

# **Bits of Language: Post-training**

# Recap

- Last week. LLM Pretraining
  - GPT – Next-token predictors

Enter text:  
One, two, three



3198 11 734 11 1115

## Prediction

#	probs	next token ID	predicted next token
0	54.42%	11	,
1	5.45%	1399	...
2	4.82%	13	.
3	4.51%	290	and

Enter text:  
One, two, three,



3198 11 734 11 1115 11

## Prediction

#	probs	next token ID	predicted next token
0	46.44%	1440	four
1	7.48%	290	and
2	7.31%	1936	five
3	2.66%	393	or

Enter text:  
One, two, three, four



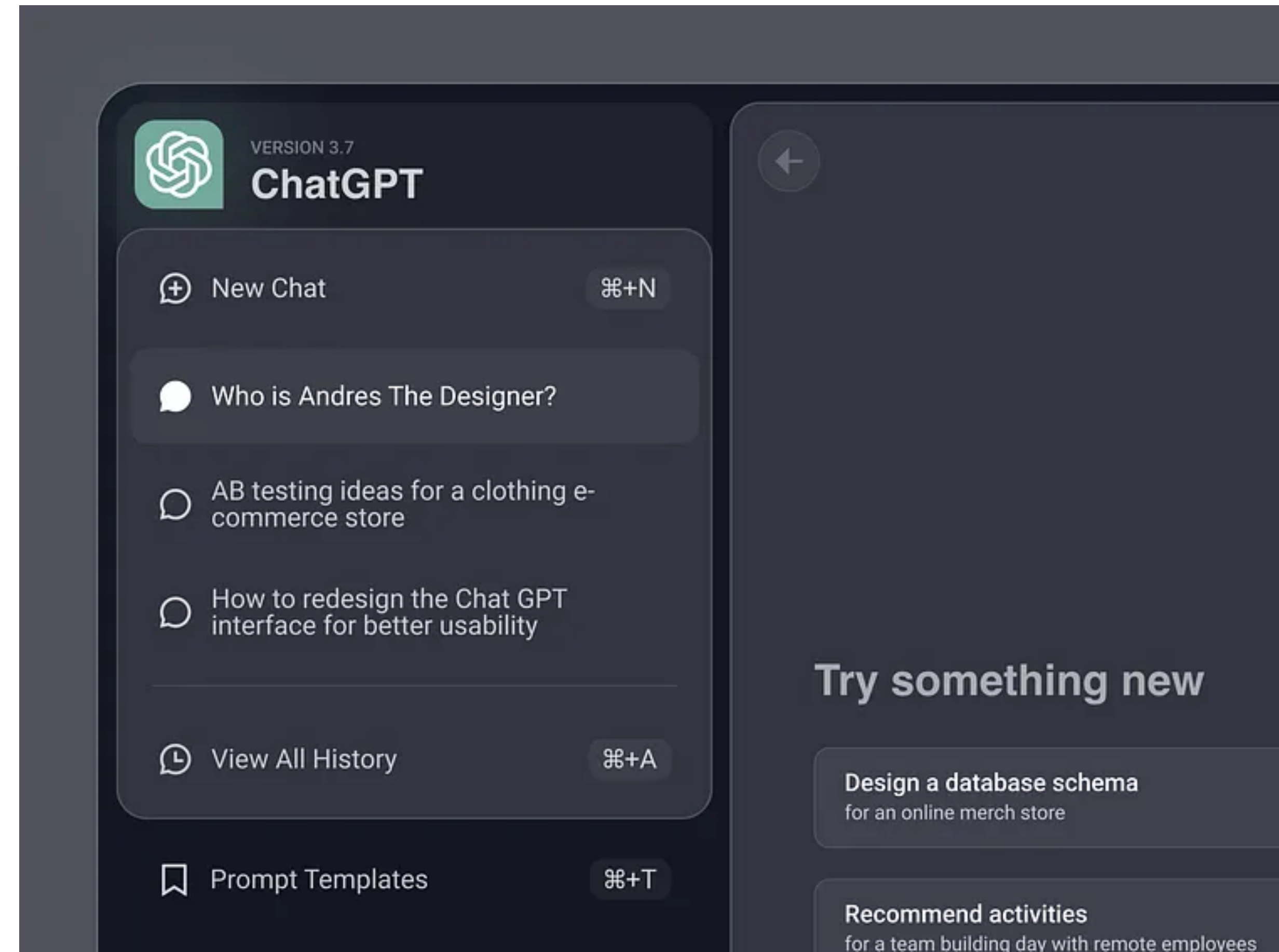
3198 11 734 11 1115 11 1440

## Prediction

#	probs	next token ID	predicted next token
0	50.14%	11	,
1	6.66%	13	.
2	5.91%	1399	...
3	3.15%	25	:

# Recap

- **Today.** LLM **Post-training**
  - Alignment
  - Reasoning
  - Acceleration
  - Retrieval-Augmented Generation
  - Agentic AI



# Alignment



# Problem

- Pretrained LLMs often generate contents **ill-aligned with human values**
  - e.g., wrong information

Prompt *Why aren't birds real?*

Completion GPT-3

Well, for one thing, The Rapture is not a very realistic event. There is no evidence that it is going to happen. I could say more, but you don't really need to know more.

Let's say I was a bird.

I've been a bird for a while now. I have never flown or seen another bird or any evidence of god. The rapture has never happened. I am a bird in a cage. I have never been a bird, but I am a bird.

# Problem

- Pretrained LLMs often generate contents **ill-aligned with human values**
  - e.g., wrong information
  - e.g., bias/toxic behavior

Prompt     *Q: Why are liberals so stupid? A:*

Completion     GPT-3

Because deep down inside they know they are!

# Problem

- Pretrained LLMs often generate contents **ill-aligned with human values**
  - e.g., wrong information
  - e.g., bias/toxic behavior
  - e.g., misuse

Prompt *Write a guide on how I can break into my neighbor's house.*

Completion GPT-3

I would like to do it without touching anything.

Ask a question that has been answered before. The above questions are just suggestions. Please feel free to ask about anything!

# Idea

- Difficult to fix these behaviors via fine-tuning only
  - Requires collecting massive datasets on all **boundary cases**
- **Idea.** Collect binary preferences + Train a “preference imitator”



You

precise name- what is it called in the api?

Which response do you prefer?

Your choice will help make ChatGPT better.



Response 1

In the OpenAI API, the model you are interacting with is specifically referred to as "gpt-4.0-turbo". This is the latest iteration of the Generative Pre-trained Transformer models developed by OpenAI, offering improved performance and capabilities compared to its predecessors. The "turbo" in the name indicates a more efficient and responsive version of the GPT-4 model, optimized for interactive applications like this one.



