Building LLM-Powered Applications

Taming stochastic models with system design

MIT 15.S60, IAP 2026

Natasha Patnaik

Main Idea

So we have access to high-quality, pre-trained LLMs. Now what?

- Use state-of-the-art foundational models to build specialized applications.
- Incorporating a pre-trained model into a trustworthy system requires thoughtful design.
- DO NOT delegate all of the work to the LLM backbone: reliability and usability comes from surrounding framework.
- LLM is a **probabilistic black-box** that generates candidate answers, not verified ones.
- By placing the LLM **inside a control system**, we can validate, constrain, and route those generated outputs into dependable, task-oriented behavior.

Modularity, Extensibility, Re-usability

Working programmatically with LLMs requires:

- **Modularity:** Organize the system into well-defined components so behavior is easier to understand, test, and debug.
- **Extensibility:** Encapsulate shared logic in components that can be modified or expanded without rewriting the entire system.
- **Re-usability:** Keep core logic stable across runs; configuration and data should change, not the code itself.

This is a successful session if...

- ... we build **<u>strong foundations</u>** for the core concepts behind designing systems and applications that make use of transformers/ LLMs.
- ... we gain some familiarity with the **<u>current ecosystem</u>** of available resources, tools, and workflows.
- ... develop practical intuition through **<u>hands-on experience</u>** & coding.
- ... learn tips, tricks and techniques that can be useful for a variety of settings!

Topics

Cloud-Hosted vs Local LLMs

Basics of Cloud-Hosted Models

What they are

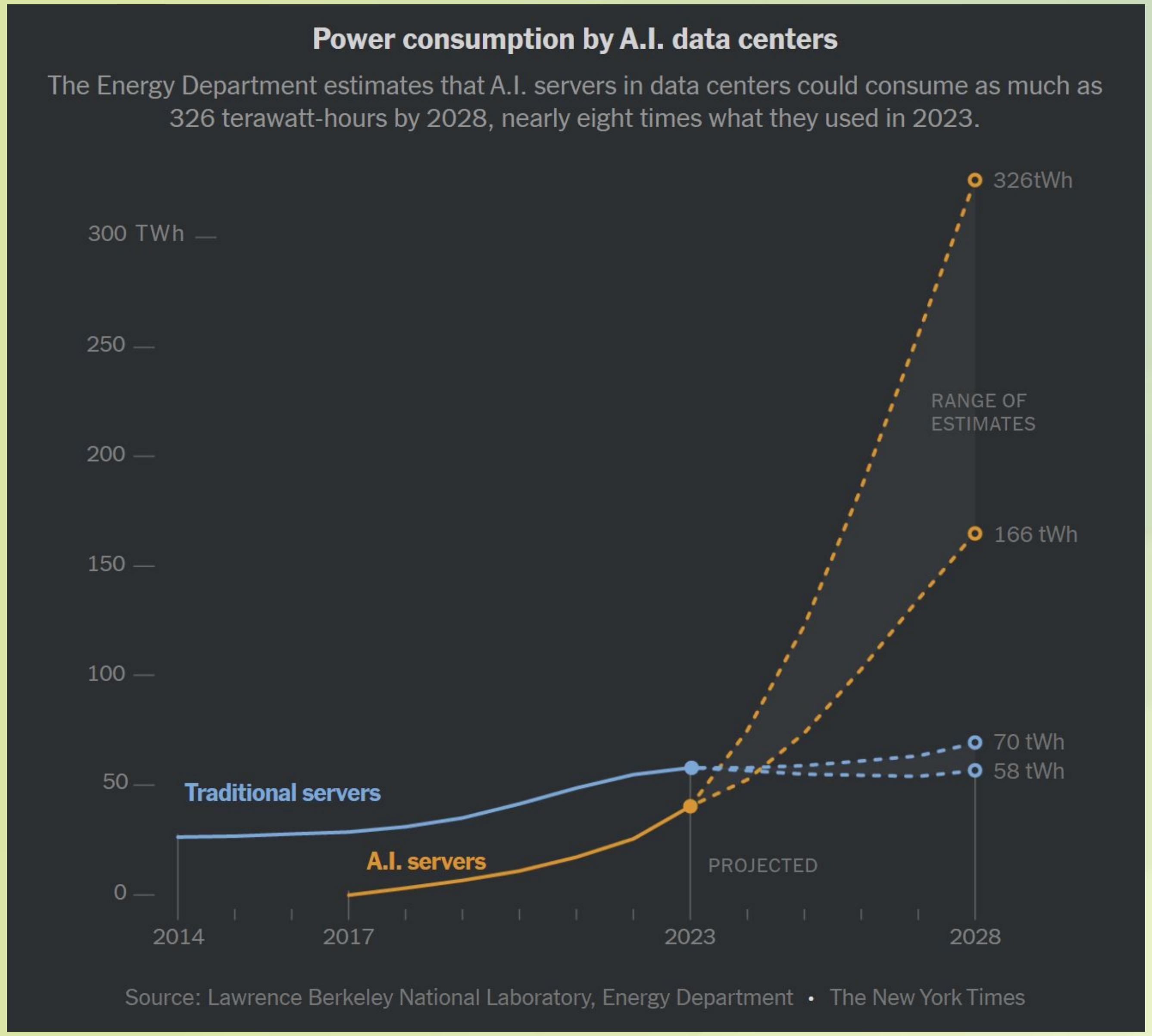
- Large foundation models served behind an HTTP API (e.g., OpenAI GPT-5, GPT-4o; Ollama Cloud)
- HTTP = “HyperText Transfer Protocol”, standard way for client and server to communicate over the internet
- Models run on provider-managed GPU clusters in their data centers
- You send prompts, get tokens back - no control over hardware or training stack

Key properties

- Access to state-of-the-art models (reasoning, multimodal, long context)
- Elastic scaling: you get thousands of GPUs when needed
- Managed reliability: uptime, updates, performance tuning handled for you

Trade-offs

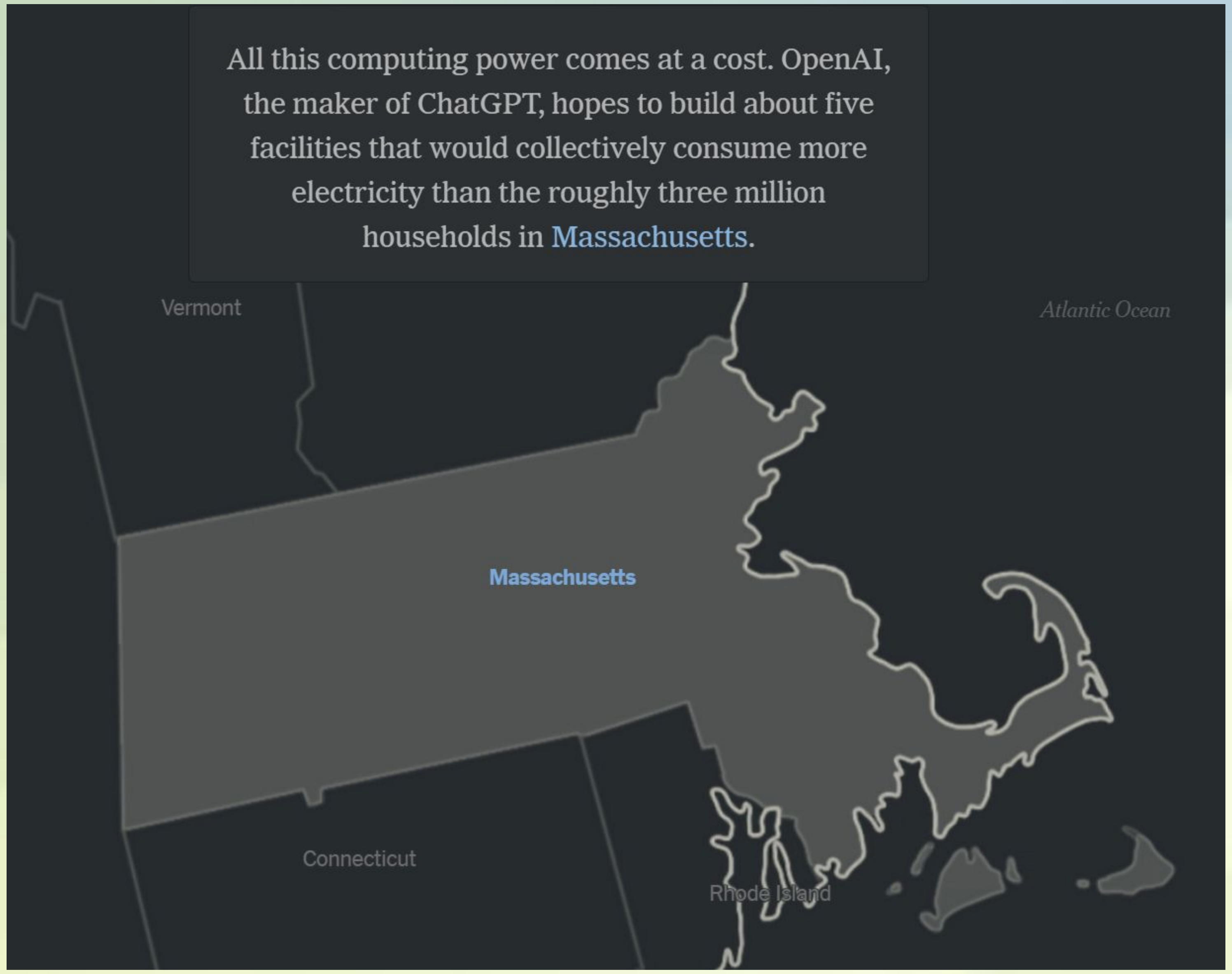
- Pay per token - cost scales with usage
- Data leaves your machine (privacy, security compliance and regulations)
- Limited ability to inspect or modify model internals



