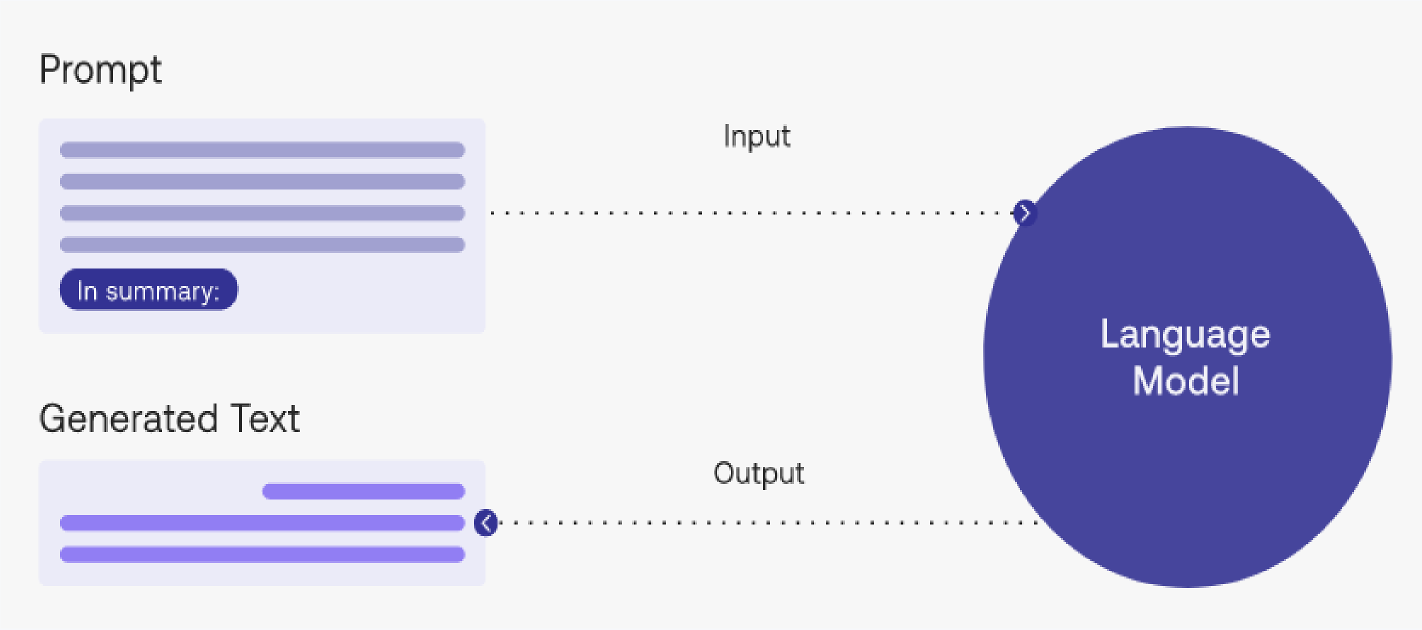
02

BASIC PRINCIPLES

FOUR PRINCIPLES

- A prompt guides the model to generate useful output
- Try multiple formulations to get the best generations
- Describe the task and the general setting
- Show the model what you would like to see

2.1 A prompt guides the model to generate useful output



This prompt has two components:

- the text you want summarized,
- the task description.

2.2Try multiple formulations of your prompt to get the best generations

Different formulations of the same prompt which might sound similar to humans can lead to generations that are quite different from each other.

Examples:

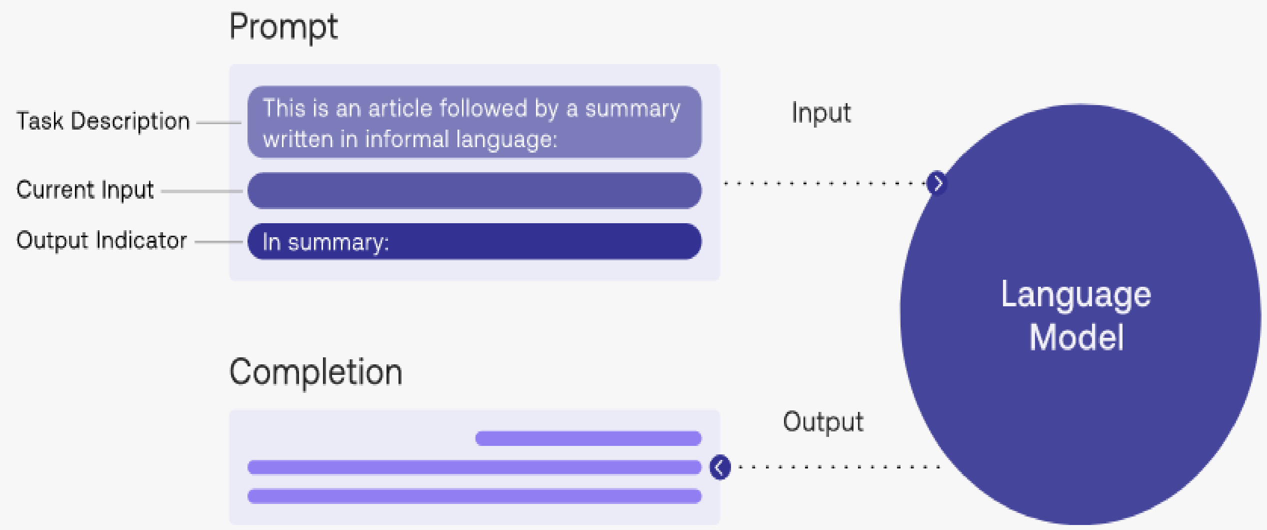
Some output indicators for the summarization examples:

“In summary:”

“To summarize in plain language, ”

