Prompting

Prompting is the process of crafting an input that guides a language model (LM), like GPT, to produce desired outputs. In essence, it's how humans communicate with AI to perform tasks, answer questions, generate text, translate languages, summarize content, and much more.

The effectiveness of AI outputs heavily depends on how well a prompt is constructed. This led to the emergence of Prompt Engineering—the skill and art of designing optimal prompts for large language models (LLMs).

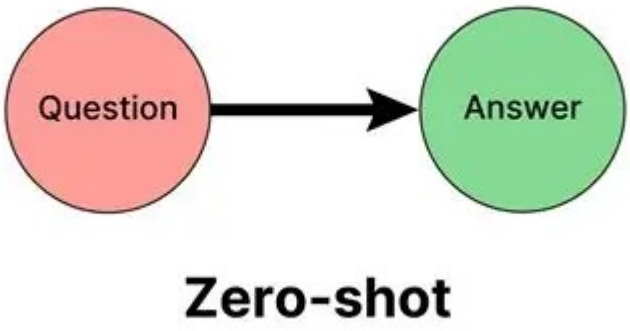
Prompting can be broadly classified into various types based on how much context or guidance is provided: zero-shot, one-shot, few-shot, chain-of-thought prompting, and prompt tuning. Each technique has distinct advantages and is suitable for different scenarios.

Prompting is the process of crafting an input (instruction or query) that guides a language model like GPT-4 to generate the desired output. In simpler terms, it's how humans "talk" to AI to complete tasks such as:

- Answering questions
- Translating languages
- Summarizing documents
- Writing code
- Solving math problems
- Generating creative content

Types of Prompting

3.1 Zero-Shot Prompting



In this approach, the prompt contains no prior examples—just a task description or query.

Example:

"Summarize the following text: *'Artificial Intelligence is transforming industries...*'"

Use Cases:

- Factual Q&A
- Basic summaries
- Quick translations
- Keyword extraction

Key Benefits:

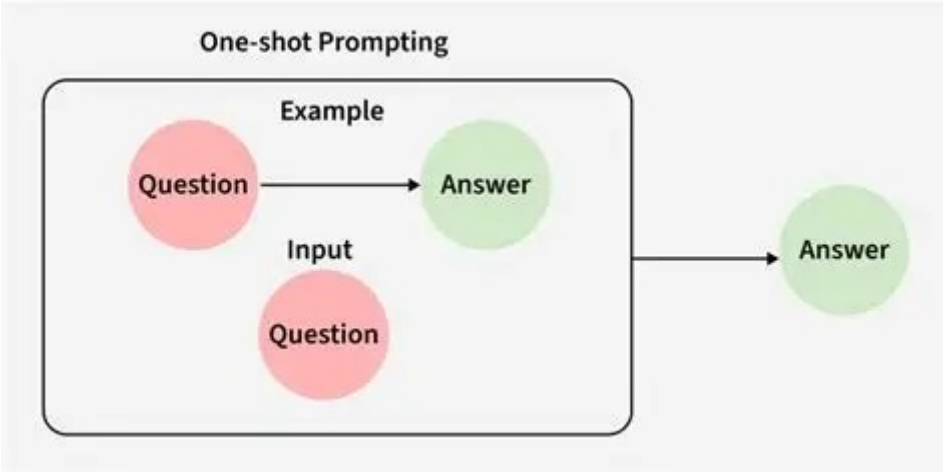
- Extremely simple to use
- Works for general knowledge tasks

- Ideal when you don’t have example data

Limitations:

- Performance varies depending on how well the model has seen similar tasks during training
- Poor handling of ambiguity or novel tasks

3.2 One-Shot Prompting



This method includes one example to guide the model in understanding the format or style required.