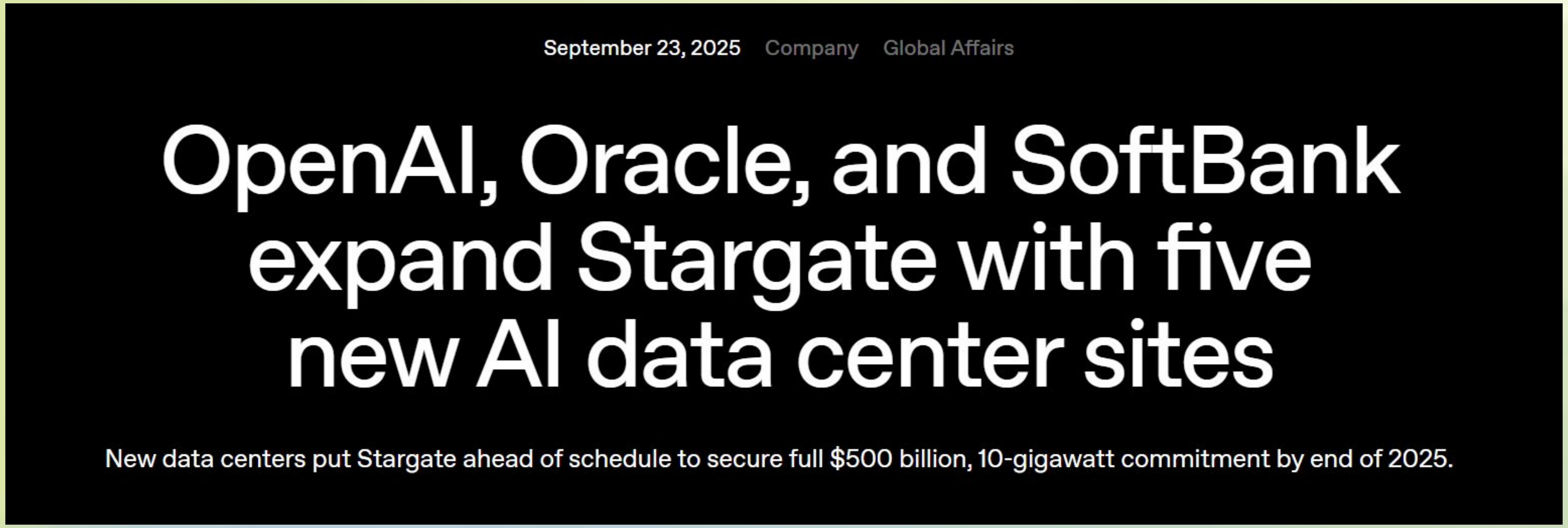
Power consumption by A.I. data centers

The Energy Department estimates that A.I. servers in data centers could consume as much as 326 terawatt-hours by 2028, nearly eight times what they used in 2023.

Source: Lawrence Berkeley National Laboratory, Energy Department • The New York Times



All this computing power comes at a cost. OpenAI, the maker of ChatGPT, hopes to build about five facilities that would collectively consume more electricity than the roughly three million households in Massachusetts.

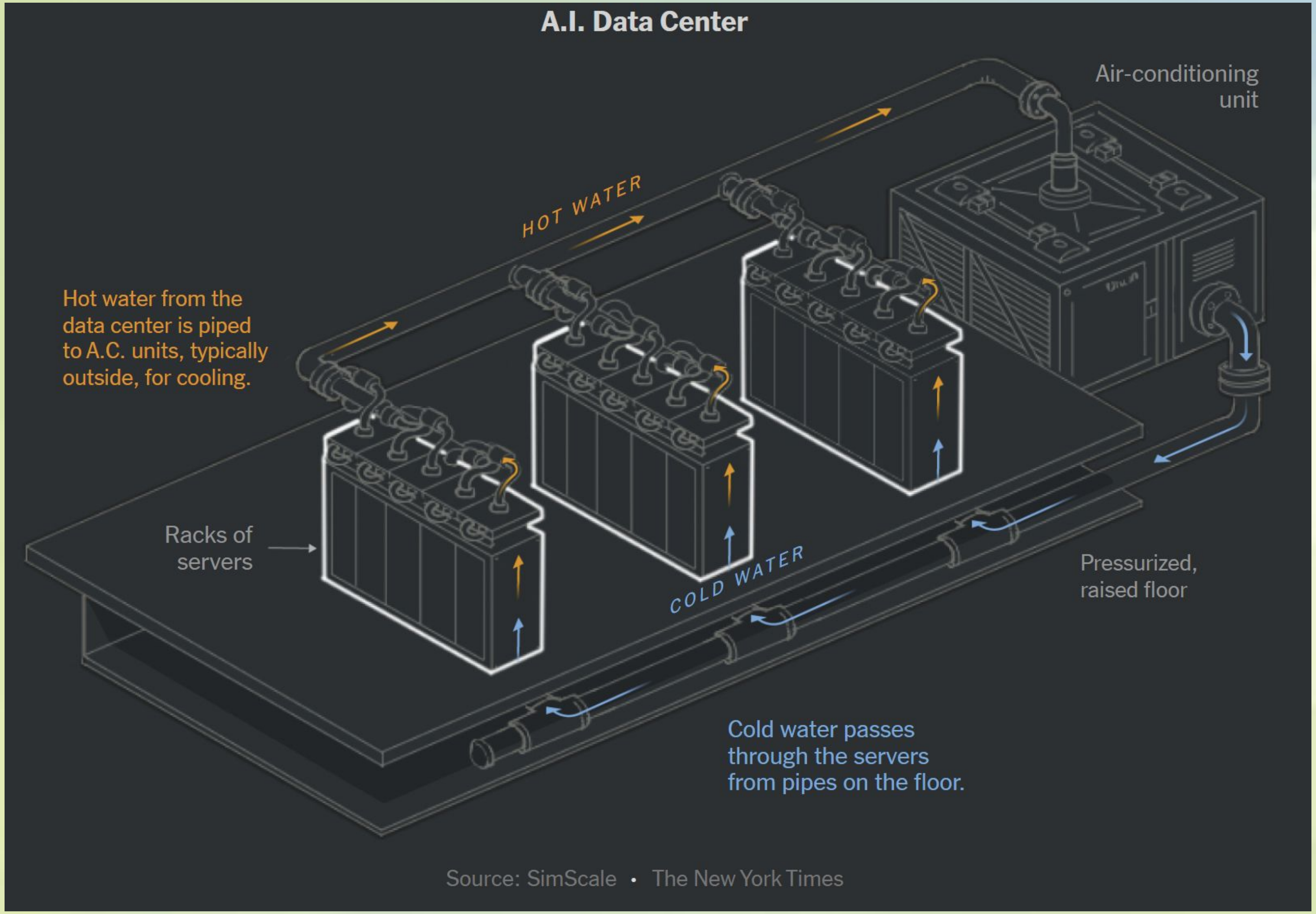


September 23, 2025 Company Global Affairs

OpenAI, Oracle, and SoftBank expand Stargate with five new AI data center sites

New data centers put Stargate ahead of schedule to secure full $500 billion, 10-gigawatt commitment by end of 2025.

- How A.I. Is Changing the Way the World Builds Computers - The New York Times
- OpenAI, Oracle, and SoftBank expand Stargate with five new AI data center sites | OpenAI


A.I. Data Center
Air-conditioning unit
HOT WATER
Hot water from the data center is piped to A.C. units, typically outside, for cooling.
Racks of servers
COLD WATER
Pressurized, raised floor
Cold water passes through the servers from pipes on the floor.
Source: SimScale • The New York Times

[How A.I. Is Changing the Way the World Builds Computers - The New York Times]

Basics of Local Models

What they are

- Open-weight models (LLaMA, Mistral, Qwen, etc.) running on your own GPU or CPU
- Inference happens on your own laptop or HPC cluster

Key properties

- Full control over model weights, quantization (reducing model size by representing weights with fewer bits), fine-tuning, etc.
- No per-token costs or artificially imposed usage constraints
- Security and Privacy: data never leaves your environment!

Trade-offs

- Inherently limited by your own available computational resources (GPU memory and compute)
- Slower than optimized cloud models or commercially available options
- More low-level control and design choices that you have to proactively consider

Hugging Face: Supervised Fine-Tuning

Supervised Fine-Tuning (Big Picture)

Alignment: Rather than only predicting the most statistically likely next token, we want the model to generate outputs that better reflect human preferences and values.

Fine-tuning permanently alters the model's weights.

- Use a general-purpose pre-trained model's existing weights as a starting point.
- Compile a high-quality dataset of labelled examples (input-output pairs) for our task.
- Continue to train further on the task-specific training objective.
- This is possible due to transfer learning.

Supervised Fine-Tuning

Our goal is to adapt a pre-trained foundation model to a specific task using labeled data.

Let f_θ be a language model that maps an input x to a predictive distribution $f_\theta(x)$. Let $\mathcal{D}_{\text{pre-train}}$ be a large corpus of unlabeled text sequences $\{x_j\}_{j=1}^{M}$. The pretrained parameters θ_0 are obtained by solving

$$\theta_0 = \arg\min_{\theta} \mathbb{E}_{x \sim \mathcal{D}_{\text{pre-train}}} \left[\mathcal{L}_{\text{pre-train}}(f_\theta, x) \right],$$

where $\mathcal{L}_{\text{pre-train}}$ is a self-supervised language modeling objective (e.g., masked token prediction or next-token prediction) applied to the sequence x. In this stage, the model learns general-purpose representations of language.

Now, suppose we have a smaller, task-specific labeled dataset $\mathcal{D}_{\text{task}} = \{(x_i, y_i)\}_{i=1}^{N}$ with $N \ll M$. Supervised Fine-Tuning (SFT) initializes the model with θ_0 and optimizes

$$\theta^* = \arg\min_{\theta} \mathbb{E}_{(x,y) \sim \mathcal{D}_{\text{task}}} \left[\mathcal{L}_{\text{task}}(f_\theta(x), y) \right].$$

Rather than training from scratch, the model transfers the semantic knowledge encoded in θ_0 and adapts it to produce task-specific outputs y for inputs x. In practice, $\mathcal{L}_{\text{pre-train}}$ is a self-supervised language modeling loss, while $\mathcal{L}_{\text{task}}$ is chosen to match the downstream task (e.g., classification loss).