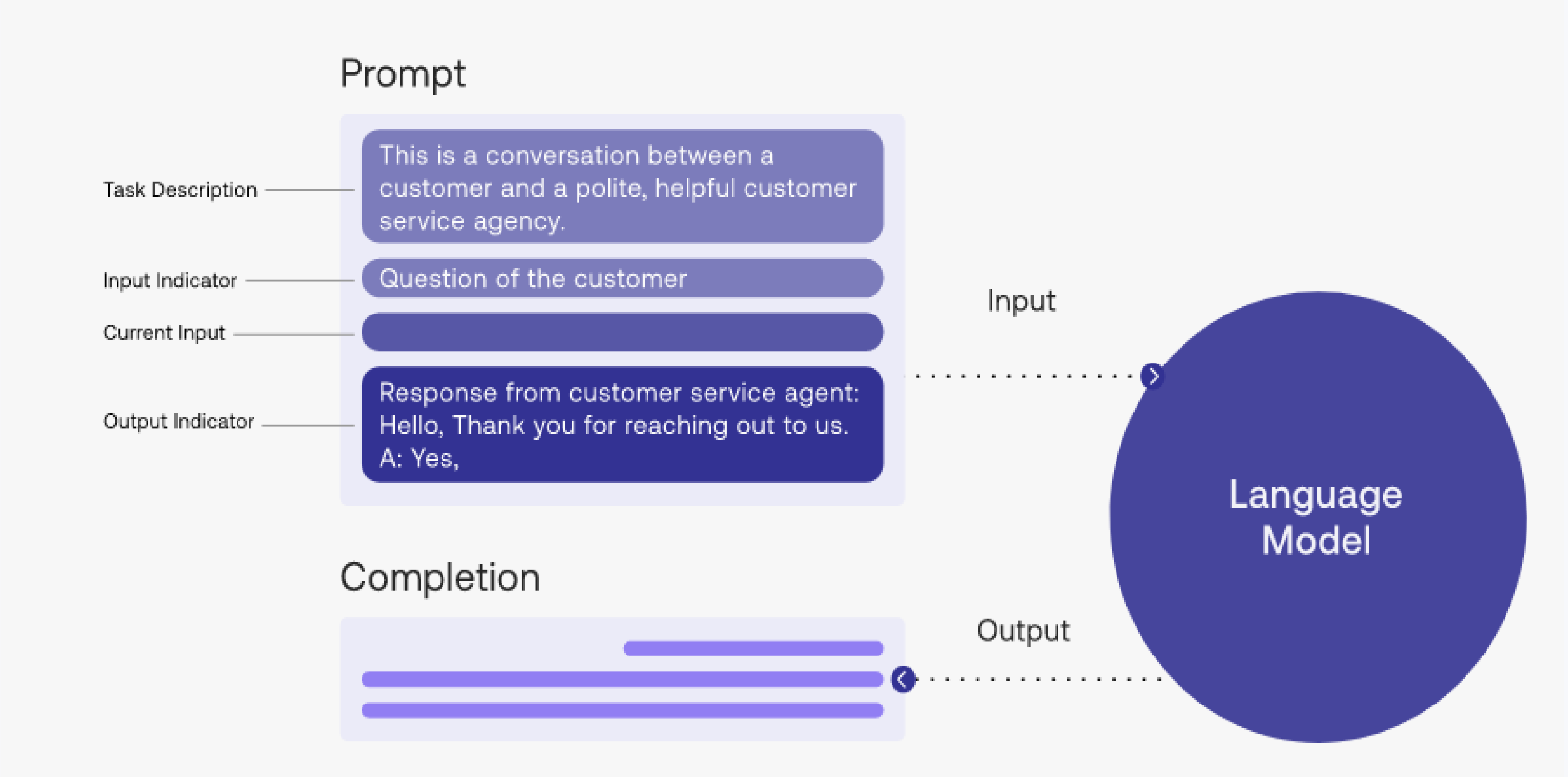
“The main point to take from this article is that” .

2.3Describe the task and the general setting



Example: shaping the task we need the model to do in natural language can use text both before and after the input text we aim to process.

2.3Describe the task and the general setting



Prompt

Task Description — This is a conversation between a customer and a polite, helpful customer service agency.

Input Indicator — Question of the customer

Current Input —

Output Indicator — Response from customer service agent: Hello, Thank you for reaching out to us. A: Yes,