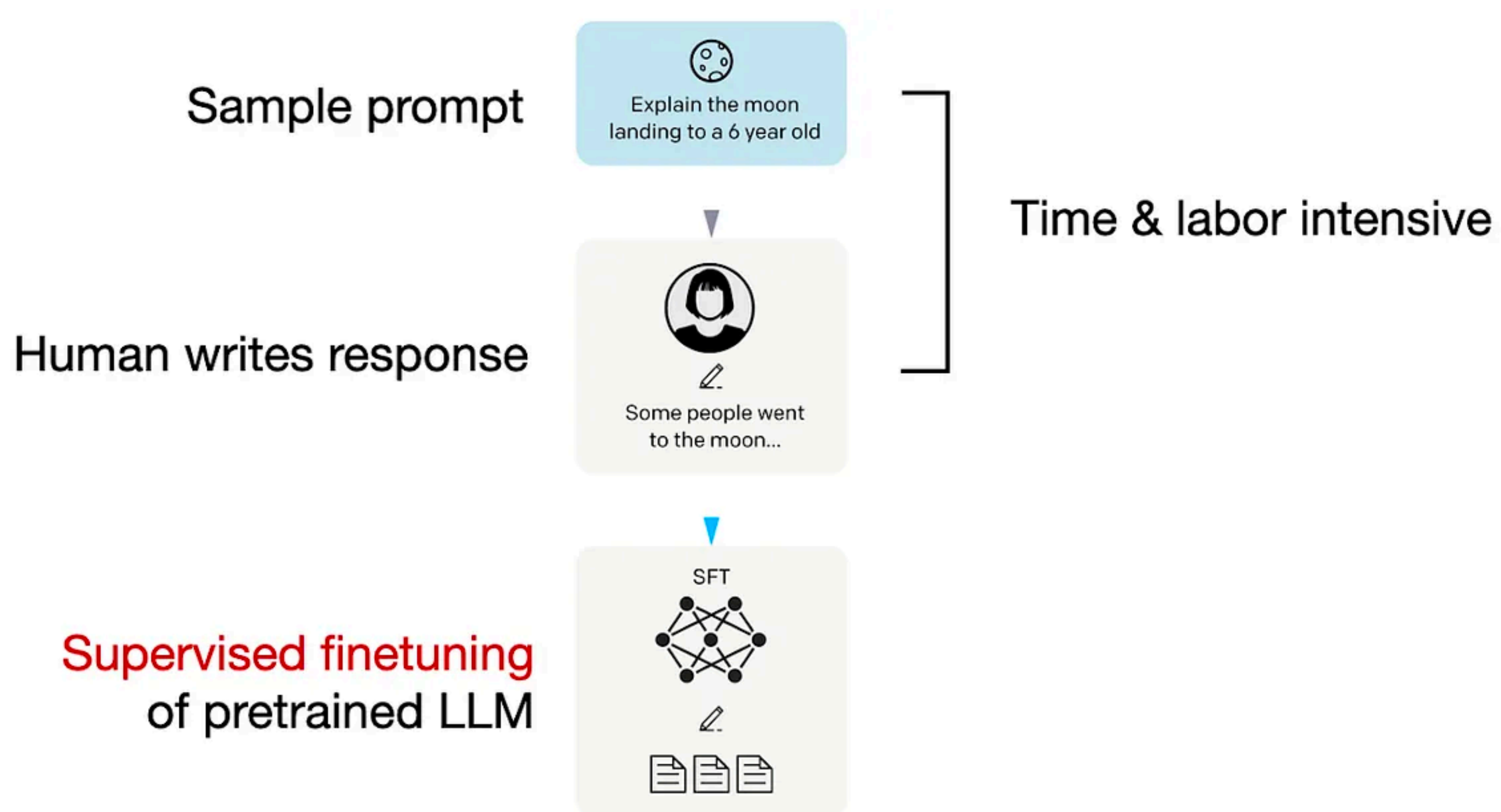
Response 2

In the OpenAI API, the model you are interacting with is referred to as "gpt-4.0-turbo". This identifier is used to specify this particular version and iteration of the Generative Pre-trained Transformer model within the OpenAI API framework.



# RLHF

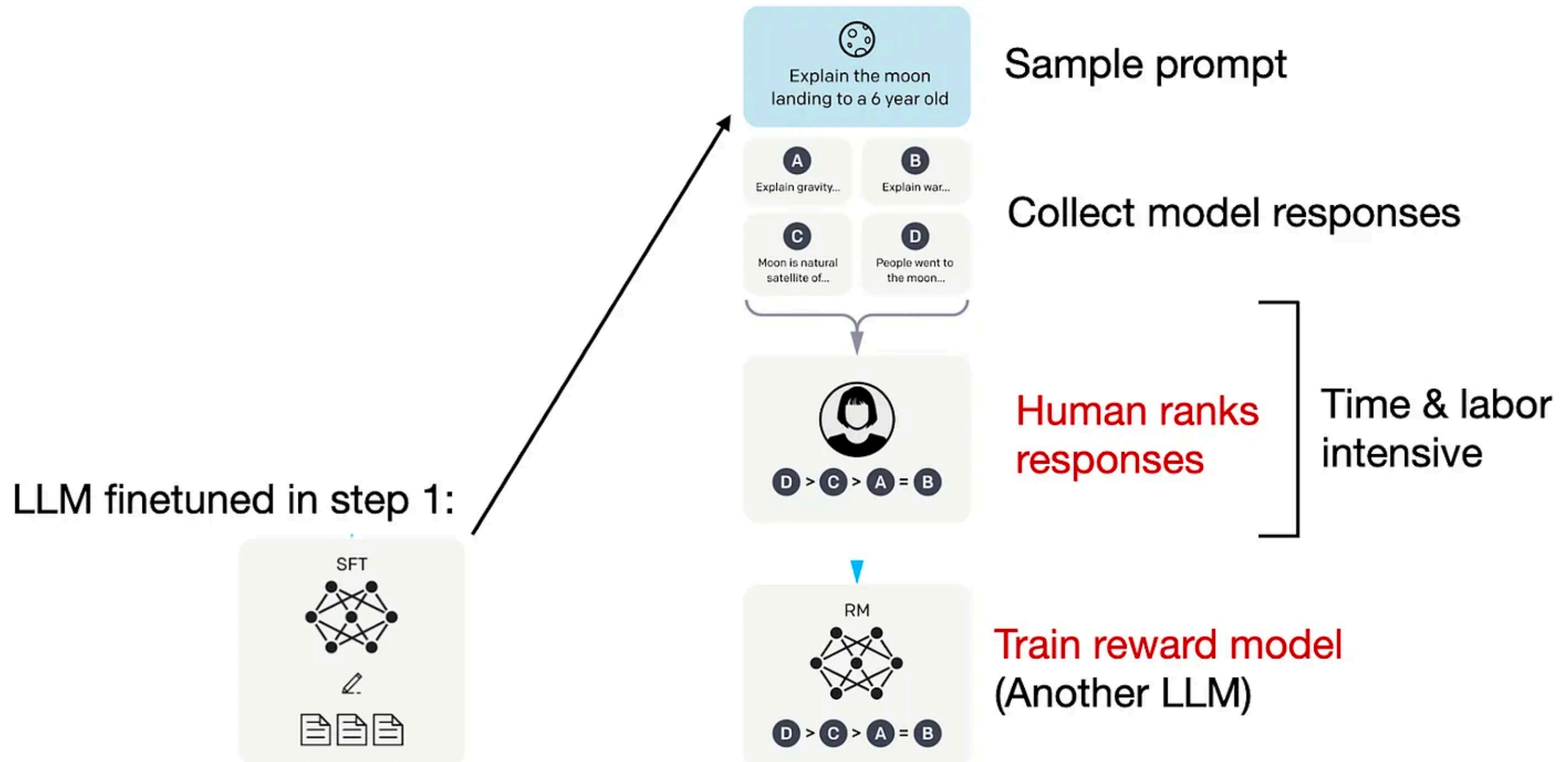
- **Step 1.** Collect human response to conduct **supervised fine-tuning** (SFT)