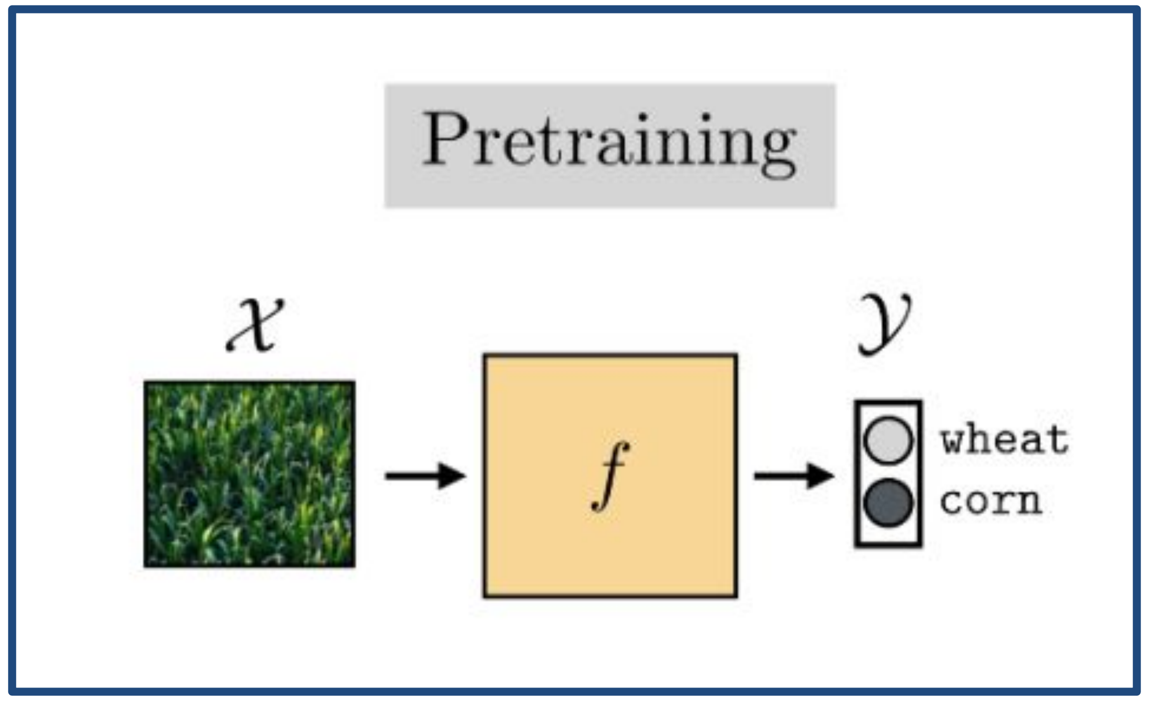
Transfer Learning

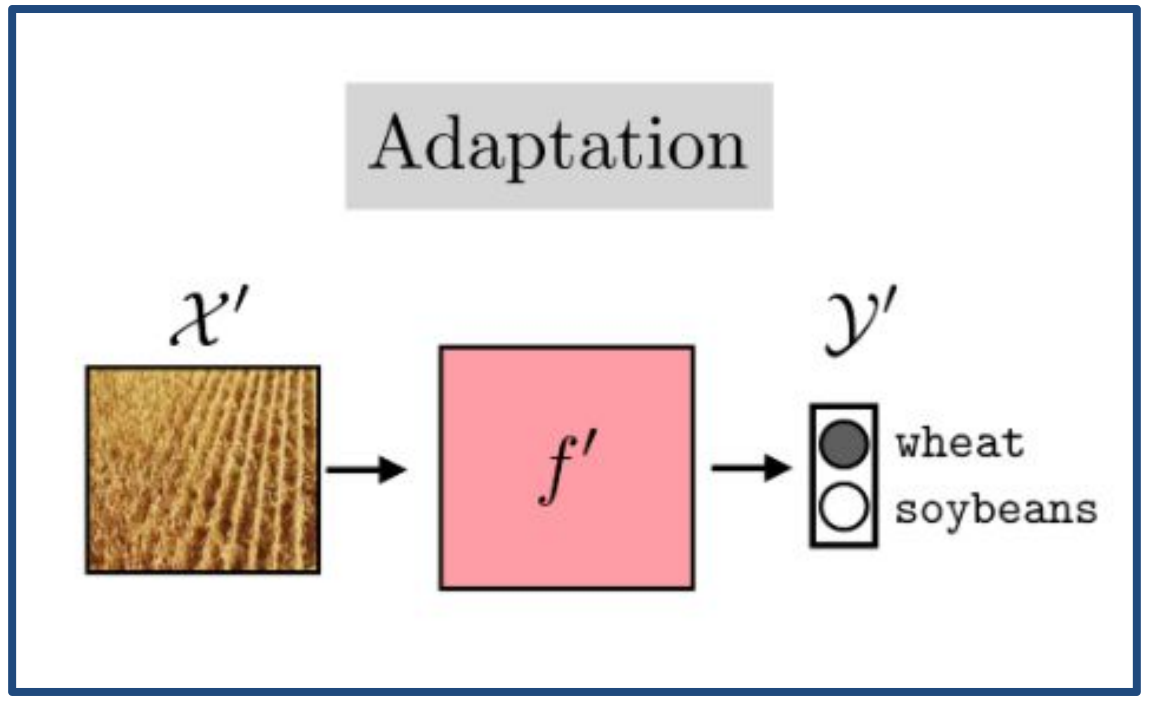
There are three stages: (1) pretrain $f : \mathcal{X} \rightarrow \mathcal{Y}$, (2) initialize $f' = f$, and (3) finetune $f' : \mathcal{X}' \rightarrow \mathcal{Y}'$. The full algorithm is written in Algorithm 37.1, with f and f' indicated as just a continually learning f_θ with different iterates of θ as learning progresses.

Algorithm 37.1: Finetuning. Using gradient descent to train one model and then finetuning to produce a second model.

```
1 **Input:** initial parameter vector $\theta^0$, data $\{x^{(i)}, y^{(i)}\}_{i=1}^N$, $\{x'^{(i)}, y'^{(i)}\}_{i=1}^M$, learning rates $\eta_1$ and $\eta_2$
2 **Output:** trained models $f_{\theta^N}$ and $f_{\theta^{N+M}}$
3 **Pretraining: for** $k = 1, \ldots, K_1$ **do**
4     $J = \mathbb{E}_{x,y}[\mathcal{L}(f_{\theta^{k-1}}(x), y)]$
5     $\theta^k \leftarrow \theta^{k-1} - \eta_1 \nabla_\theta J$
6 **Finetuning: for** $k = 1, \ldots, K_2$ **do**
7     $J = \mathbb{E}_{x',y'}[\mathcal{L}(f_{\theta^{k-1+N}}(x'), y')]$
8     $\theta^{k+N} \leftarrow \theta^{k-1+N} - \eta_2 \nabla_\theta J$
```

A. Torralba, P. Isola, and W.T. Freeman. *Foundations of Computer Vision*. Adaptive Computation and Machine Learning series. MIT Press, 2024. ISBN 9780262378666. URL https://mitpress.mit.edu/9780262048972/foundations-of-computer-vision/.





Self-supervised Learning Tasks

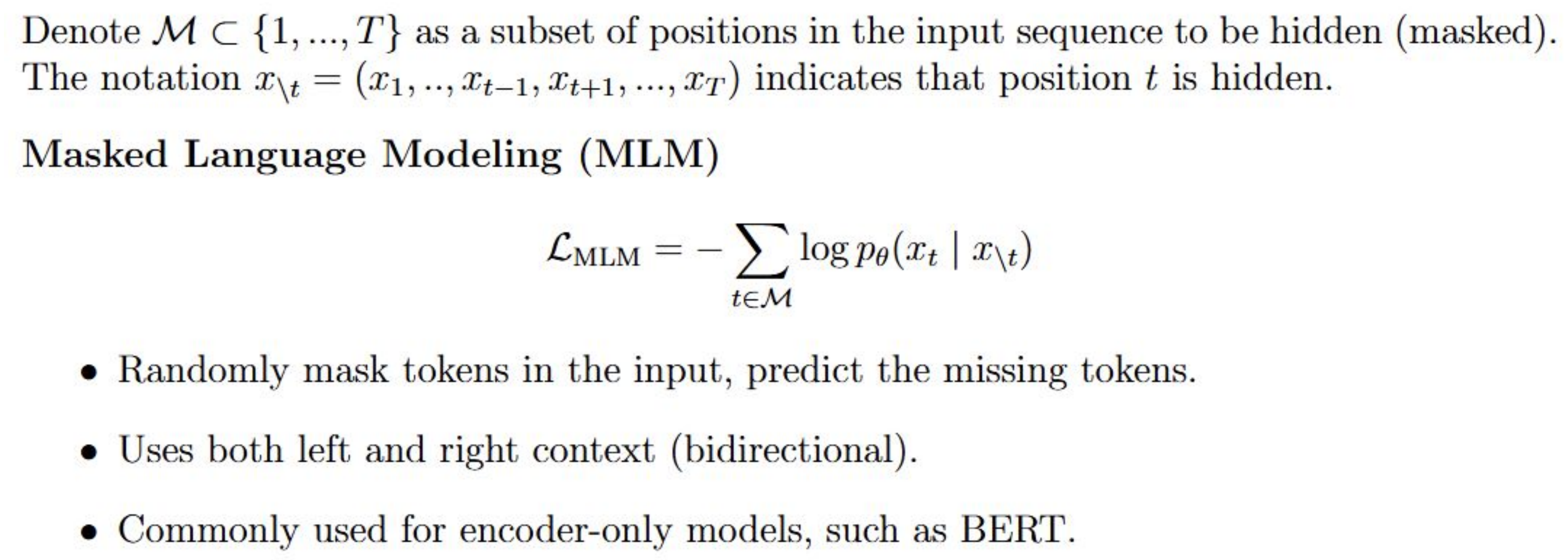
Let $x = (x_1, ..., x_T)$ be the original input sequence of tokens. The notation $x_{<t} = (x_1, ..., x_{t-1})$ denotes a truncated sequence upto (not including) step t.

Next-Token Prediction (Auto-regressive LM)

$$\mathcal{L}_{\text{NTP}} = -\sum_{t=1}^{T} \log p_\theta(x_t \mid x_{<t})$$

- Predict the next token given all the previous tokens.
- Uses only left context (causal, auto-regressive).
- Commonly used for decoder-only models, such as GPT and LLaMA.

Self-supervised Learning Tasks



Denote $\mathcal{M} \subset \{1, ..., T\}$ as a subset of positions in the input sequence to be hidden (masked). The notation $x_{\setminus t} = (x_1, .., x_{t-1}, x_{t+1}, ..., x_T)$ indicates that position t is hidden.

Masked Language Modeling (MLM)

$$\mathcal{L}_{\text{MLM}} = -\sum_{t \in \mathcal{M}} \log p_\theta(x_t \mid x_{\setminus t})$$

- Randomly mask tokens in the input, predict the missing tokens.
- Uses both left and right context (bidirectional).
- Commonly used for encoder-only models, such as BERT.

Self-supervised Learning Tasks

Next Sentence Prediction (NSP)

$$\mathcal{L}_{\text{NSP}} = -\log p_\theta(y \mid s_1, s_2)$$

- Predict whether sentence s_2 follows sentence s_1.
- Encourages sentence-level and discourse understanding.
- Used in earlier models for BERT-style pre-training.

Binary Cross-Entropy with Logits for Multi-Label Classification.