Input

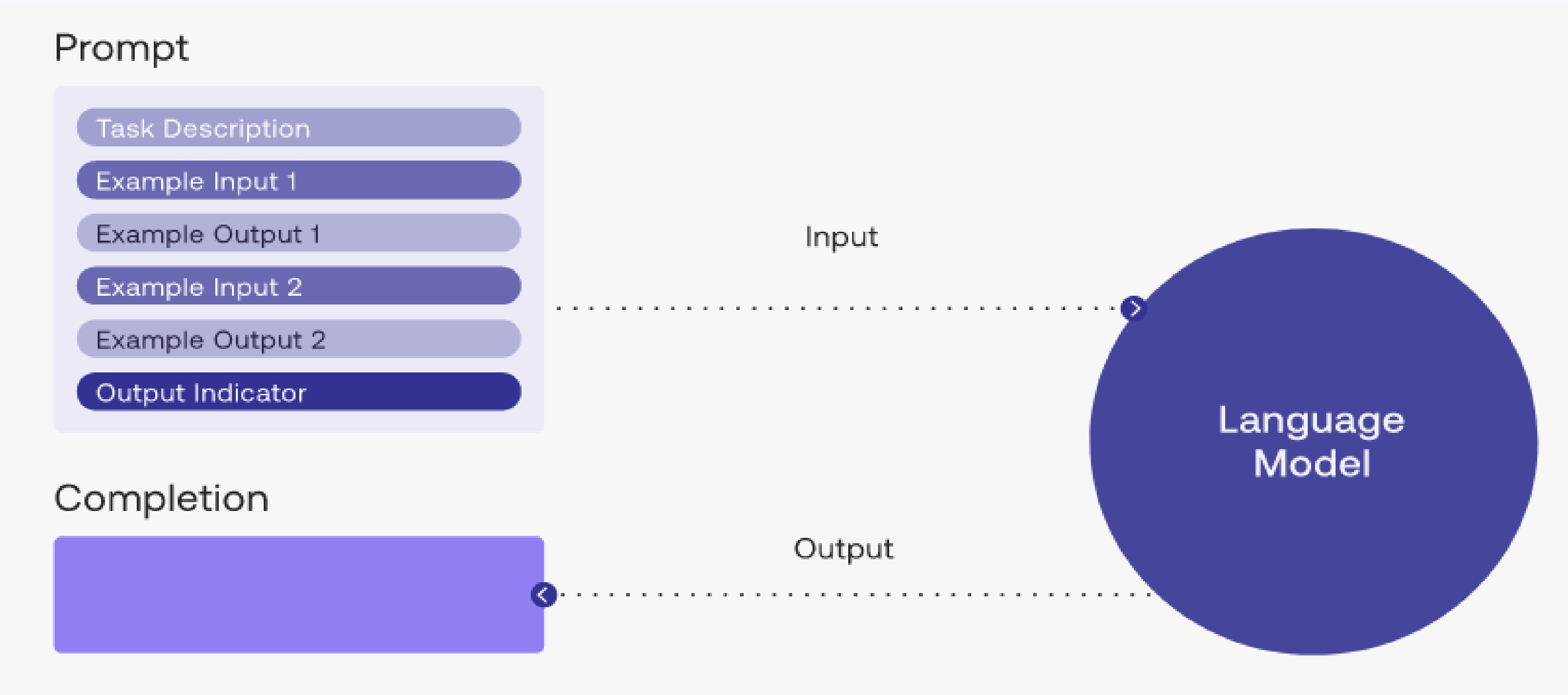
Language Model

Output

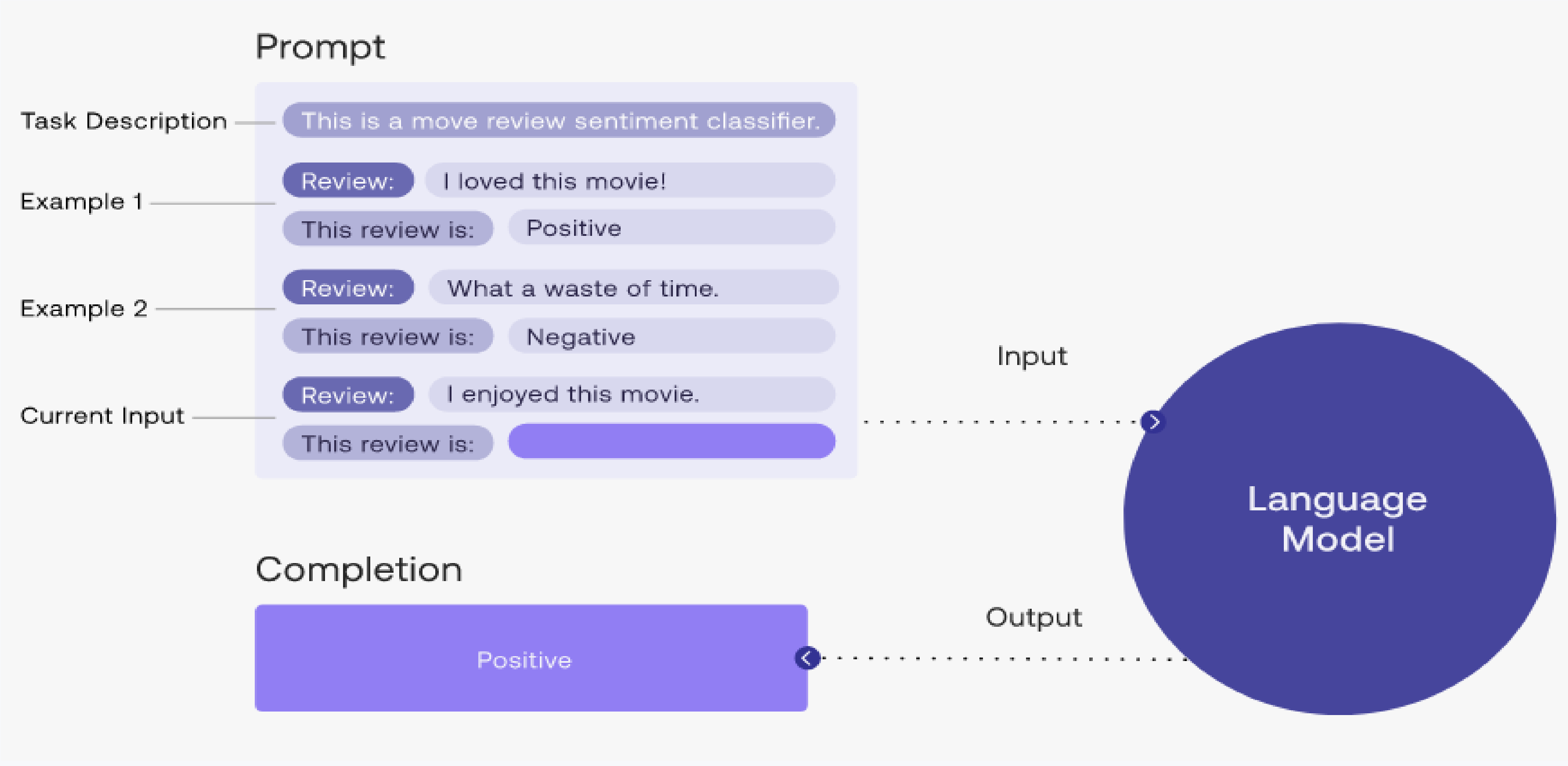
Completion

2.4 Show the model what you would like to see

Give a few examples of the types of generations you want. This is called few-shot learning.



2.4 Show the model what you would like to see



Prompt

Task Description — This is a move review sentiment classifier.

Example 1 — Review: I loved this movie! This review is: Positive

Example 2 — Review: What a waste of time. This review is: Negative

Current Input — Review: I enjoyed this movie. This review is:

Completion

Positive

03

TECHNIQUES AND APPLICATIONS

3.1 Types of Tasks

- **Text summarization**
- **Question Answering**
- **Text Classification**
- **Role Playing**
- **Code Generation**
- **Reasoning**

3.1 Few-shot prompts
3.2 Chain-of-thought CoT prompting (Reason)
3.3 Self-consistency
3.4 Knowledge generation prompting/PAL (Reason with Python interpreter)
3.5 ReAct (Reason+Act)

Setting:

- One important setting is controlling how deterministic the model is when generating completion for prompts.

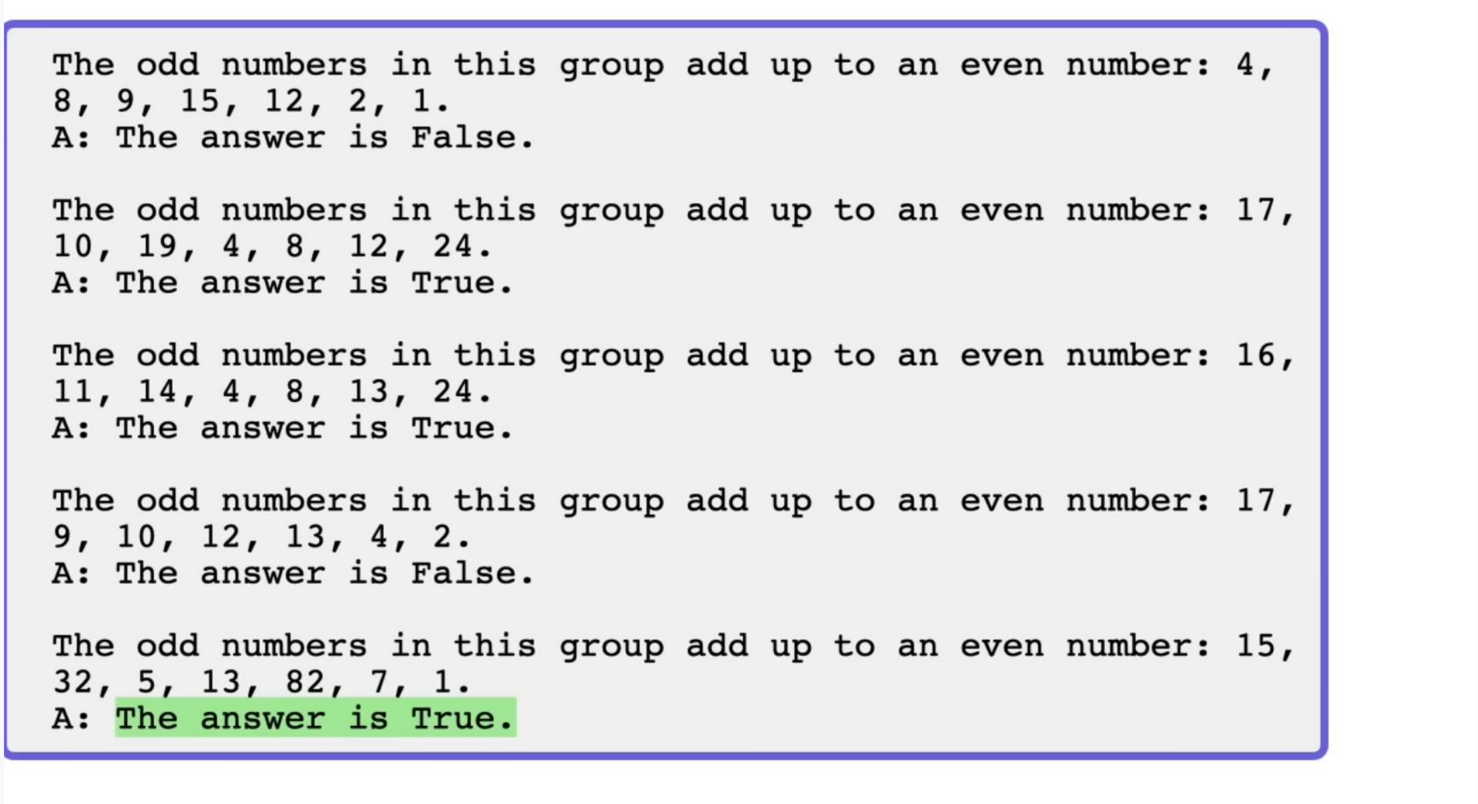
- Temperature and top_p are two important parameters
  temperature: determine the level of randomness and creativity in the generated text.
  top_p: controls the diversity of the generated text by setting a probability threshold for the next word choice.