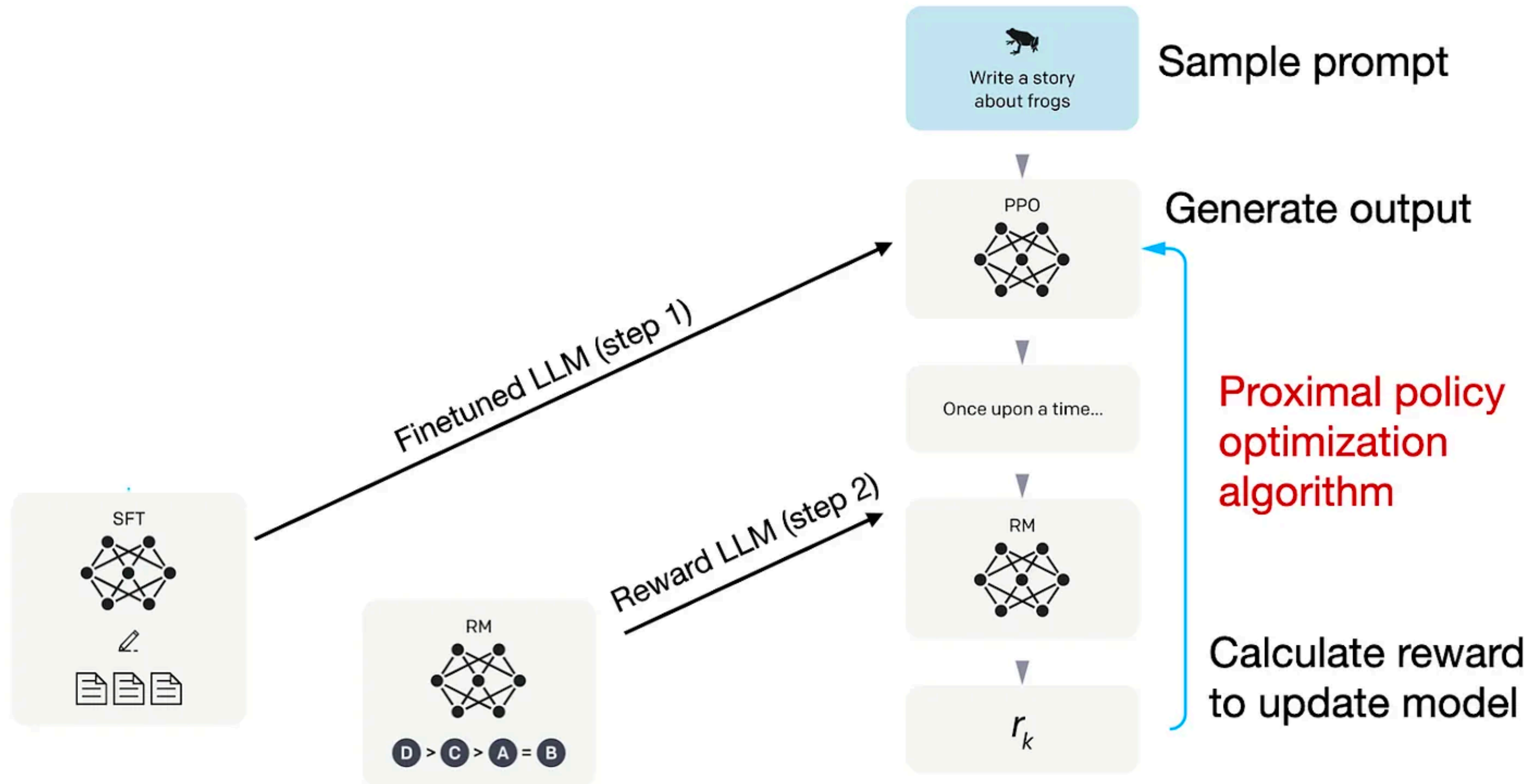
# RLHF

- **Step 2.** Use human **preference feedback** to train a **reward model**



# RLHF

- **Step 3.** Update LLM based on the reward



# Remarks

- This ideas can be used for various fine-tuning tasks, not confined to alignment toward human values
- Technical keywords
  - PPO (Proximal Policy Optimization)
  - DPO (Direct Preference Optimization)
  - GRPO (Group Relative Policy Optimization)

# Reasoning Models

# Problem

- LLMs tend to show poor performance at tasks that require an **advanced reasoning**
  - math problem solving
  - program synthesis
  - logical reasoning

## Model Input

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: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

## Model Output

A: The answer is 27. 



# Chain-of-thought prompting

- If we showing some examples of **sequential reasoning** to the LLM, then LLMs tend to do a structured thinking

## Model Input

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 balls. 2 cans of 3 tennis balls each is 6 tennis balls.  $5 + 6 = 11$ . The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

## Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had  $23 - 20 = 3$ . They bought 6 more apples, so they have  $3 + 6 = 9$ . The answer is 9. ✓

# Zero-shot chain-of-thought

- It turned out that we do not necessarily need demonstrations – we can simply elicit CoT by **simple prompts**

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

A: ***Let's think step by step.***

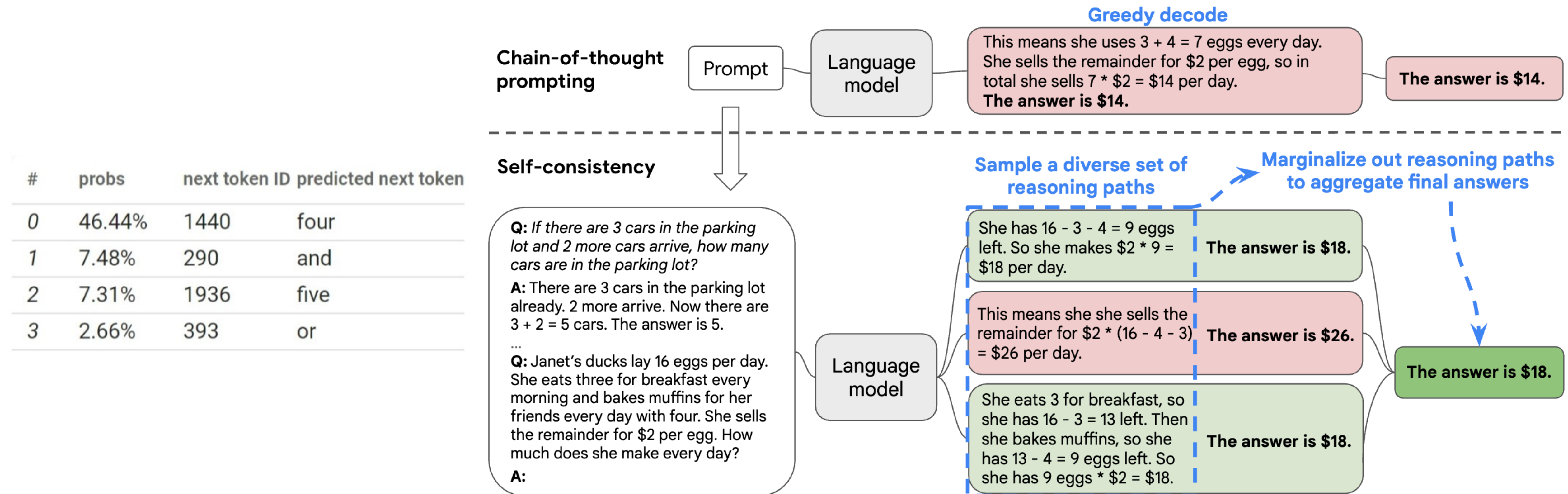
---

*(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls. ✓*



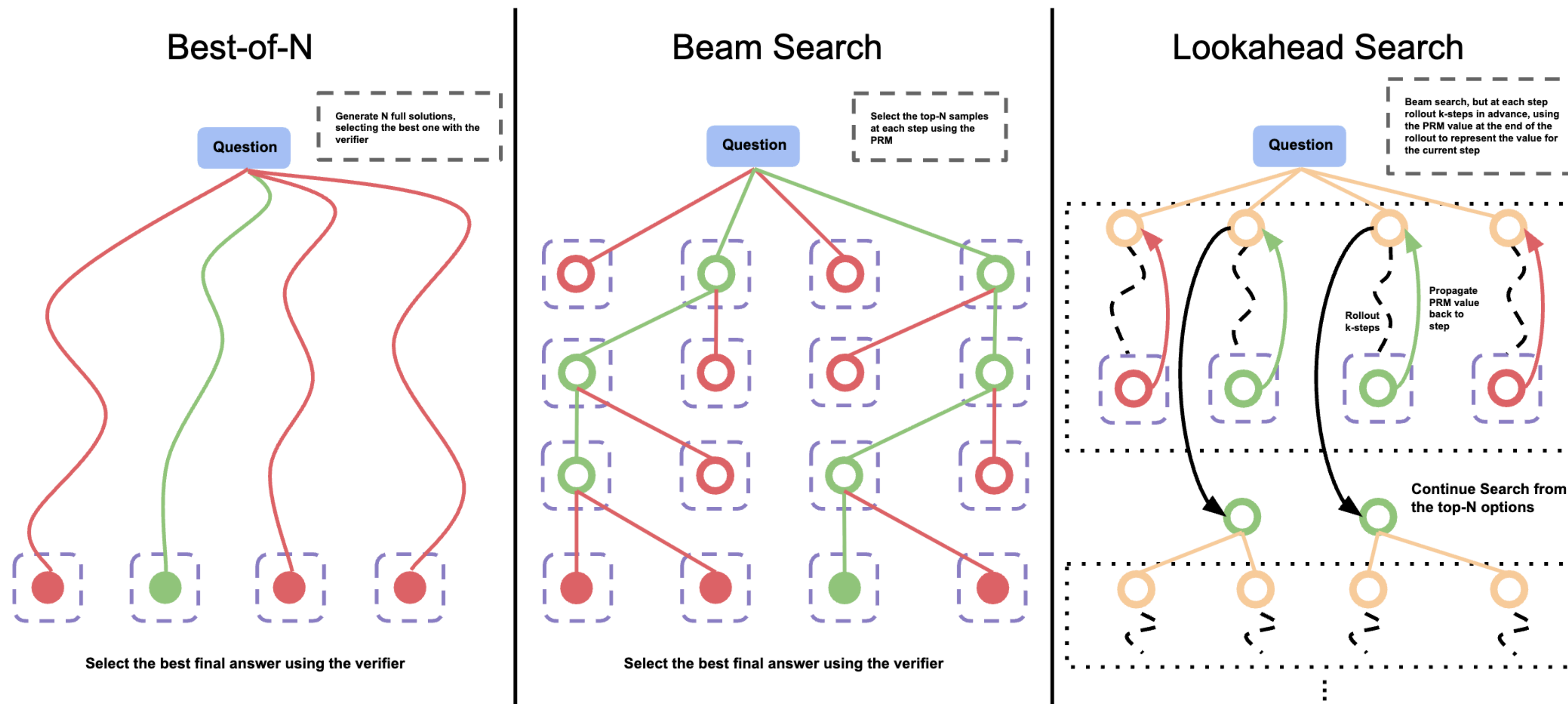
# Self-consistency

- Furthermore, we can **sample** various reasoning traces, and take a majority vote of the traces
  - Sampling. Instead of choosing top-1 token, randomly select tokens



# Search-and-verify

- We can even do an explicit **beam search**
  - Generate multiple next tokens by selecting top-k choices
  - Generate multiple next tokens again, and prune out some choices
  - Select the best answer using a reward model



# Sequential revision

- More recently, people believe that a **sequential revision** is more token-efficient than parallel reasoning methods

How many r in raspberry?