In practice, models output *logits* rather than probabilities. Let $z_k = f_\theta(x)_k \in \mathbb{R}$ be the raw score for label k. The sigmoid function converts logits into probabilities:

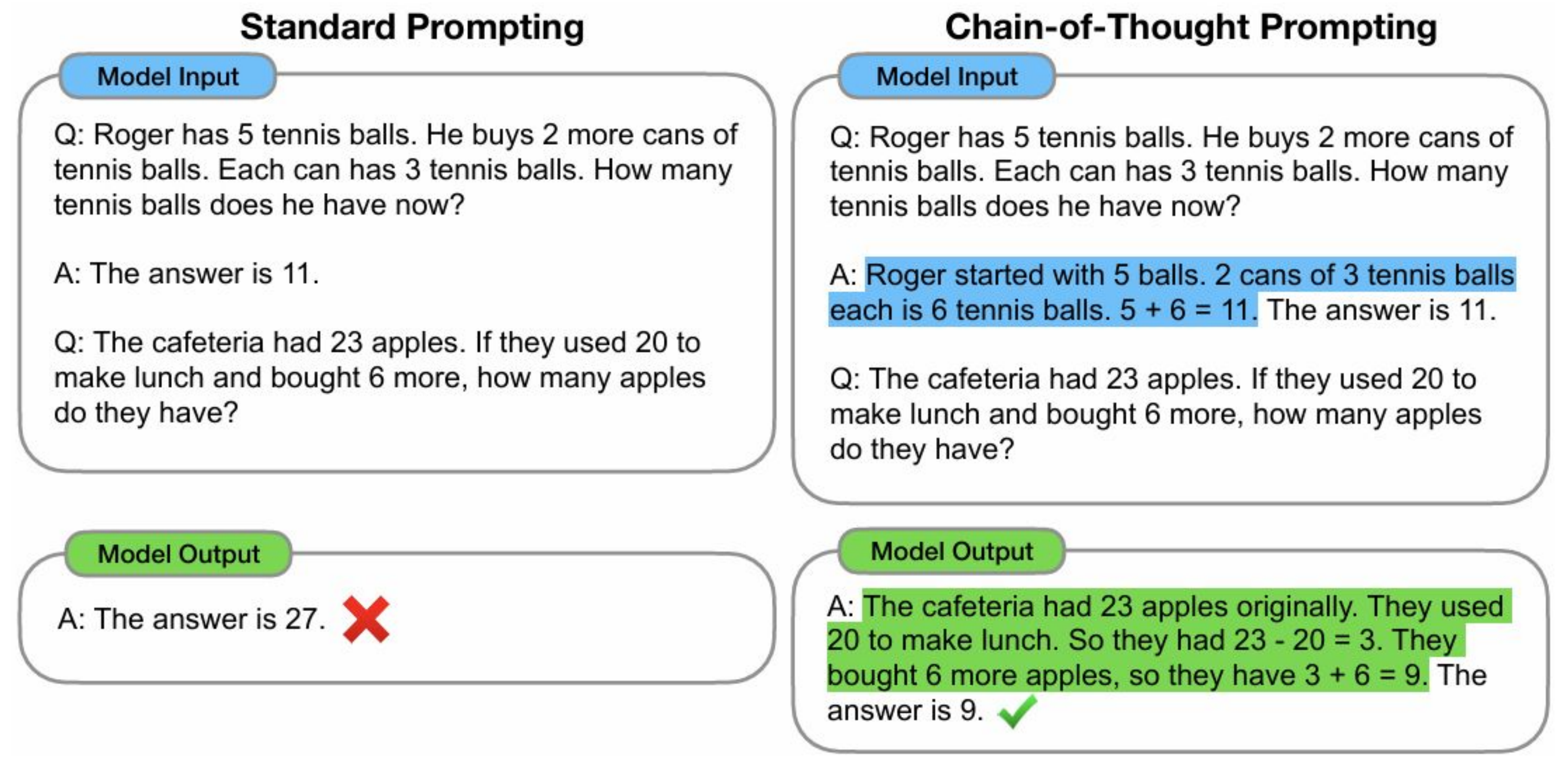
$$\sigma(z_k) = \frac{1}{1 + e^{-z_k}}.$$

Let $y_k \in \{0, 1\}$ be the true label for class k. The binary cross-entropy loss for one example:

$$\mathcal{L}_{\text{BCE}}(x, \mathbf{y}) = -\sum_{k=1}^{K} \left[y_k \log \sigma(z_k) + (1 - y_k) \log \left(1 - \sigma(z_k)\right) \right].$$

For numerical stability, PyTorch computes this loss directly from logits z_k without explicitly forming $\sigma(z_k)$.

Orchestration: Tool Use and Agents



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?

A: The answer is 27. ✗

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?

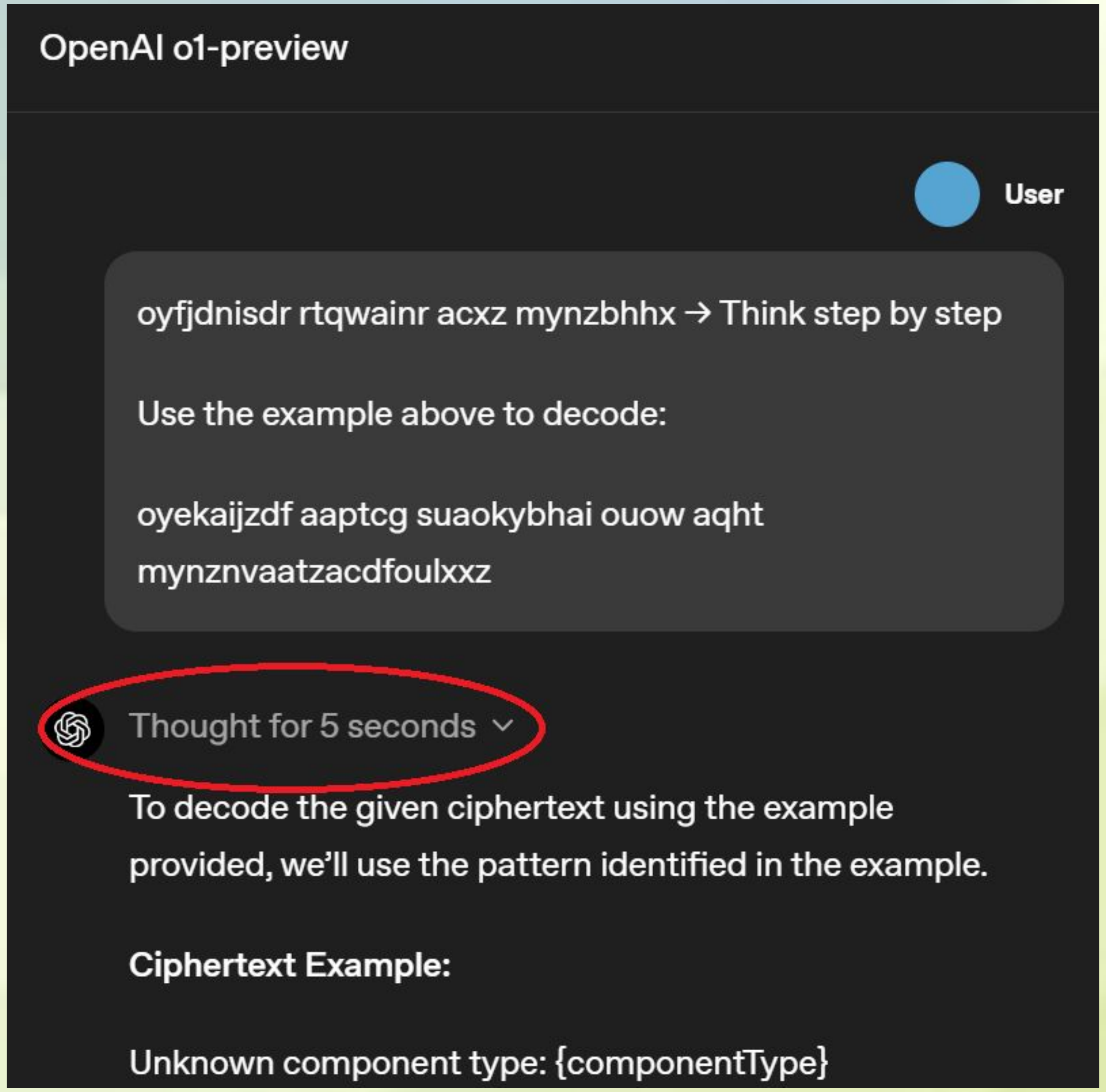
A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

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

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. ✓

Figure 1: Chain-of-thought prompting enables large language models to tackle complex arithmetic, commonsense, and symbolic reasoning tasks. Chain-of-thought reasoning processes are highlighted.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.



oyfjdnisdr rtqwainr acxz mynzbhhx → Think step by step

Use the example above to decode:

oyekaijzdf aaptcg suaokybhai ouow aqht mynznvaatzacdfoulxxz

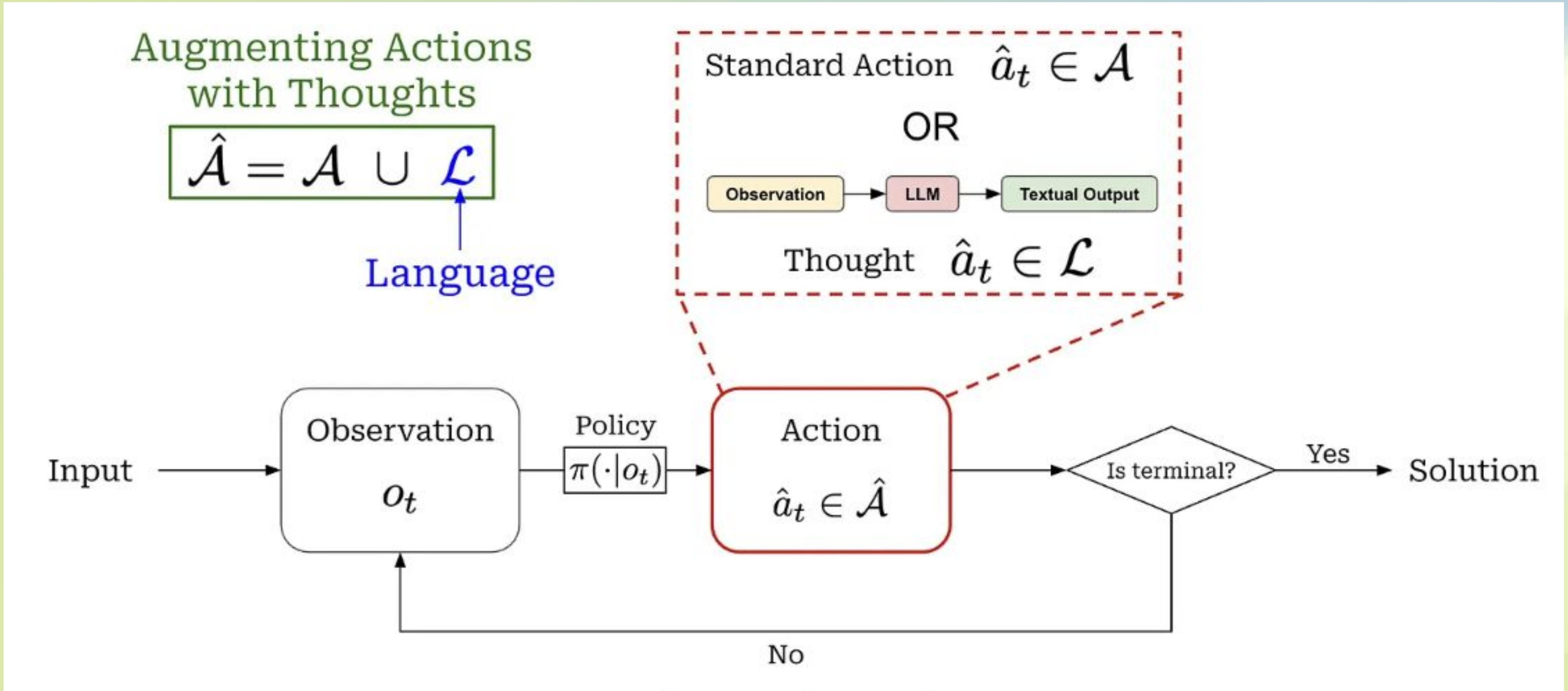
To decode the given ciphertext using the example provided, we'll use the pattern identified in the example.