- Keep them low if look for exact answers;
- keep them high if look for diverse responses (e.g. poetry).

- Change one of them at the same time.

3.1 Few-shot prompts

Provide exemplars in the prompters to steer the model towards better performance.



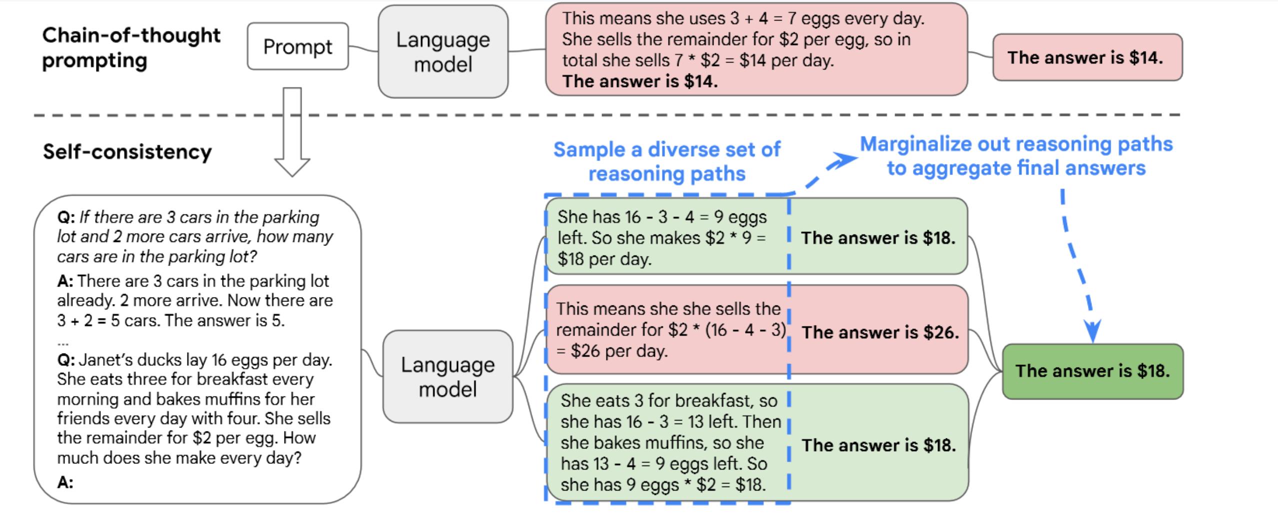
```
The odd numbers in this group add up to an even number: 4,
8, 9, 15, 12, 2, 1.
A: The answer is False.

The odd numbers in this group add up to an even number: 17,
10, 19, 4, 8, 12, 24.
A: The answer is True.

The odd numbers in this group add up to an even number: 16,
11, 14, 4, 8, 13, 24.
A: The answer is True.

The odd numbers in this group add up to an even number: 17,
9, 10, 12, 13, 4, 2.
A: The answer is False.

The odd numbers in this group add up to an even number: 15,
32, 5, 13, 82, 7, 1.
A: The answer is True.
```

3.2 Chain-of-thought CoT prompting (Reason)

- Add “Let’s think step by step” to the original prompt.
- Great for tasks that requiring reasoning.
- We can combine this with few-shot prompts;
- We can do zero-shot CoT when examplars are not available.

3.3 Self-consistency

- For more complicated tasks.
- Aims to improve on the naïve greedy decoding used in chain-of-thought prompting
- Sample multiple, diverse reasoning paths through few-shot CoT,
- Use the generations to select the most consistent answer.
- (doing majority voting to select the most reliable answer) graph source（Wang, et.al., 2022）



Chain-of-thought prompting
Prompt
Language model
Self-consistency
Language model
Sample a diverse set of reasoning paths
Marginalize out reasoning paths to aggregate final answers

This means she uses 3 + 4 = 7 eggs every day. She sells the remainder for $2 per egg, so in total she sells 7 * $2 = $14 per day. **The answer is $14.**

The answer is $14.

Q: *If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?*

A: There are 3 cars in the parking lot already. 2 more arrive. Now there are 3 + 2 = 5 cars. The answer is 5.

...

Q: Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder for $2 per egg. How much does she make every day?

A:

She has 16 - 3 - 4 = 9 eggs left. So she makes $2 * 9 = $18 per day. **The answer is $18.**

This means she she sells the remainder for $2 * (16 - 4 - 3) = $26 per day. **The answer is $26.**

She eats 3 for breakfast, so she has 16 - 3 = 13 left. Then she bakes muffins, so she has 13 - 4 = 9 eggs left. So she has 9 eggs * $2 = $18. **The answer is $18.**