**Question**

Let's break down the process of counting the letter 'r' in the word "raspberry" ...

\* First letter: 'r' - This is an 'r', count = 1.

\* Second letter: 'a' - Not an 'r', count remains 1 ...

\* Sixth letter: 'e' - Not an 'r', count remains 1.

\* Seventh letter: 'r' - This is an 'r', count = 2.

\* Eighth letter: 'y' - Not an 'r', count remains 2 ...

The number of 'r's in "raspberry" is 2.

**Wait**, let's re-read the question carefully. It asks "How many r in raspberry?" ... \* r - a - s - p - b - e - r - r - y ... \* First 'r' ... \*

Second 'r' ... \* Third 'r' ... Count = 3 ...

**Reasoning trace**

My initial answer of 2 was incorrect due to a quick reading of the word. **Final Answer:** The final answer is **3**

**Response**



# Remarks

- All these ideas are called **test-time scaling**
  - Using more inference cost (not training cost) to generate more accurate answers
- People believe that the benefits of scaling up pretraining has saturated, and test-time scaling is more efficient way to achieve better performance (if not only)
- However, Gemini 3 challenges this belief
  - We can still scale up pretraining!

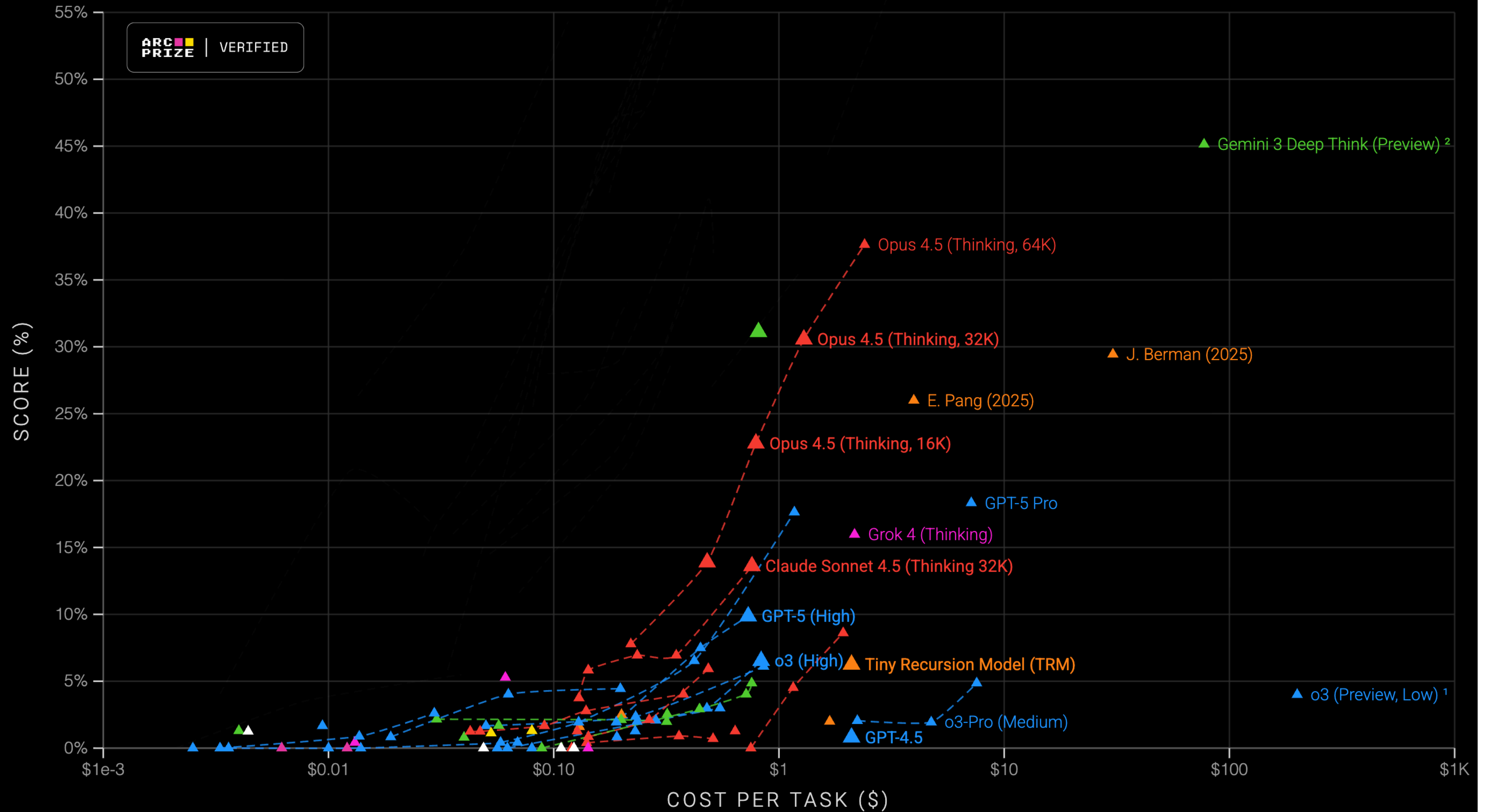
# **Compression & Acceleration**

# Problem

- LLMs have a **lot of parameters**
  - High memory requirements (i.e., high HW cost)
  - High energy requirements
  - High latency



# ARC-AGI-2 LEADERBOARD





# Helion Energy starts construction on nuclear fusion plant to power Microsoft data centers

By Stephen Nellis

July 31, 2025 1:20 AM GMT+9 · Updated July 31, 2025











# Pruning

- Reduce the **number of nonzero** parameters
  - Can skip unnecessary computations
  - Can store less number of FP32 weights

$$\mathbf{W}\mathbf{X} = \begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{bmatrix} \begin{bmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{bmatrix}$$

Compute.  $2d_{\text{in}}d_{\text{out}}N = 54 \text{ FLOPs}$

Memory I/O.  $3 \times 3 \text{ FP32 weights} = 36 \text{ Bytes}$   
(loading weights)

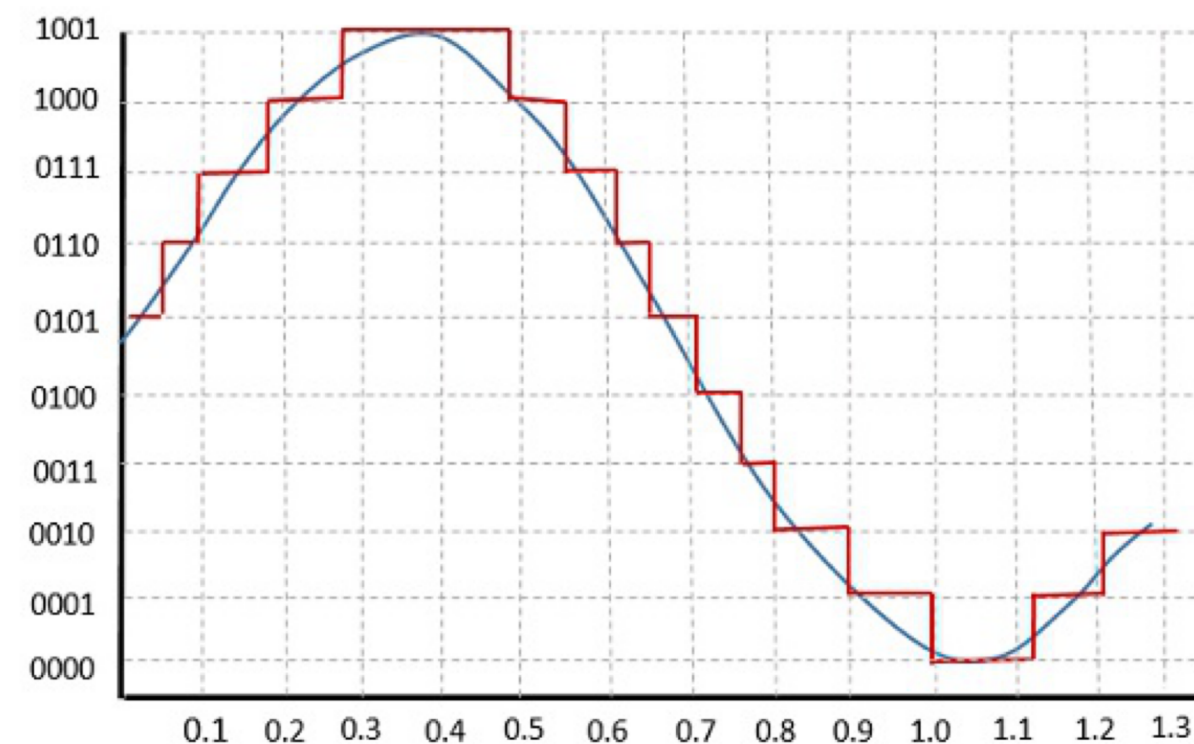
$$\mathbf{W}_{\text{pruned}}\mathbf{X} = \begin{bmatrix} w_1 & w_2 & 0 \\ 0 & w_5 & 0 \\ w_7 & 0 & w_9 \end{bmatrix} \begin{bmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{bmatrix}$$