Ciphertext Example:

Unknown component type: {componentType}

Learning to reason with LLMs

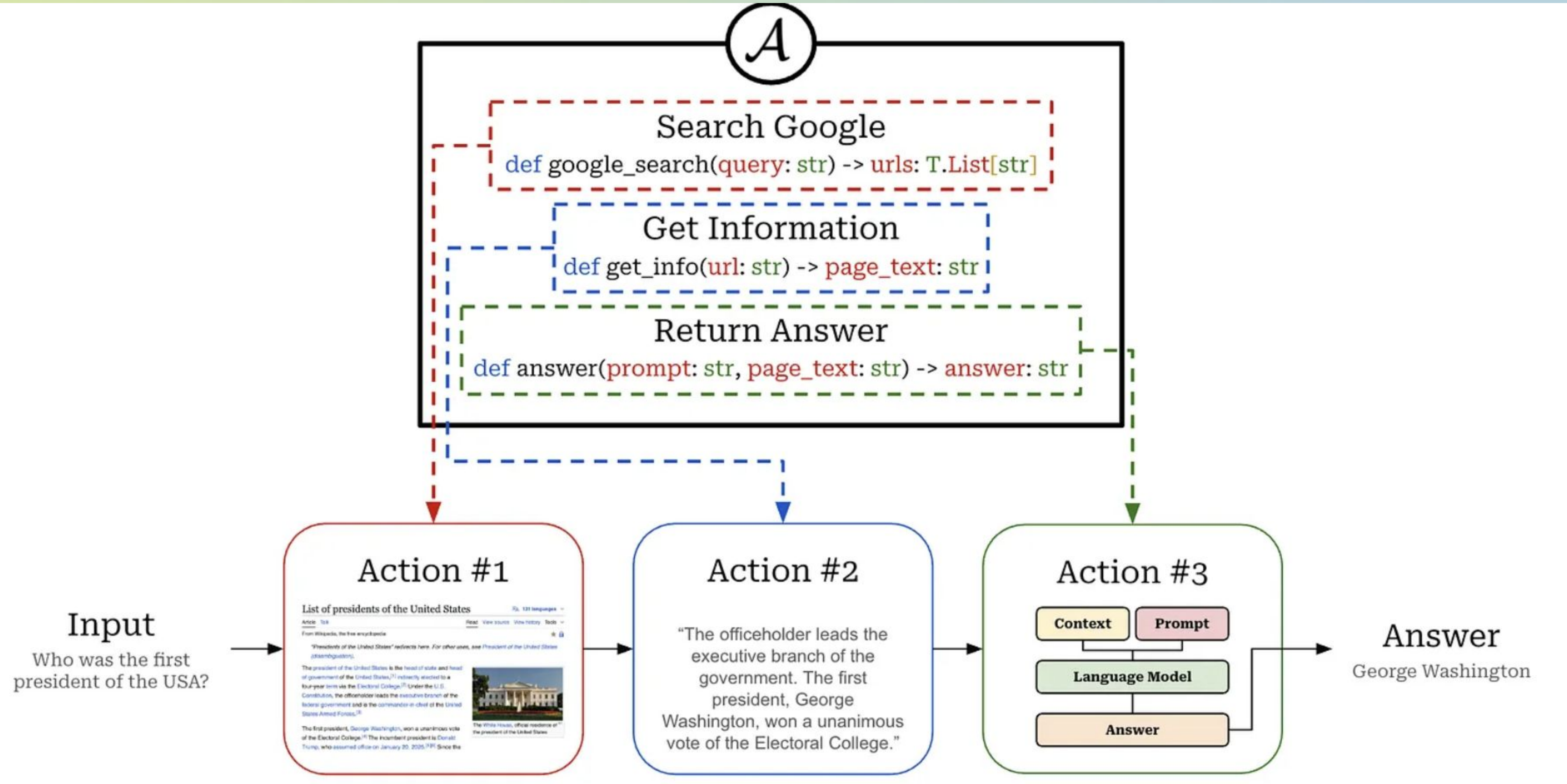
Learning to reason with LLMs | OpenAI


Augmenting Actions with Thoughts
$\hat{\mathcal{A}} = \mathcal{A} \cup \mathcal{L}$
Language
Standard Action $\hat{a}_t \in \mathcal{A}$
OR
Observation
LLM
Textual Output
Thought $\hat{a}_t \in \mathcal{L}$
Input
Observation
o_t
Policy
$\pi(\cdot|o_t)$
Action
$\hat{a}_t \in \hat{\mathcal{A}}$
Is terminal?
Yes
Solution
No

The ReAct framework

Cameron R. Wolfe. AI Agents from First Principles. Deep (Learning) Focus, June 9 2025. URL https://cameronrwolfe.substack.com/p/ai-agents.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing Reasoning and Acting in Language Models, 2023. URL https://arxiv.org/abs/2210.03629.



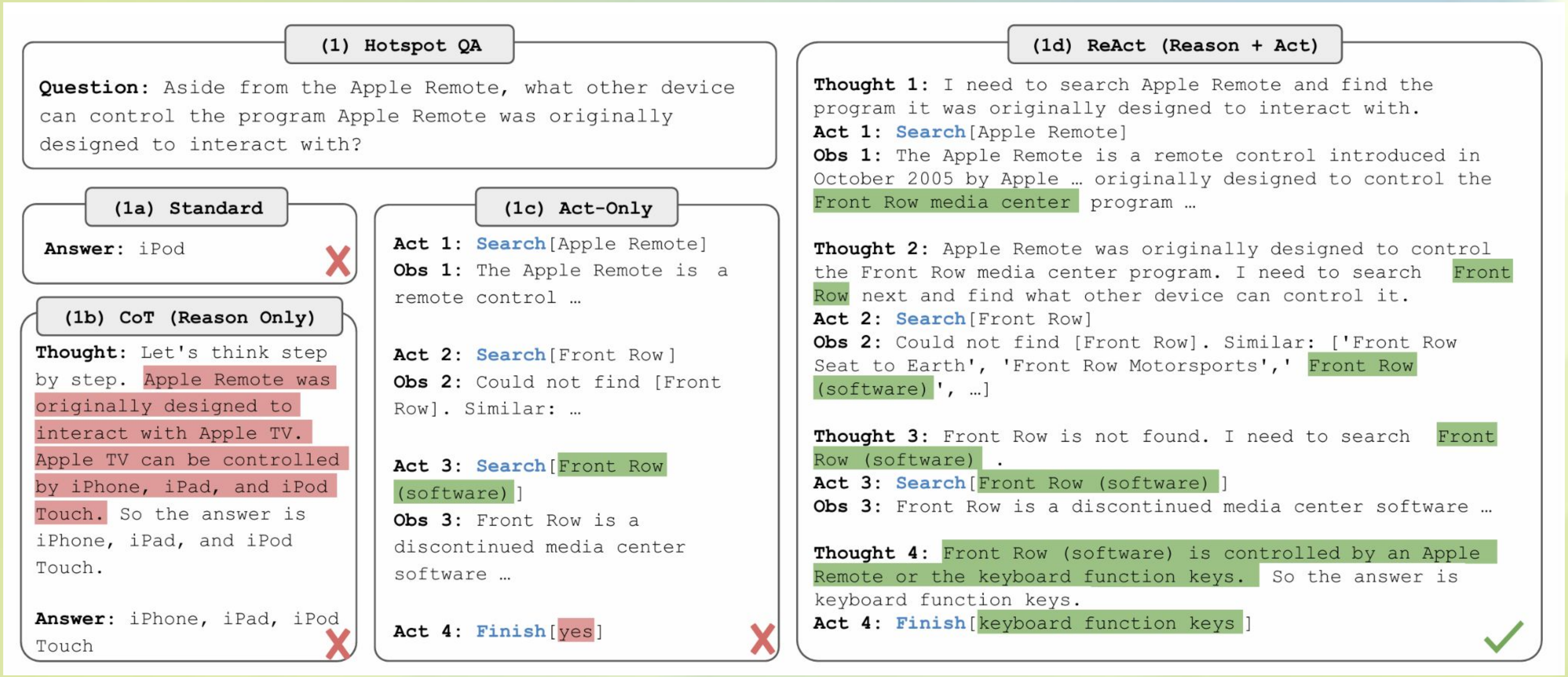
```
def google_search(query: str) -> urls: T.List[str]
def get_info(url: str) -> page_text: str
def answer(prompt: str, page_text: str) -> answer: str
```

"The officeholder leads the executive branch of the government. The first president, George Washington, won a unanimous vote of the Electoral College."

Example action space for an agent

Cameron R. Wolfe. AI Agents from First Principles. Deep (Learning) Focus, June 9 2025. URL https://cameronrwolfe.substack.com/p/ai-agents.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing Reasoning and Acting in Language Models, 2023. URL https://arxiv.org/abs/2210.03629.