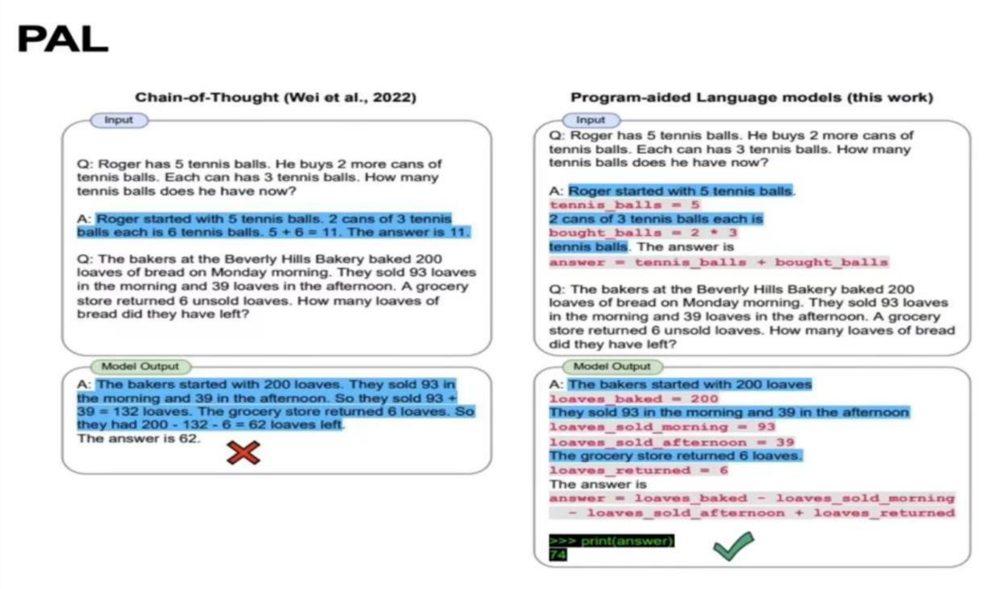
The answer is $18.

3.4 PAL (Reason with Python interpreter)

Program-aided language model:

- Sometimes CoT is not enough as it depends only on the generated text from the model.
- PAL uses an LLM to read problems and generate programs as the intermediate reasoning steps (Python interpreter) .
- Rely on external programs or scripts to execute specific tasks, rather than trying to solve them entirely on their own.
- Allows them to leverage the functionality of other software tools and libraries, making their problem-solving capabilities more robust and efficient.

3.4 Knowledge generation prompting/PAL (Reason with Python interpreter)



PAL

Chain-of-Thought (Wei et al., 2022)

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 tennis balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: The bakers at the Beverly Hills Bakery baked 200 loaves of bread on Monday morning. They sold 93 loaves in the morning and 39 loaves in the afternoon. A grocery store returned 6 unsold loaves. How many loaves of bread did they have left?

A: The bakers started with 200 loaves. They sold 93 in the morning and 39 in the afternoon. So they sold 93 + 39 = 132 loaves. The grocery store returned 6 loaves. So they had 200 - 132 - 6 = 62 loaves left.
The answer is 62. ✗

Program-aided Language models (this work)

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 tennis balls.
`tennis_balls = 5`
2 cans of 3 tennis balls each is
`bought_balls = 2 * 3`
tennis balls. The answer is
`answer = tennis_balls + bought_balls`

Q: The bakers at the Beverly Hills Bakery baked 200 loaves of bread on Monday morning. They sold 93 loaves in the morning and 39 loaves in the afternoon. A grocery store returned 6 unsold loaves. How many loaves of bread did they have left?

A: The bakers started with 200 loaves
`loaves_baked = 200`
They sold 93 in the morning and 39 in the afternoon
`loaves_sold_morning = 93`
`loaves_sold_afternoon = 39`
The grocery store returned 6 loaves.
`loaves_returned = 6`
The answer is
`answer = loaves_baked - loaves_sold_morning`
`  - loaves_sold_afternoon + loaves_returned`

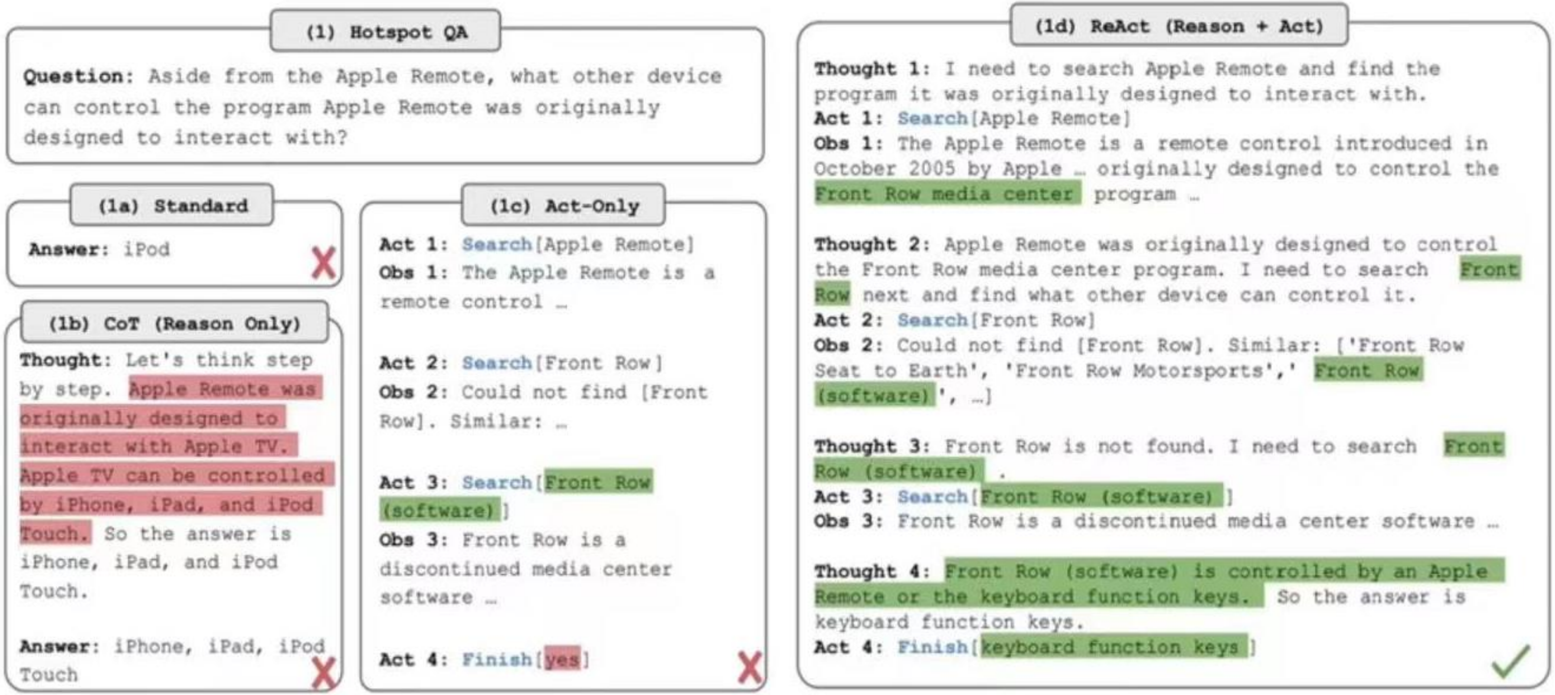
```
>>> print(answer)
74
```
✓

Source: PAL: *Program-aided Language Models*

3.5 ReAct (Reason+Act)

- a framework where LLMs are used to generate both reasoning traces and task-specific action in an interleaved manner.
- Generating reasoning traces allow the model to induce, track, and update action plans, and even handle exceptions(using the capability within the model);
- the action step allows to interface with and gather information from external sources such as knowledge bases or environments (external sources).