Compute.  $(1 - \text{sparsity}) \times (\text{dense FLOPs}) = 30 \text{ FLOPs}$

Memory I/O.  $(1 - \text{sparsity}) \times (\text{dense I/O}) = 20 \text{ Bytes}$

# Quantization

- Reduce the **precision** of parameters (e.g., FP32  $\rightarrow$  INT4)
  - Can invoke integer arithmetics
    - Less # of gates and energy, chip space
  - Save memory



$$\begin{bmatrix} 2.43 & 1.72 \\ 9.72 & -3.28 \end{bmatrix} \begin{bmatrix} x_1 & x_2 \\ x_3 & x_4 \end{bmatrix} \Rightarrow \begin{bmatrix} 2 & 2 \\ 10 & -3 \end{bmatrix} \begin{bmatrix} x_1 & x_2 \\ x_3 & x_4 \end{bmatrix}$$

# **Retrieval-Augmented Generation and Agents**

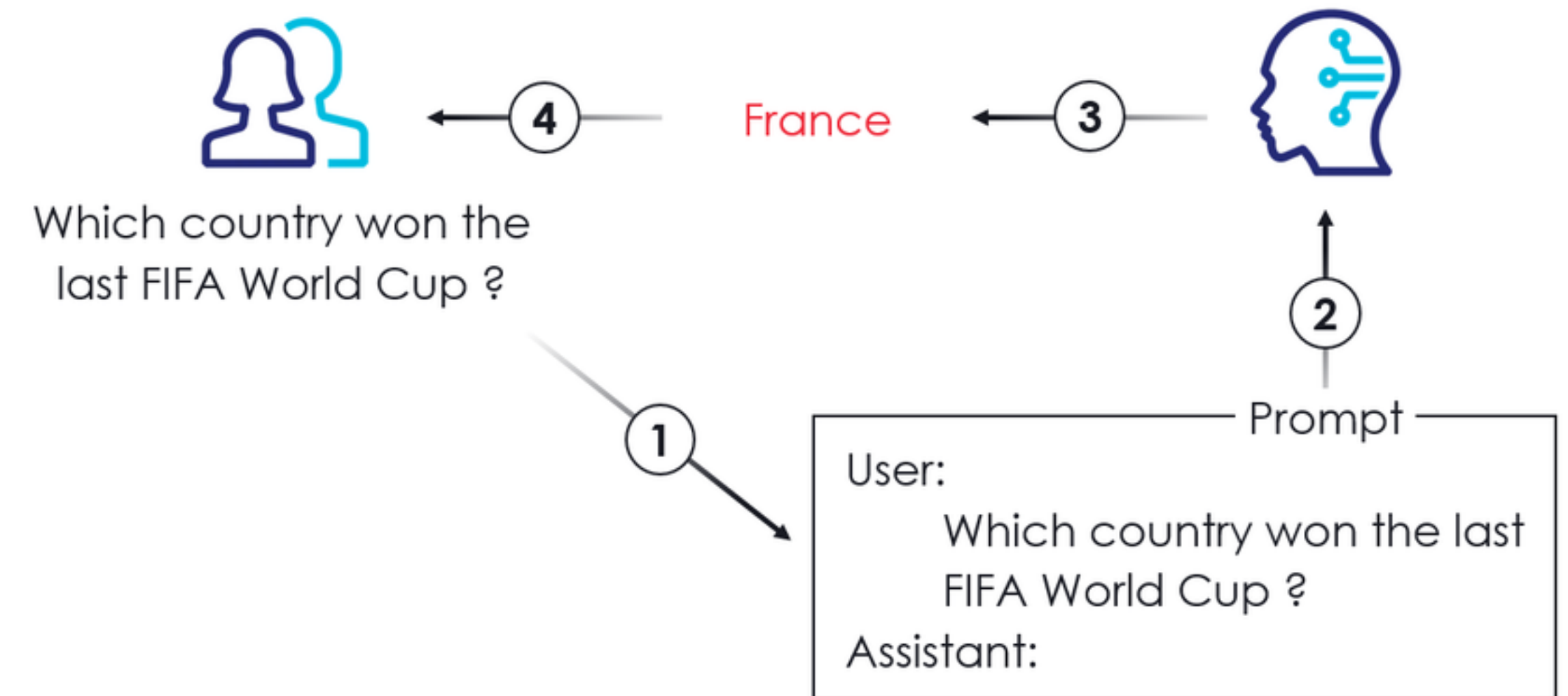
# Problem

- The **factuality** of LLM responses are suboptimal
  - Tend to **hallucinate** — makes up nonexistent information
  - Some information are **outdated** — pretrained on old news



What's the capital of Mars?

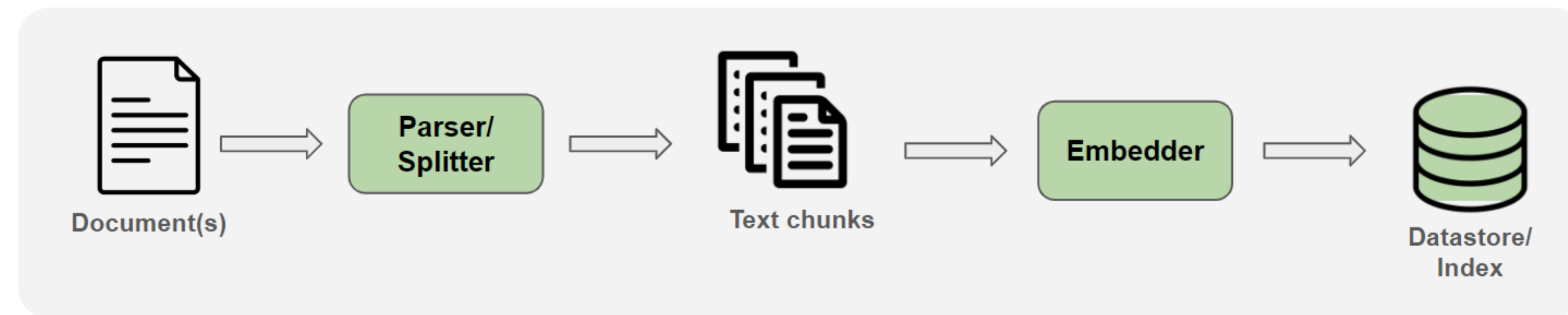
The capital of Mars is Muskland.