Example:

"English: I love music → French: J’aime la musique
English: I like cats → French:"

Use Cases:

- Pattern recognition
- Basic classification with formatting
- Language translation or template generation

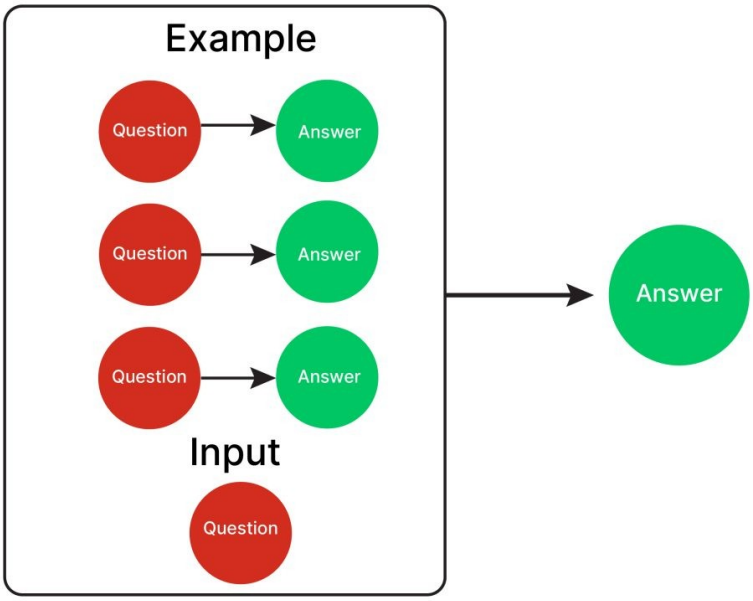
Key Benefits:

- Offers basic clarity on structure
- Useful for non-standard or domain-specific formats

Limitations:

- One example might not generalize well to unseen inputs
- Edge cases may confuse the model

3.3 Few-Shot Prompting



Few-shot prompting offers several examples before requesting a new output, leveraging the model's pattern recognition.

Example:

"Sentiment classification:
Text: 'I hated the ending.' → Sentiment: Negative
Text: 'Absolutely loved it!' → Sentiment: Positive
Text: 'It was okay, nothing special.' → Sentiment:"

Use Cases:

- Content classification
- Code generation (e.g., Python functions)
- Language translation
- Extracting structured information

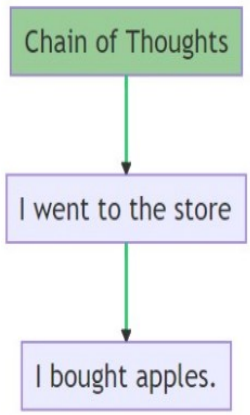
Benefits:

- Higher accuracy than zero or one-shot
- Easy to implement with no model retraining

Challenges:

- Limited by input token length
- Example quality heavily impacts performance
- Can become verbose or hard to scale

3.4 Chain-of-Thought (CoT) Prompting



CoT prompting includes intermediate reasoning steps, helping the model break down the task and generate a final answer.

Example:

> Q: If Jack has 3 apples and buys 2 more, then gives 1 away, how many apples does he have?
> A: Jack starts with 3 apples. He buys 2 more, now he has 5. He gives away 1. So, he has 4 apples.

Use Cases:

- Math word problems
- Logic reasoning
- Commonsense tasks
- Code explanation

Variants:

- **Manual CoT**: Human writes the reasoning steps
- **Auto-CoT**: Model generates reasoning chain from a few examples

Benefits:

- Improves factual accuracy and reasoning

- Makes model’s thinking transparent
- Helps avoid hallucinations in complex tasks

Drawbacks:

- Long responses
- Might produce logically wrong steps with confident tone

3.5 Prompt Tuning (Soft Prompting)

Prompt tuning is an **automated** way to optimize prompts. It replaces manual prompt design with **learnable vectors** (called “soft prompts”), which are trained using gradient descent.

These soft prompts are not natural language strings but are embedded vectors inserted into the model’s input layer.

Use Cases:

- Domain-specific applications (legal, medical, finance)
- Customer service chatbots
- Enterprise-level deployments of LLMs

Key Differences from Manual Prompting:

- Doesn’t involve rewriting prompts
- Instead, trains a small parameter set (typically <1% of total model size)
- Requires computational training but is more efficient than full fine-tuning

Advantages:

- Consistent, reproducible behavior
- Less risk of prompt forgetting or drift
- Ideal for repeated or production-grade tasks

Challenges:

- Not human-readable
- Needs ML knowledge and infrastructure
- Works only with tunable models (e.g., GPT-2, T5, etc.)