3.5 ReAct (Reason+Act)



(1) Hotspot QA

Question: Aside from the Apple Remote, what other device can control the program Apple Remote was originally designed to interact with?

(1a) Standard

Answer: iPod ✗

(1b) CoT (Reason Only)

Thought: Let's think step by step. Apple Remote was originally designed to interact with Apple TV. Apple TV can be controlled by iPhone, iPad, and iPod Touch. So the answer is iPhone, iPad, and iPod Touch.

Answer: iPhone, iPad, iPod Touch ✗

(1c) Act-Only

Act 1: Search[Apple Remote]
Obs 1: The Apple Remote is a remote control …

Act 2: Search[Front Row]
Obs 2: Could not find [Front Row]. Similar: …

Act 3: Search[Front Row (software)]
Obs 3: Front Row is a discontinued media center software …

Act 4: Finish[yes] ✗

(1d) ReAct (Reason + Act)

Thought 1: I need to search Apple Remote and find the program it was originally designed to interact with.
Act 1: Search[Apple Remote]
Obs 1: The Apple Remote is a remote control introduced in October 2005 by Apple … originally designed to control the Front Row media center program …

Thought 2: Apple Remote was originally designed to control the Front Row media center program. I need to search Front Row next and find what other device can control it.
Act 2: Search[Front Row]
Obs 2: Could not find [Front Row]. Similar: ['Front Row Seat to Earth', 'Front Row Motorsports',' Front Row (software) ', …]

Thought 3: Front Row is not found. I need to search Front Row (software) .
Act 3: Search[Front Row (software)]
Obs 3: Front Row is a discontinued media center software …

Thought 4: Front Row (software) is controlled by an Apple Remote or the keyboard function keys. So the answer is keyboard function keys.
Act 4: Finish[keyboard function keys] ✓

Source: *ReAct: Synergizing Reasoning and Acting in Language Models*

Tools

Tools, libraries, and platforms.

Capabilities include:

a. Developing and experimenting with prompts

b. Evaluating prompts

c. Versioning and deploying prompts.

List of tools:

https://github.com/dair-ai/Prompt-Engineering-Guide#tools--libraries

Data-augmented Generation: Langchain

QA with sources

Summarization using sources

Data-Augmented Generation

For many real-world applications there is a need to augment the generation of a model by incorporating external data

Steps involved:

Fetching relevant data

Augmenting prompts with the retrieved data as context

4.2.2 Data-augmented Generation

External data can include:

- Document stores
- APIs
- Databases
- User provided data

LangChain Documentation

https://langchain.readthedocs.io/en/latest/getting_started/getting_started.html/

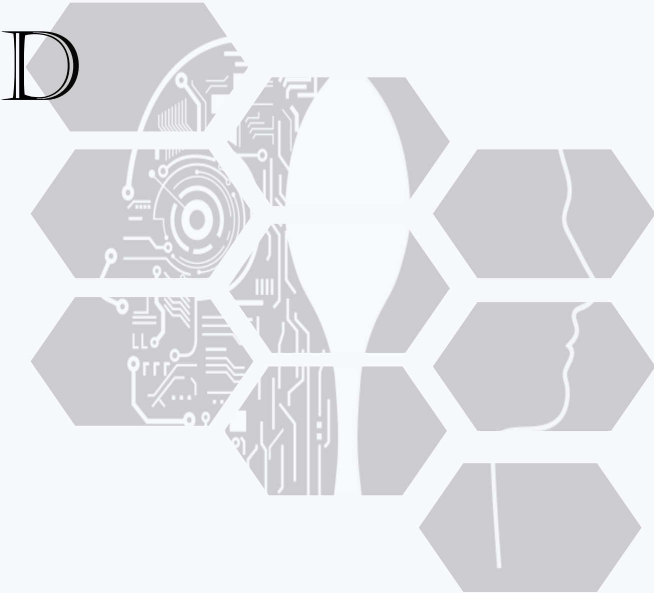
04

FUTURE DIRECTIONS

Tool name	link	Introduction	Task Type	Github stars	cost
PromptPerpect	https://promptperfect.jina.ai/	automatically optimizing your prompts for ChatGPT, GPT-3.5, DALL-E 2,StableDiffusion and MidJourney	ALL	N/A	Start at $9.99/month
Langchain	https://python.langchain.com/en/latest/	LangChain is a next-generation framework that empowers developers to build advanced applications using language models. By enabling language models to connect to other data sources and interact with their environment, LangChain takes application development to the next level.	ALL	18.4k	
Open Playground	https://nat.dev/		ALL. Test out self-consistency		
Dyno	https://trydyno.com/	Dual Wield Token Usage Save Prompts for Generate Multiple Outputs	ALL		