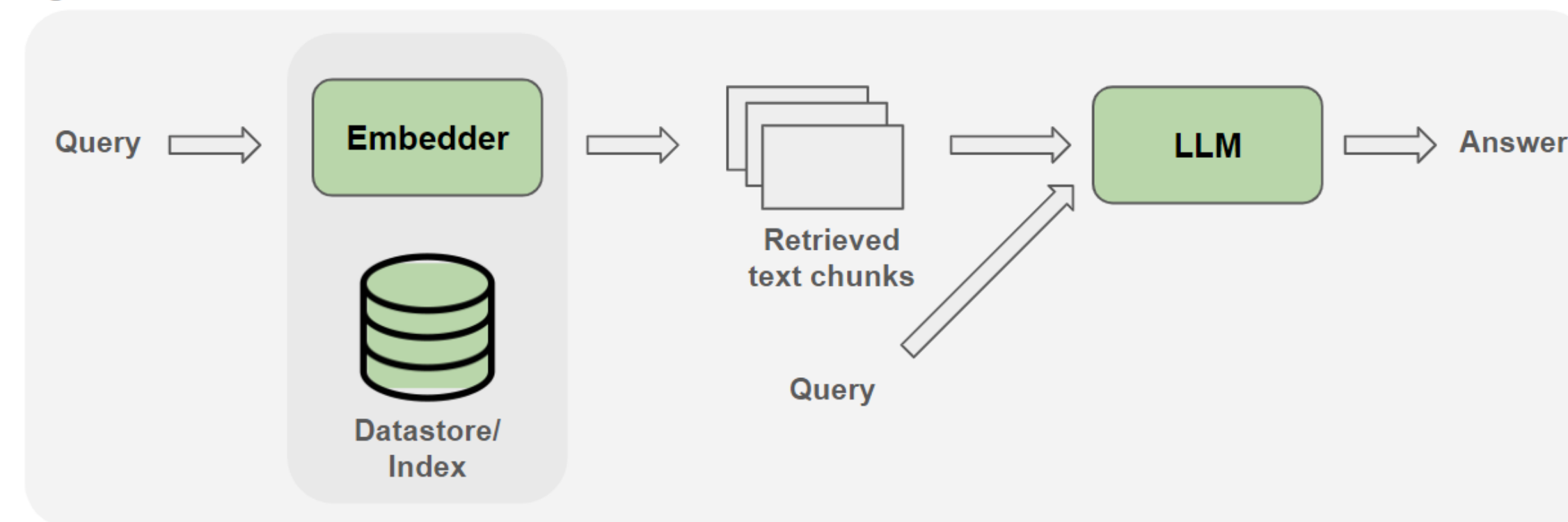
# Idea

- Give an access to a **database**
  - Given some user query, retrieve relevant document from the DB
    - Uses some embedding model
  - Retrieved document is given to LLM as a context

Indexing



Generating





# Idea

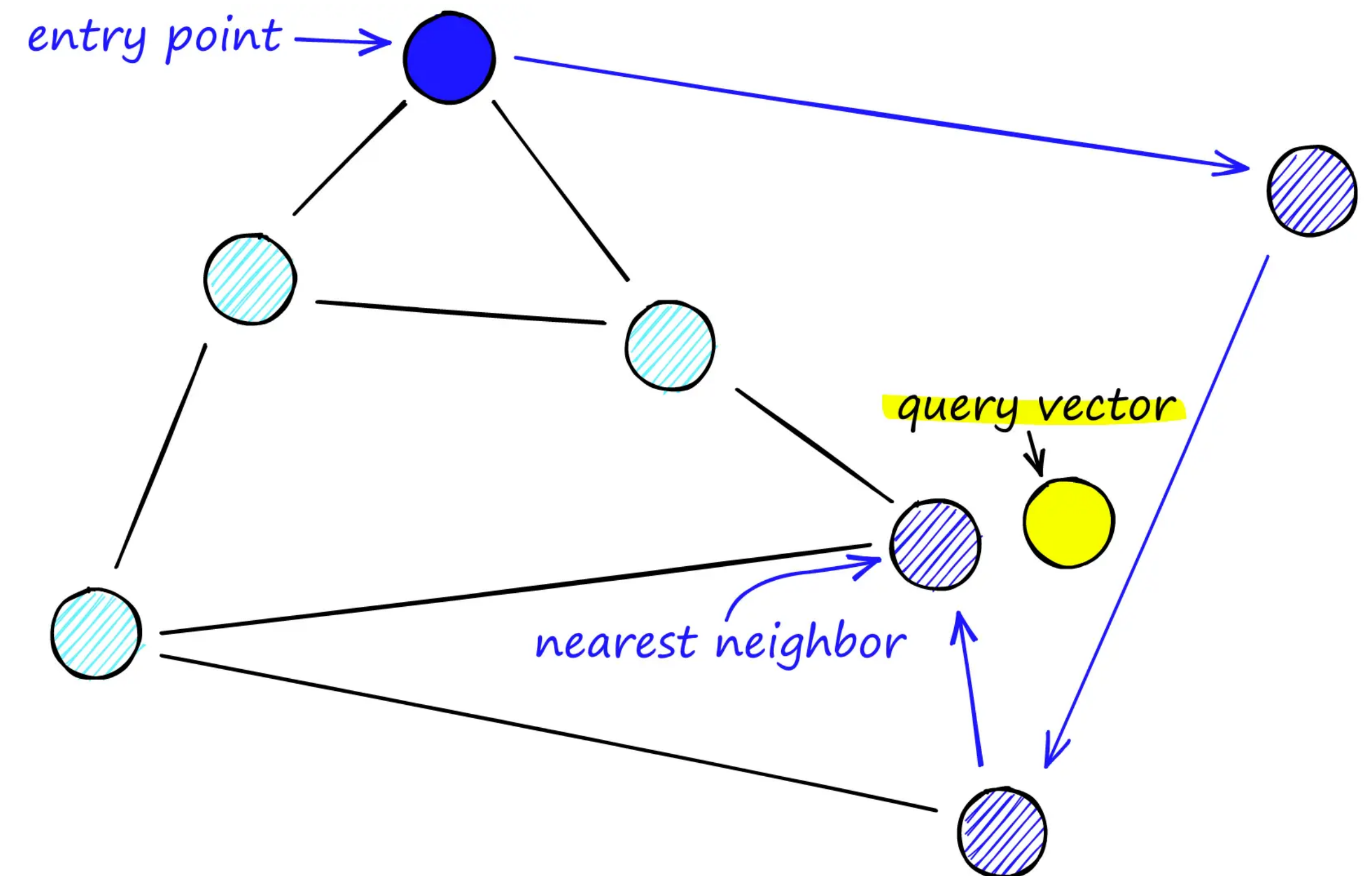
- Top-k relevant documents are discovered by **similarity matching**
  - Formally, given a set of documents  $\mathcal{V}$  and the query  $q$ , we do:

$$\arg \max_{v \in \mathcal{V}} \text{sim}(q, v)$$

- As the DB is too large, we organize  $\mathcal{V}$  using some data structure
  - e.g., hierarchical navigable small world