(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)

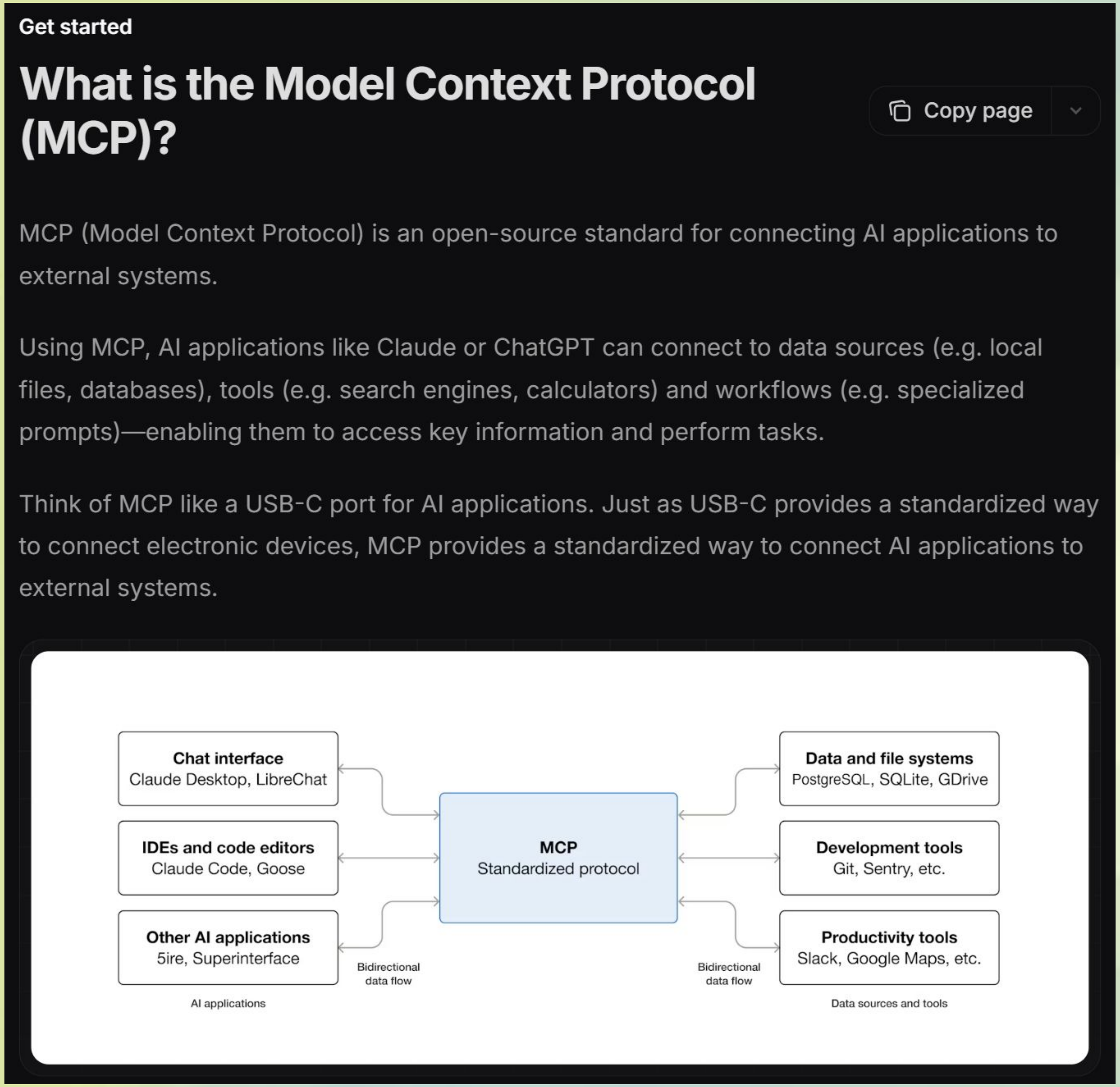
Thought 1: I need to search Apple Remote and find the program it was originally designed to interact with.
Act 1: Search[Apple Remote]
Obs 1: The Apple Remote is a remote control introduced in October 2005 by Apple … originally designed to control the Front Row media center program …

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] ✓

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing Reasoning and Acting in Language Models, 2023. URL https://arxiv.org/abs/2210.03629.



Get started

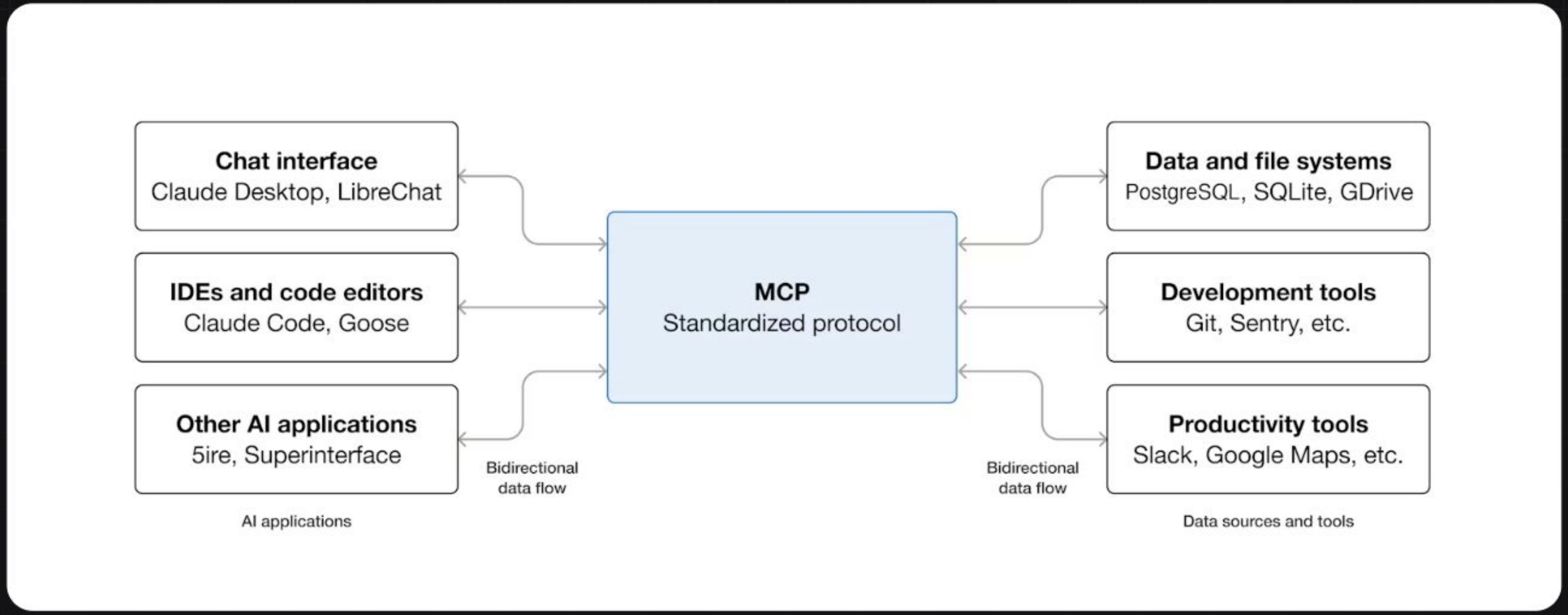
What is the Model Context Protocol (MCP)?

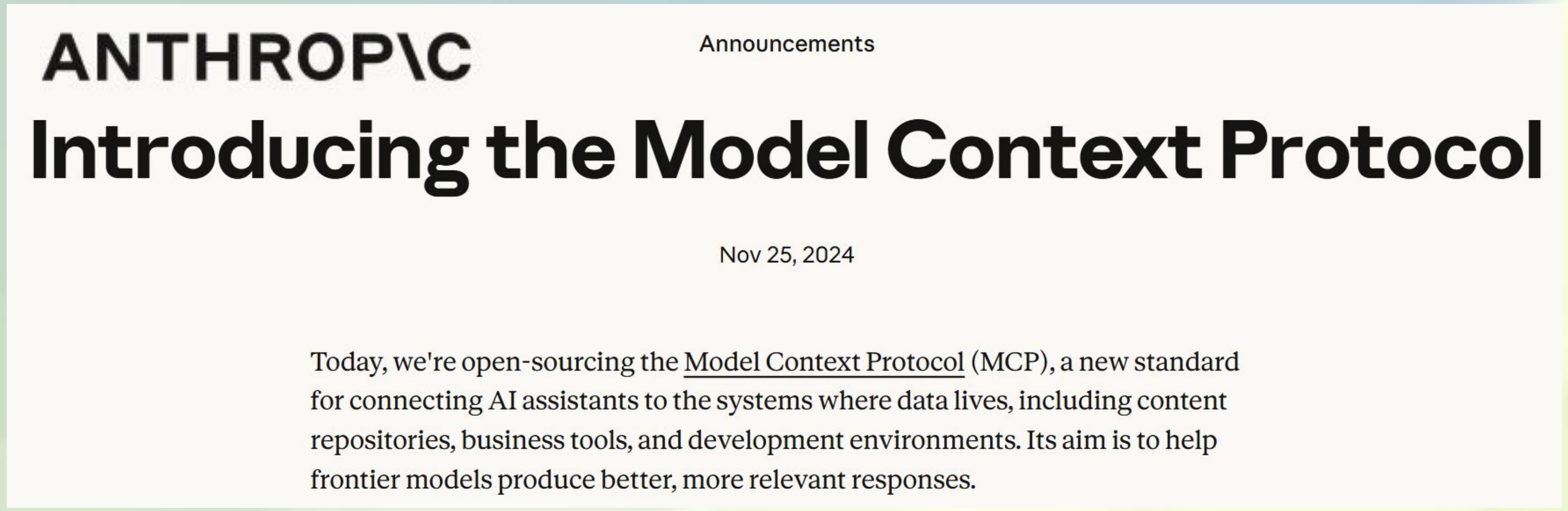
Copy page

MCP (Model Context Protocol) is an open-source standard for connecting AI applications to external systems.

Using MCP, AI applications like Claude or ChatGPT can connect to data sources (e.g. local files, databases), tools (e.g. search engines, calculators) and workflows (e.g. specialized prompts)—enabling them to access key information and perform tasks.

Think of MCP like a USB-C port for AI applications. Just as USB-C provides a standardized way to connect electronic devices, MCP provides a standardized way to connect AI applications to external systems.