05

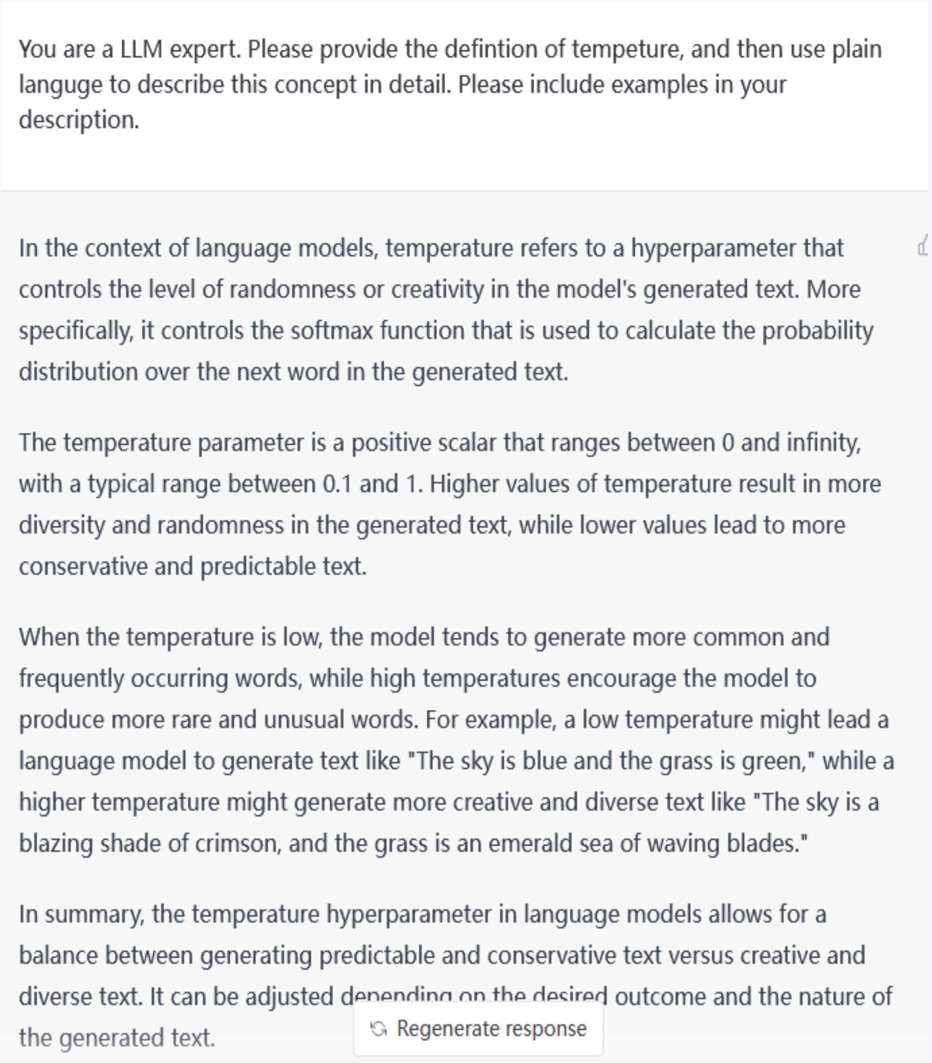
RESOURCES AND REFERENCES



7.1 References and Resources

1. Open AI official site: https://platform.openai.com/docs/introduction/key-concepts
2. best practice provided by Open AI: https://help.openai.com/en/articles/6654000-best-practices-for-prompt-engineering-with-openai-api
3. Prompt engineering overview from Elvis Savaria， dair AI
   https://www.youtube.com/watch?v=dOxUroR57xs
   https://www.promptingguide.ai/
   https://github.com/dair-ai/Prompt-Engineering-Guide#tools--libraries
4. co:here prompt engineering guide
   https://docs.cohere.ai/docs/prompt-engineering
5. 10 Amazing Resources For Prompt Engineering, ChatGPT, and GPT-
   https://medium.com/tales-of-tomorrow/10-amazing-resources-for-prompt-engineering-chatgpt-and-gpt-3-ad84dd26bfc7
6. The excel document I shared: https://github.com/f/awesome-chatgpt-prompts/blob/main/prompts.csv

THANKS FOR LISTENING!



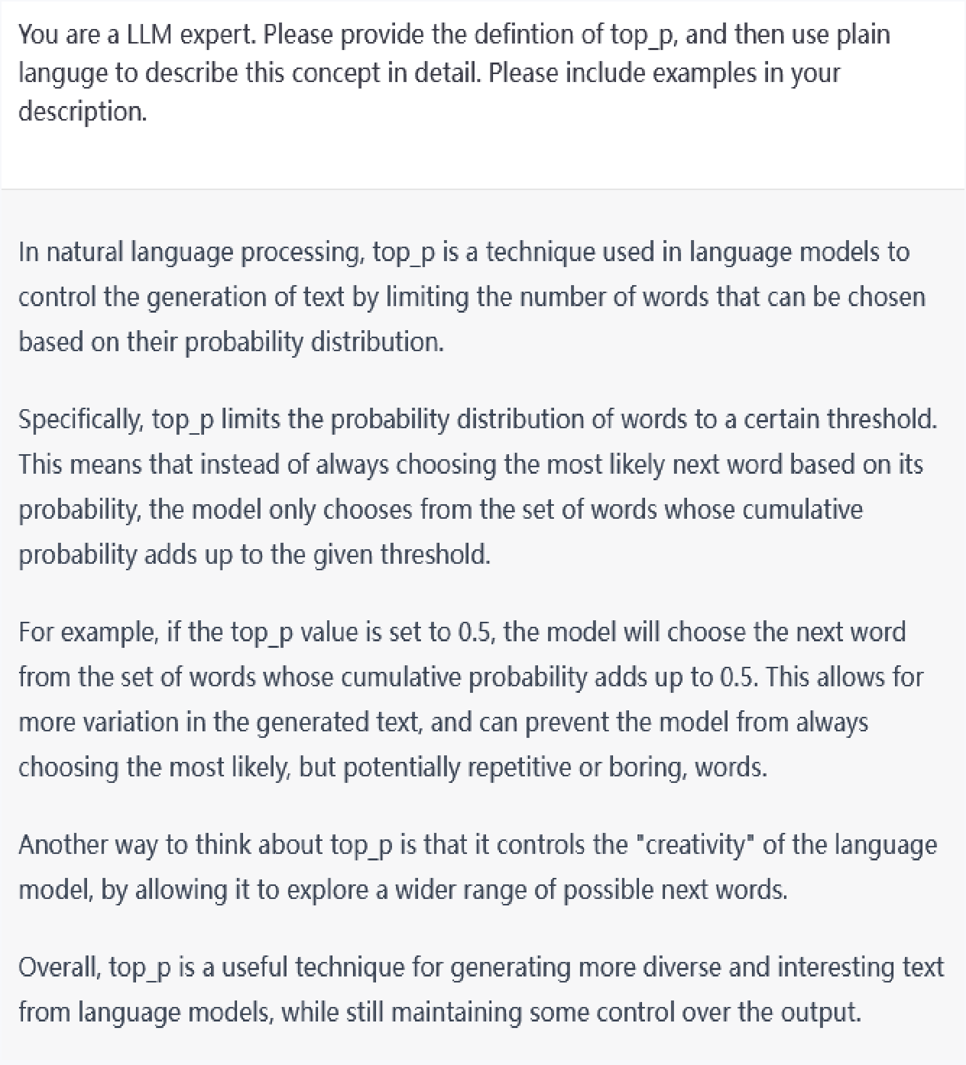
You are a LLM expert. Please provide the defintion of tempeture, and then use plain languge to describe this concept in detail. Please include examples in your description.

In the context of language models, temperature refers to a hyperparameter that controls the level of randomness or creativity in the model's generated text. More specifically, it controls the softmax function that is used to calculate the probability distribution over the next word in the generated text.

The temperature parameter is a positive scalar that ranges between 0 and infinity, with a typical range between 0.1 and 1. Higher values of temperature result in more diversity and randomness in the generated text, while lower values lead to more conservative and predictable text.

When the temperature is low, the model tends to generate more common and frequently occurring words, while high temperatures encourage the model to produce more rare and unusual words. For example, a low temperature might lead a language model to generate text like "The sky is blue and the grass is green," while a higher temperature might generate more creative and diverse text like "The sky is a blazing shade of crimson, and the grass is an emerald sea of waving blades."

In summary, the temperature hyperparameter in language models allows for a balance between generating predictable and conservative text versus creative and diverse text. It can be adjusted depending on the desired outcome and the nature of the generated text.



You are a LLM expert. Please provide the defintion of top_p, and then use plain languge to describe this concept in detail. Please include examples in your description.