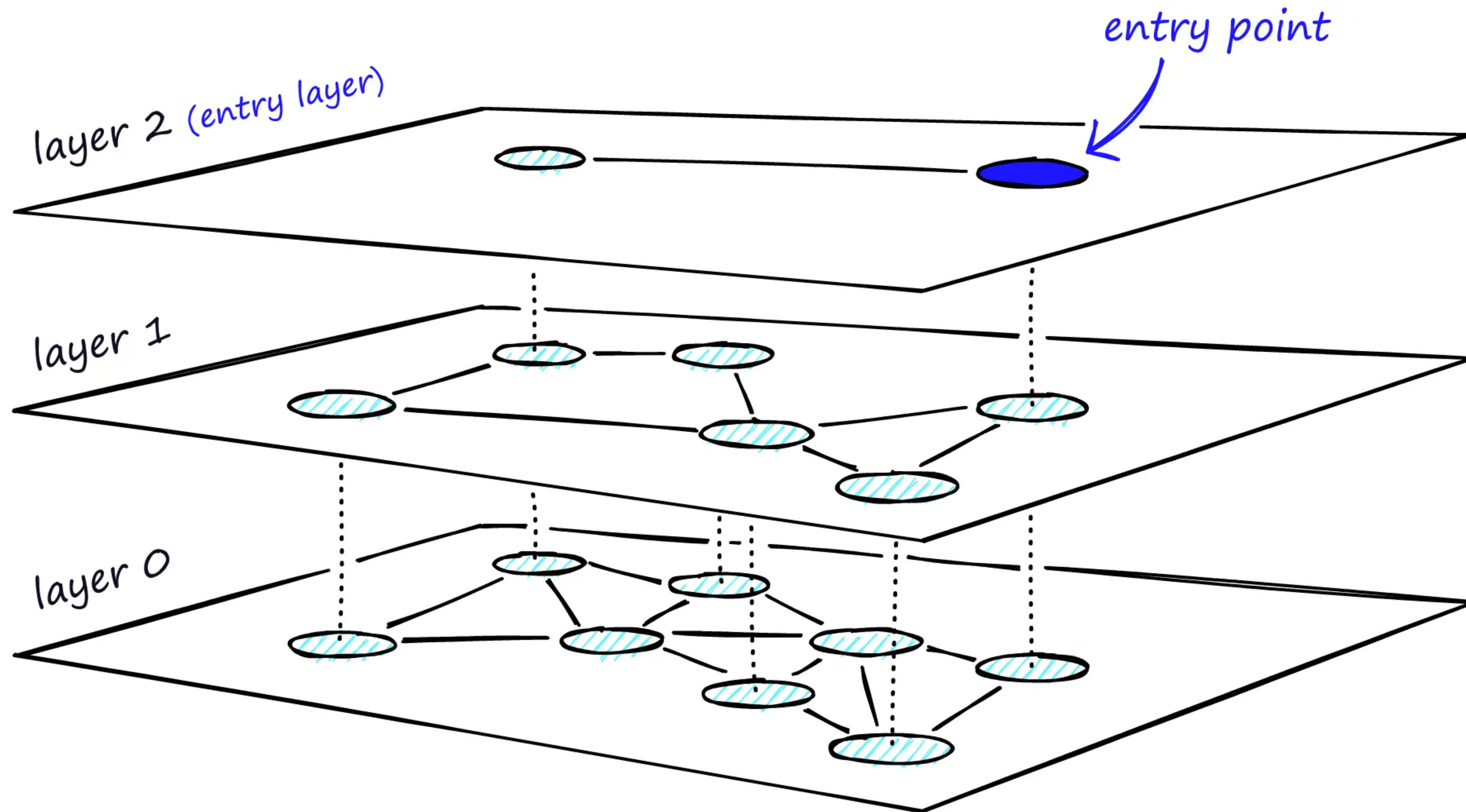
# Navigable small world

- We construct a loose graph of vectors
  - Each document is connected to nearby documents
- Enters some point
- Explores all neighbor documents
  - Evaluate similarity with query
- Jump to the most similar neighbor
- Repeat
- Can skip evaluating many documents



# Hierarchical navigable small world

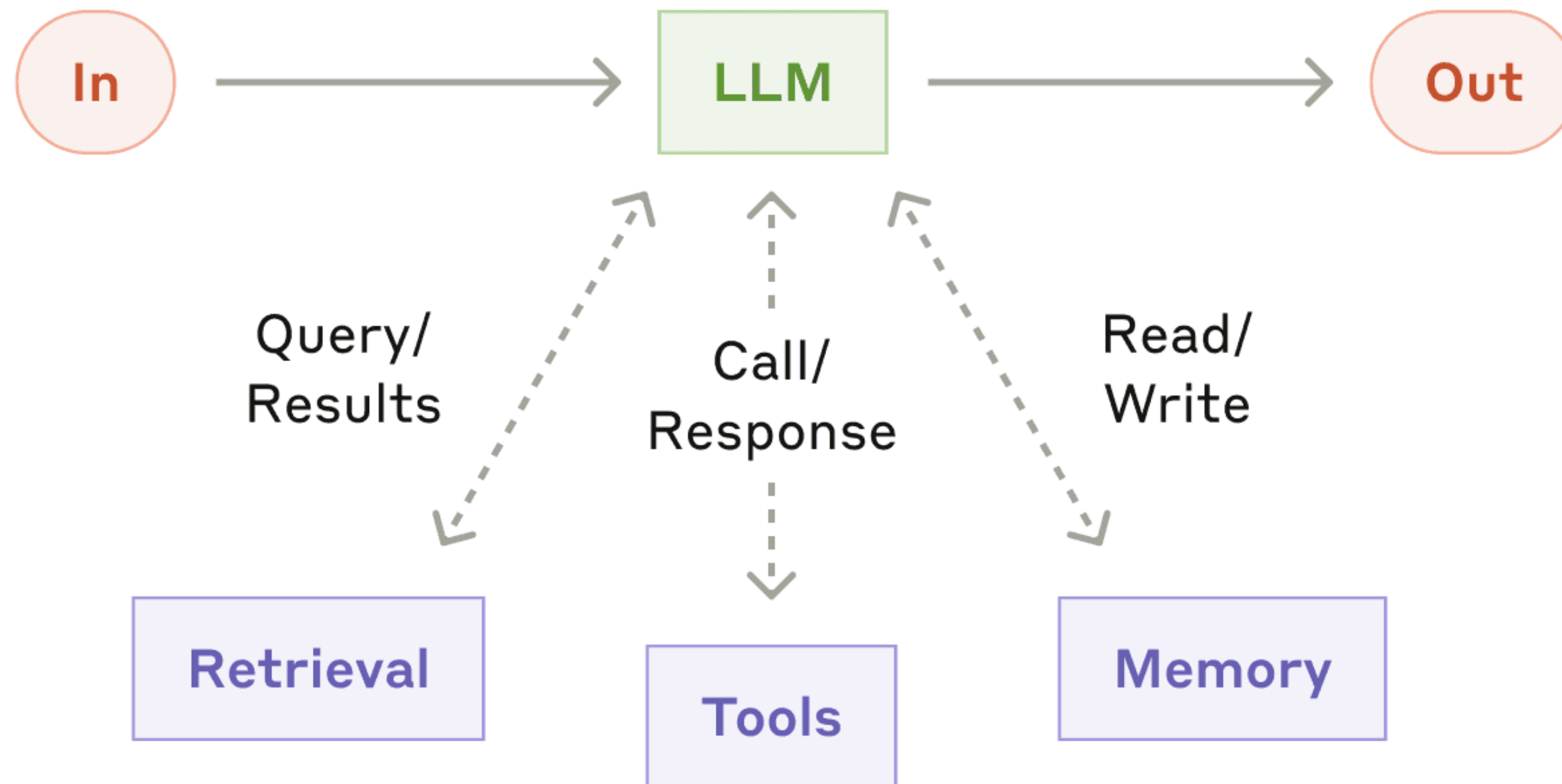
- We can do this with a hierarchy of graphs
  - Enables us to skip much more documents





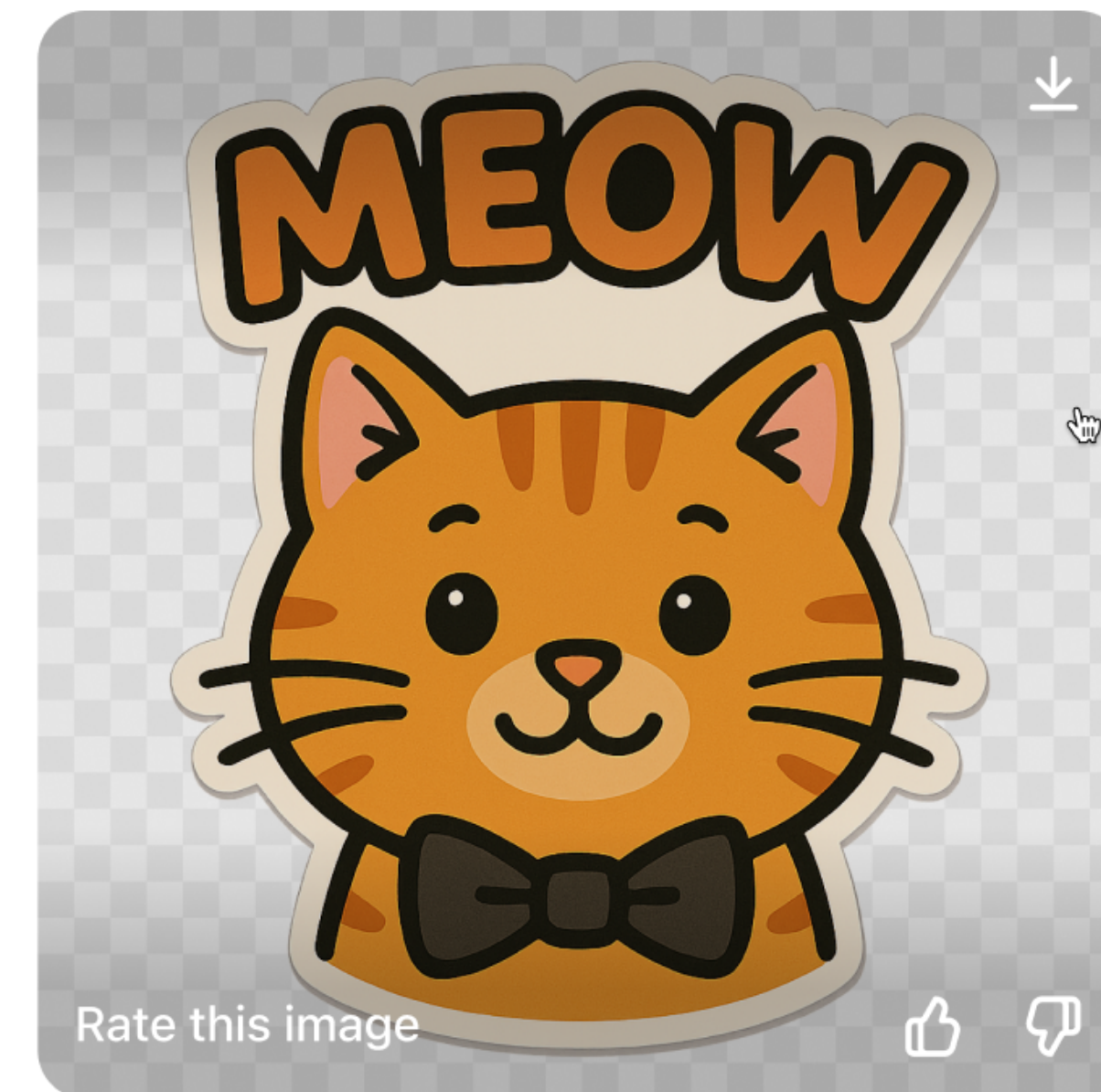
# Tool-Augmented Generation

- This paradigm is extended to a use of general tools
  - e.g., use diffusion models to generate answers
  - e.g., write codes for numerical evaluation