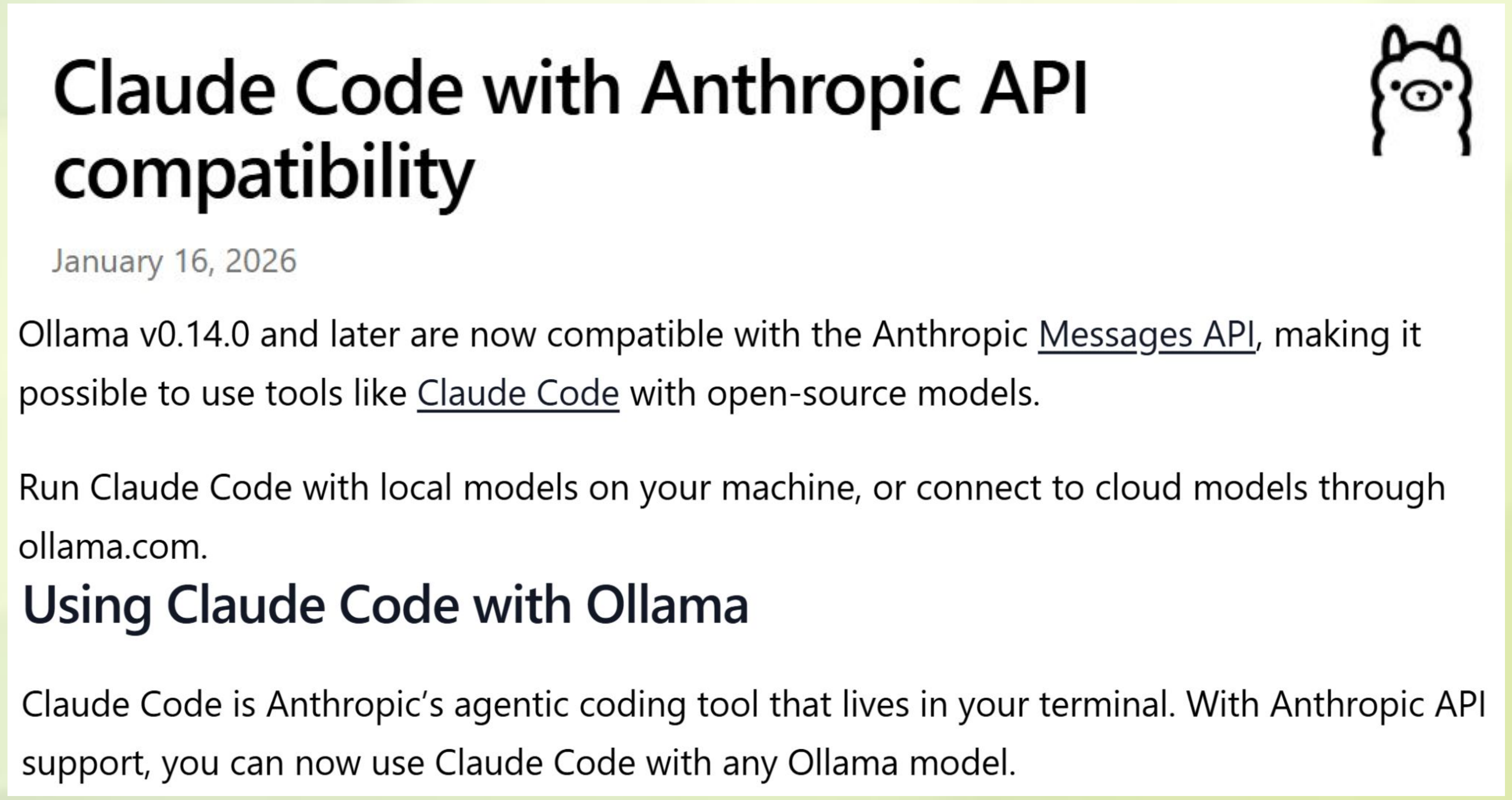


ANTHROP\C Announcements

Introducing the Model Context Protocol

Nov 25, 2024

Today, we're open-sourcing the Model Context Protocol (MCP), a new standard for connecting AI assistants to the systems where data lives, including content repositories, business tools, and development environments. Its aim is to help frontier models produce better, more relevant responses.



Claude Code with Anthropic API compatibility

January 16, 2026

Ollama v0.14.0 and later are now compatible with the Anthropic Messages API, making it possible to use tools like Claude Code with open-source models.

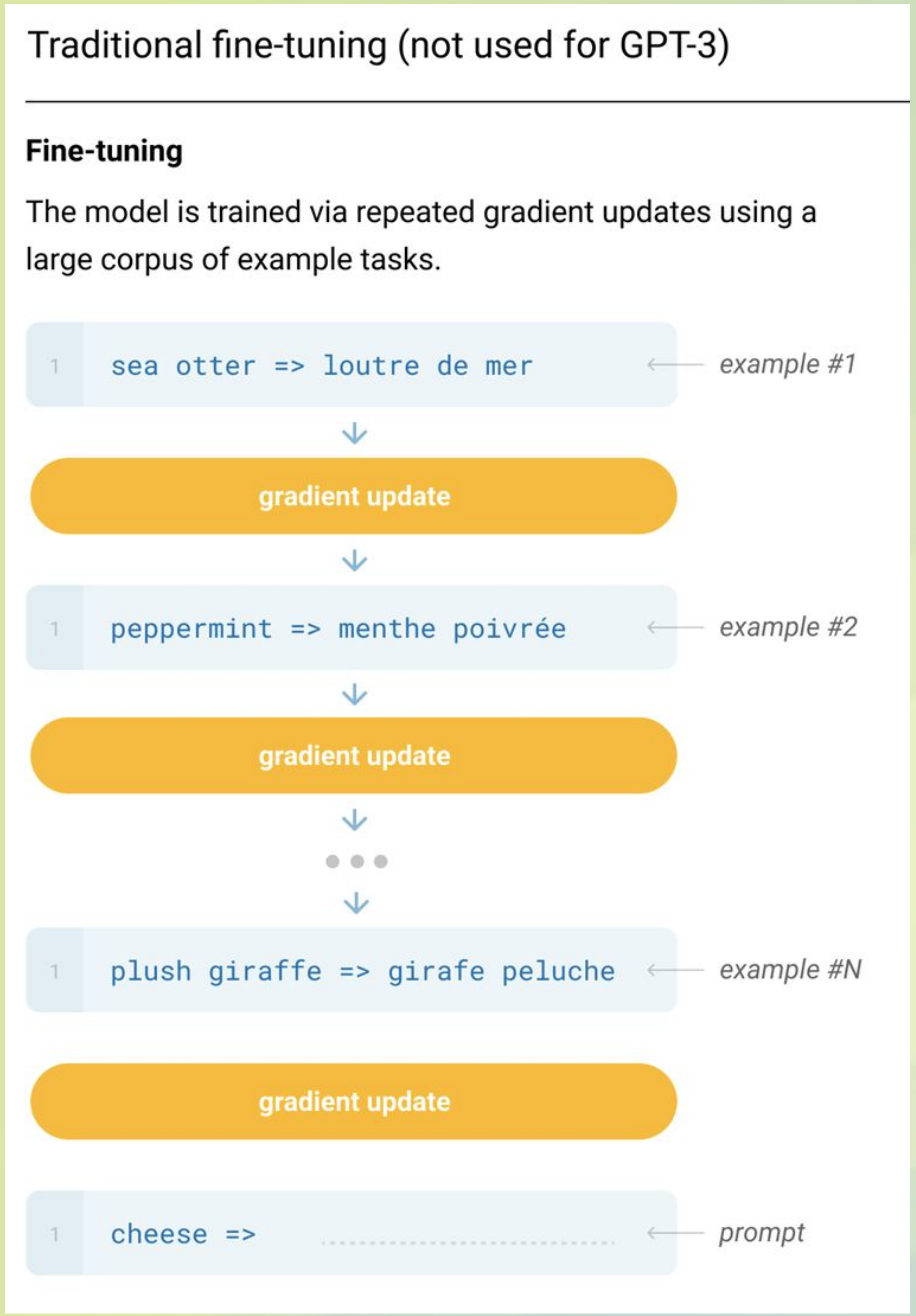
Run Claude Code with local models on your machine, or connect to cloud models through ollama.com.

Using Claude Code with Ollama

Claude Code is Anthropic's agentic coding tool that lives in your terminal. With Anthropic API support, you can now use Claude Code with any Ollama model.

- Introducing the Model Context Protocol \ Anthropic
- What is the Model Context Protocol (MCP)? - Model Context Protocol
- Claude Code with Anthropic API compatibility · Ollama Blog

Prompt Engineering: The Power of In-Context Learning


Traditional fine-tuning (not used for GPT-3)

Fine-tuning

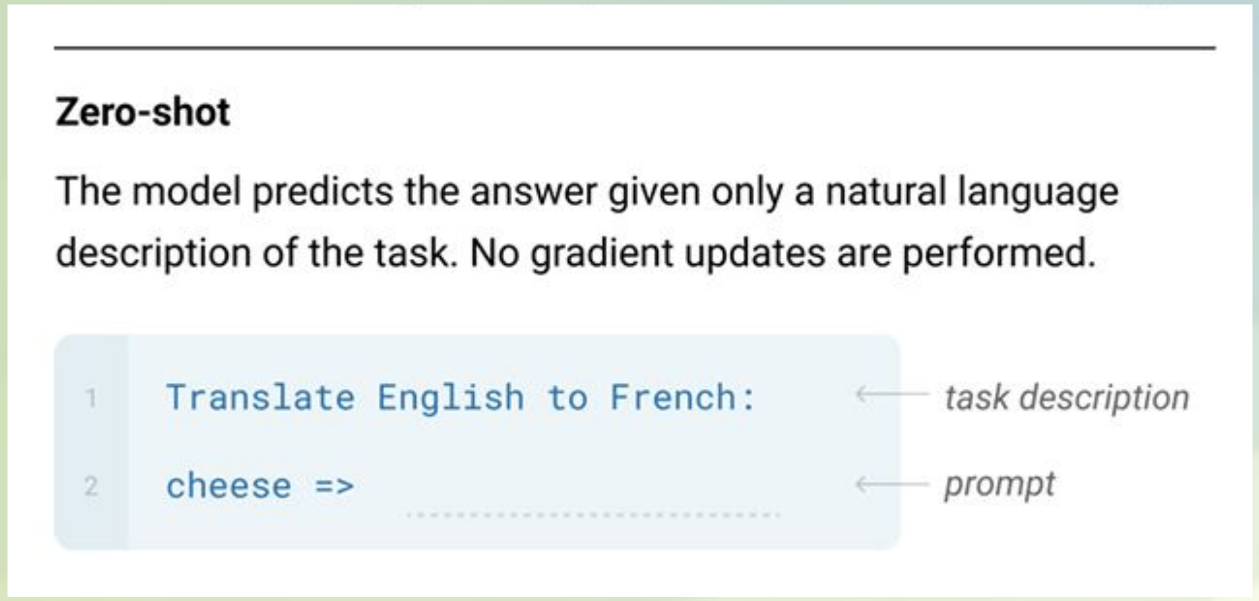
The model is trained via repeated gradient updates using a large corpus of example tasks.

```
sea otter => loutre de mer
```

```
peppermint => menthe poivrée
```