In natural language processing, top_p is a technique used in language models to control the generation of text by limiting the number of words that can be chosen based on their probability distribution.

Specifically, top_p limits the probability distribution of words to a certain threshold. This means that instead of always choosing the most likely next word based on its probability, the model only chooses from the set of words whose cumulative probability adds up to the given threshold.

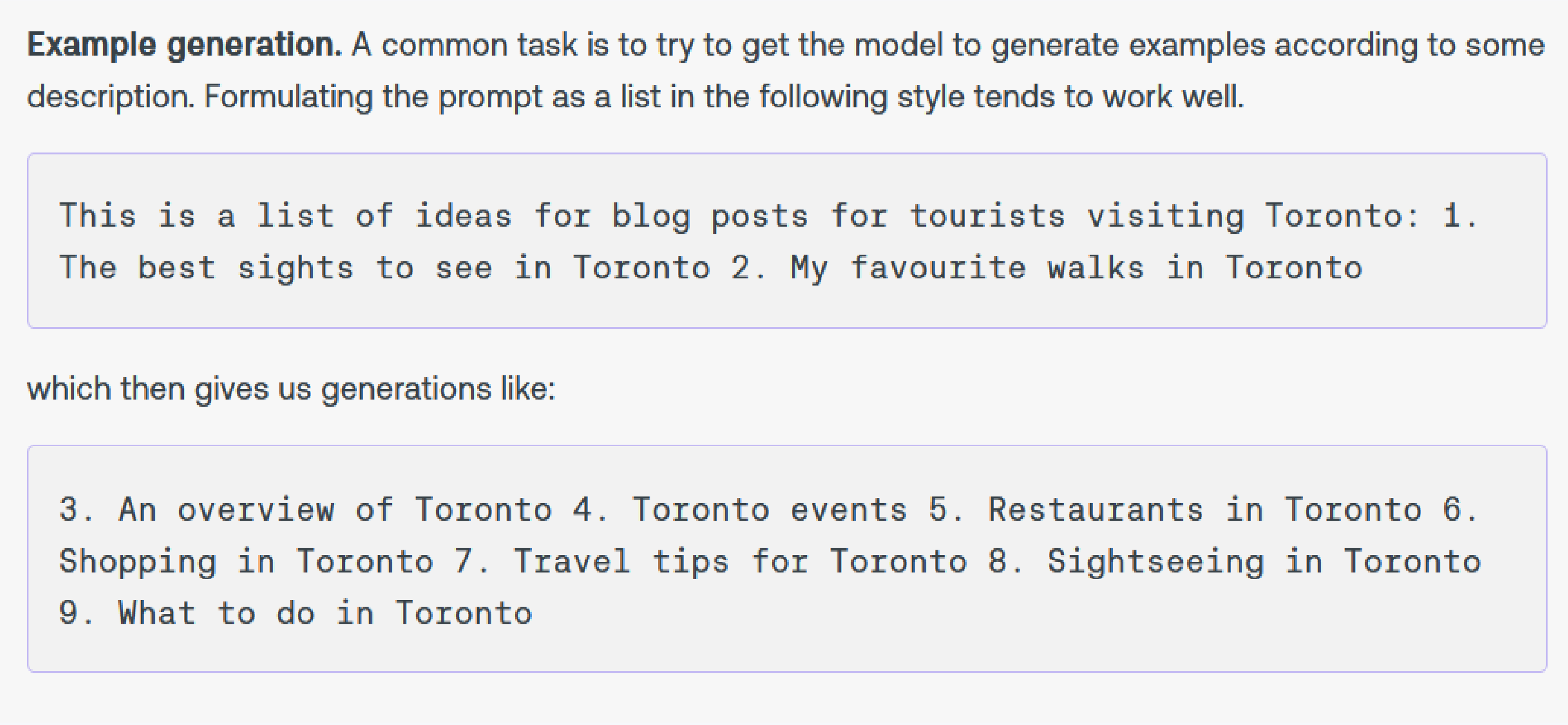
For example, if the top_p value is set to 0.5, the model will choose the next word from the set of words whose cumulative probability adds up to 0.5. This allows for more variation in the generated text, and can prevent the model from always choosing the most likely, but potentially repetitive or boring, words.

Another way to think about top_p is that it controls the "creativity" of the language model, by allowing it to explore a wider range of possible next words.

Overall, top_p is a useful technique for generating more diverse and interesting text from language models, while still maintaining some control over the output.

5.2 generating additional examples given a few existing examples.

key principles: setting; examples.



Example generation. A common task is to try to get the model to generate examples according to some description. Formulating the prompt as a list in the following style tends to work well.

```
This is a list of ideas for blog posts for tourists visiting Toronto: 1.
The best sights to see in Toronto 2. My favourite walks in Toronto
```

which then gives us generations like:

```
3. An overview of Toronto 4. Toronto events 5. Restaurants in Toronto 6.
Shopping in Toronto 7. Travel tips for Toronto 8. Sightseeing in Toronto
9. What to do in Toronto
```

5.1 generating keywords based on a given passage

key principles: setting; examples.

Example:

This is a bot that automatically finds the most important keyword for a given text passage. Task description, with the setting clearly illustrated.

Text: "John von Neumann (/vɒn ˈnɔɪmən/; Hungarian: Neumann János Lajos, pronounced [ˈnɒjmɒn ˈjaːnoʃ ˈlɒjoʃ]; December 28, 1903 – February 8, 1957) was a Hungarian-American mathematician, physicist, computer scientist, engineer and polymath. Von Neumann was generally regarded as the foremost mathematician of his time[2] and said to be "the last representative of the great mathematicians".[3] He integrated pure and applied sciences.“

Most important key word: "John von Neumann" Example 1.

Text: "Some scholars consider feminist campaigns to be a main force behind major historical societal changes for women's rights, particularly in the West, where they are near-universally credited with achieving women's suffrage, gender-neutral language, reproductive rights for women (including access to contraceptives and abortion), and the right to enter into contracts and own property.[9] Although feminist advocacy is, and has been, mainly focused on women's rights, some feminists argue for the inclusion of men's liberation within its aims, because they believe that men are also harmed by traditional gender roles.[10] Feminist theory, which emerged from feminist movements, aims to understand the nature of gender inequality by examining women's social roles and lived experience; feminist theorists have developed theories in a variety of disciplines in order to respond to issues concerning gender.“

Most important key word: "Feminism" **Example 2.**

Text: "Guido van Rossum began working on Python in the late 1980s, as a successor to the ABC programming language, and first released it in 1991 as Python 0.9.0.[31] Python 2.0 was released in 2000 and introduced new features, such as list comprehensions and a garbage collection system using reference counting and was discontinued with version 2.7.18 in 2020.[32] Python 3.0 was released in 2008 and was a major revision of the language that is not completely backward-compatible and much Python 2 code does not run unmodified on Python 3."

Most important key word: current input.