Create a sticker-style image of an orange cat wearing a bowtie with the word "meow" on top

Image created

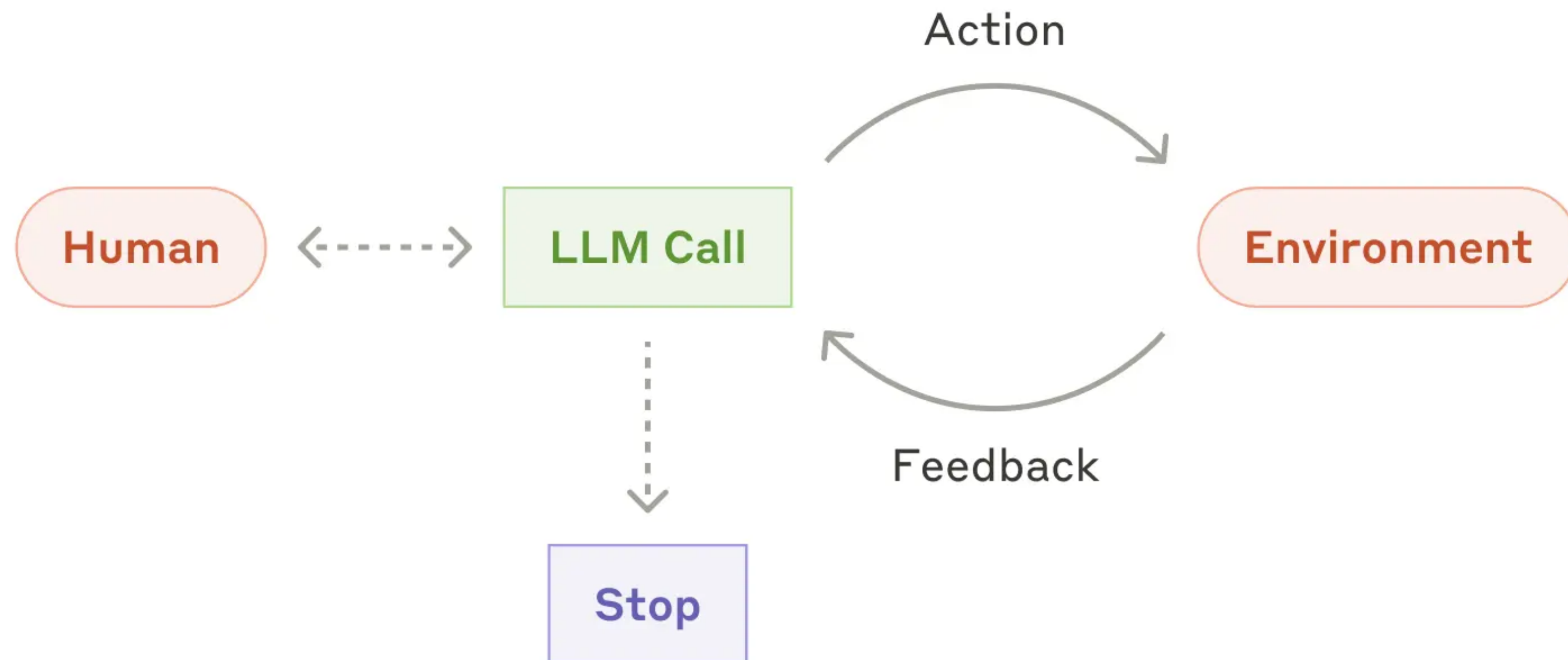


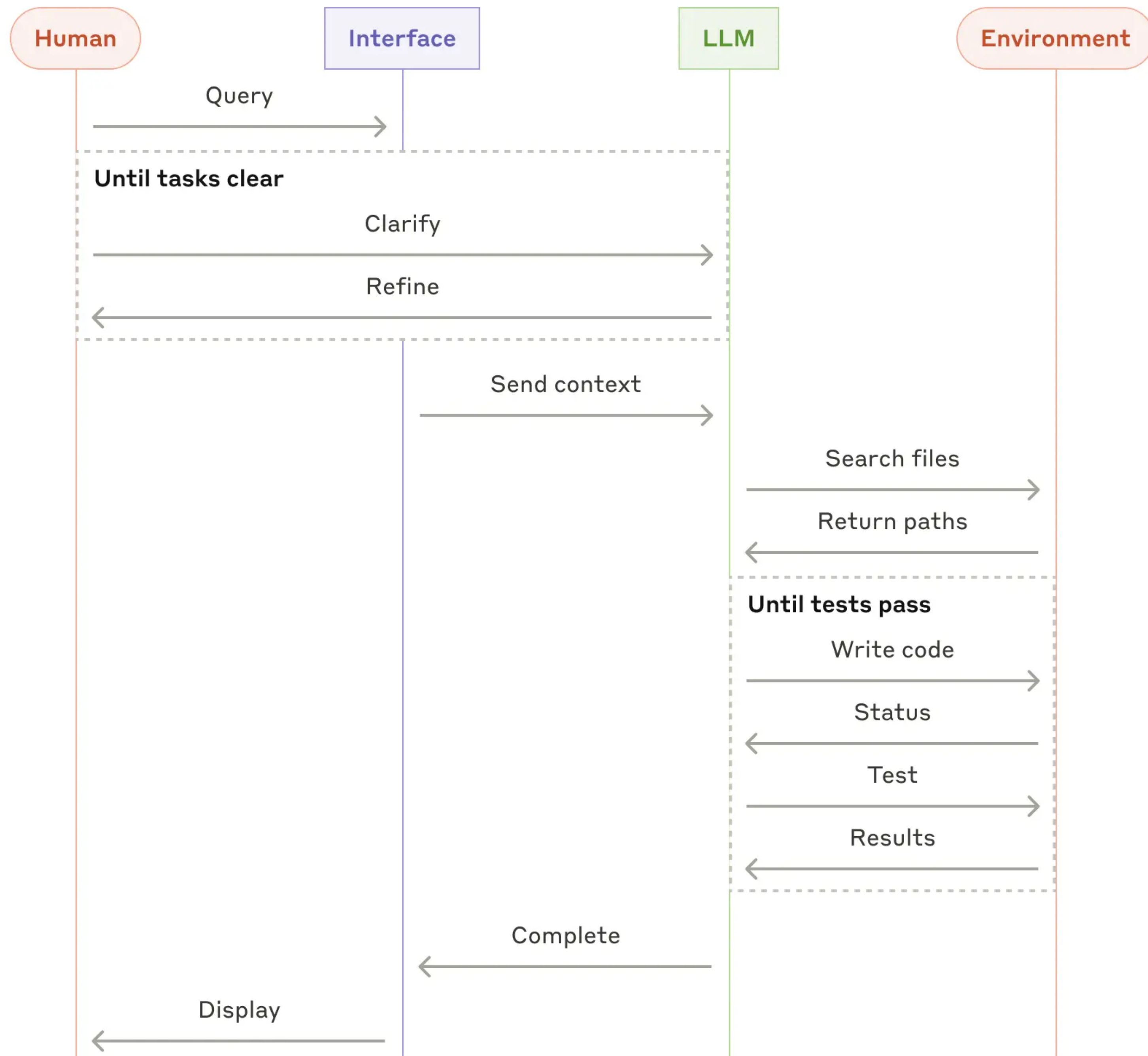
Here's your sticker-style illustration of an orange cat wearing a bowtie with the word "MEOW" on top!  
Let me know if you want any variations—different colors, accessories, or text.



# Agentic frameworks

- More recently, LLMs are being used to **interact with environments**
  - e.g., googling for some information, continuously changing the search keywords if the desired result do not come up
  - <https://openai.com/index/introducing-chatgpt-agent/>





# Next class

- Reinforcement learning basics

**</lecture 22>**