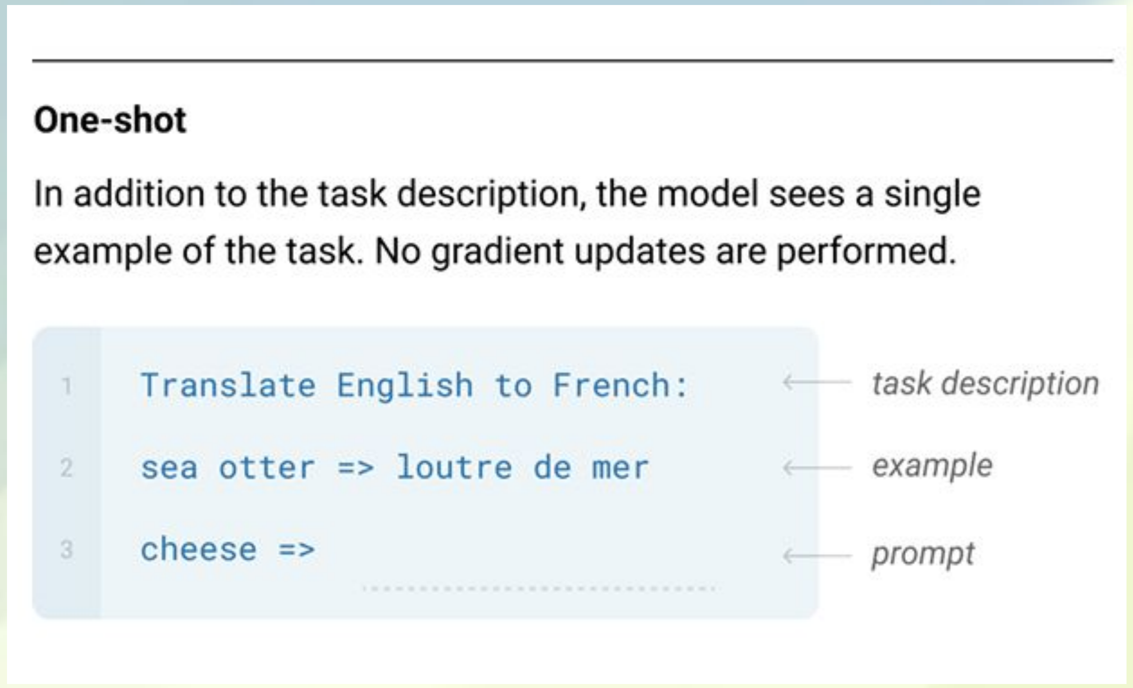
```
plush giraffe => girafe peluche
```

```
cheese =>
```


Zero-shot

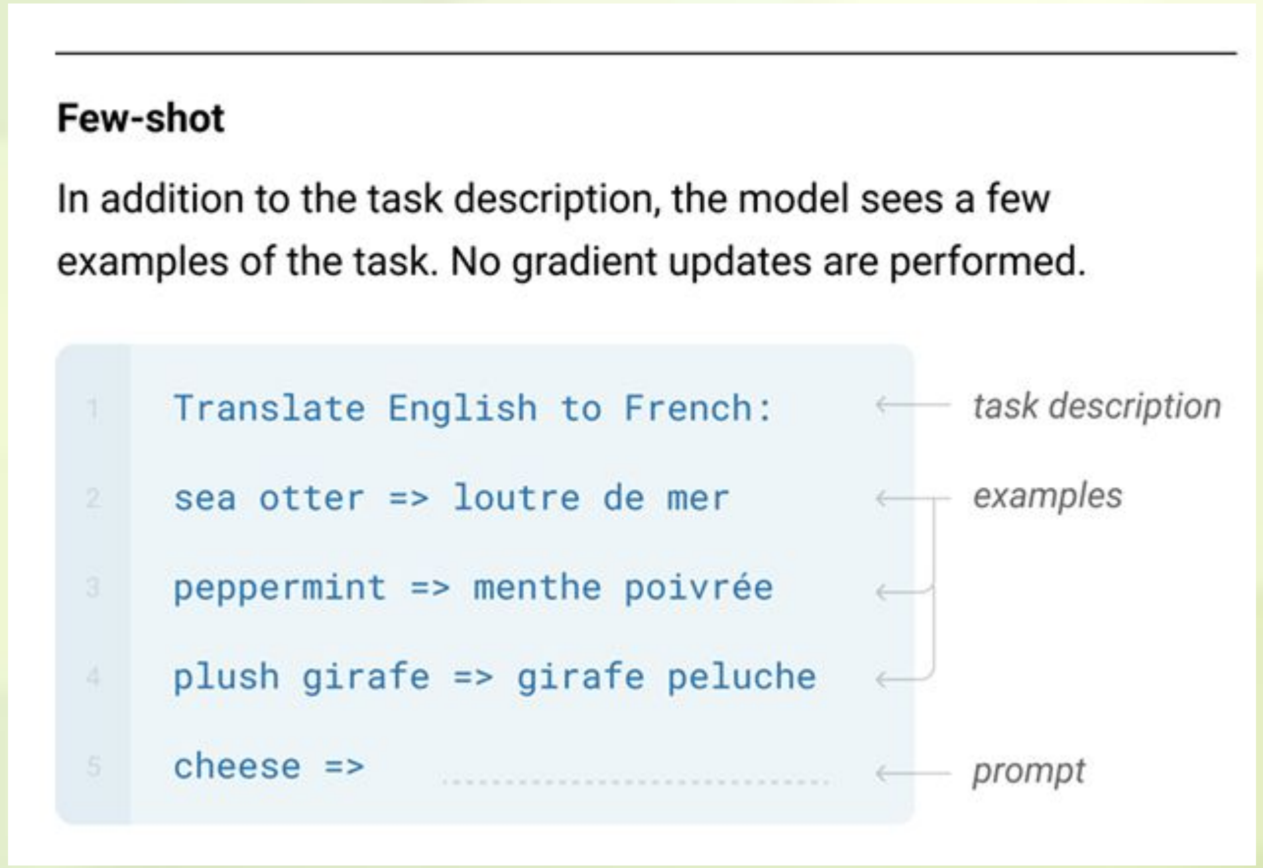
The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
Translate English to French:
cheese =>
```


One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

```
Translate English to French:
sea otter => loutre de mer
cheese =>
```


Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
Translate English to French:
sea otter => loutre de mer
peppermint => menthe poivrée
plush girafe => girafe peluche
cheese =>
```

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

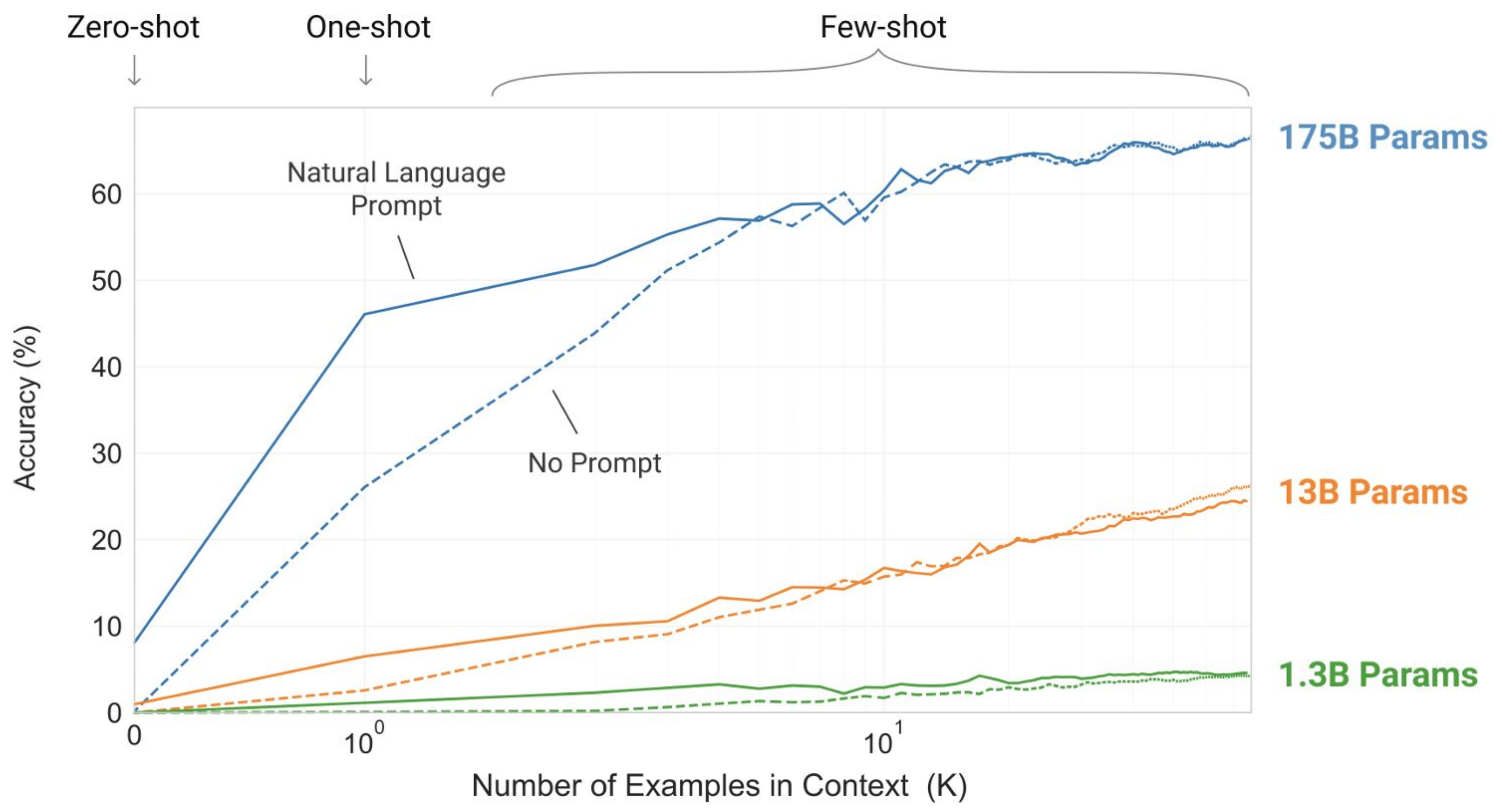


Figure 1.2: Larger models make increasingly efficient use of in-context information. We show in-context learning performance on a simple task requiring the model to remove random symbols from a word, both with and without a natural language task description (see Sec. 3.9.2). The steeper "in-context learning curves" for large models demonstrate improved ability to learn a task from contextual information. We see qualitatively similar behavior across a wide range of tasks.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

In-Context Learning. Let $\mathcal{X}$ denote an input space and $\mathcal{Y}$ an output space. A language model with parameters θ defines a conditional distribution over token sequences $\mathbf{t} = (t_1, \ldots, t_T)$.

$$p_\theta(\mathbf{t}) = \prod_{i=1}^{T} p_\theta(t_i \mid t_{<i}),$$

For an in-context learning (ICL) task, the model is provided with a prompt consisting of K labeled examples, with $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$ for all $i = 1, .., K$, followed by the question/ task to be completed:

$$\mathcal{P} = \big[(x_1, y_1),\ (x_2, y_2),\ \ldots,\ (x_K, y_K),\ x_{K+1}\big].$$

Notice that each pair (x_i, y_i) is concatenated to form a single token sequence. The in-context learning objective is to predict the output y_{K+1} conditioned on the text prompt:

$$p_\theta(y_{K+1} \mid x_{K+1}, (x_1, y_1), \ldots, (x_K, y_K)).$$

No parameters are updated during in-context learning. Adaptation to the task is achieved solely through conditioning on the labeled examples in the prompt.