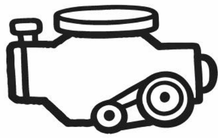
From Conventional LLMs to Reasoning Models to Agents

Sebastian Raschka, PhD (@rasbt)
https://sebastianraschka.com

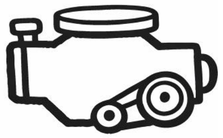
Aug 19, 2026 11:00 AM CDT

From Conventional LLMs

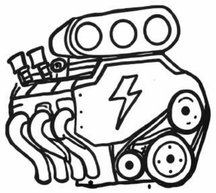


Conventional LLM

From Conventional LLMs to Reasoning Models

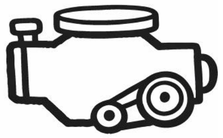


Conventional LLM

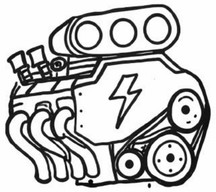


Reasoning LLM
(more expensive to run)

From Conventional LLMs to Reasoning Models to Agents



Conventional LLM

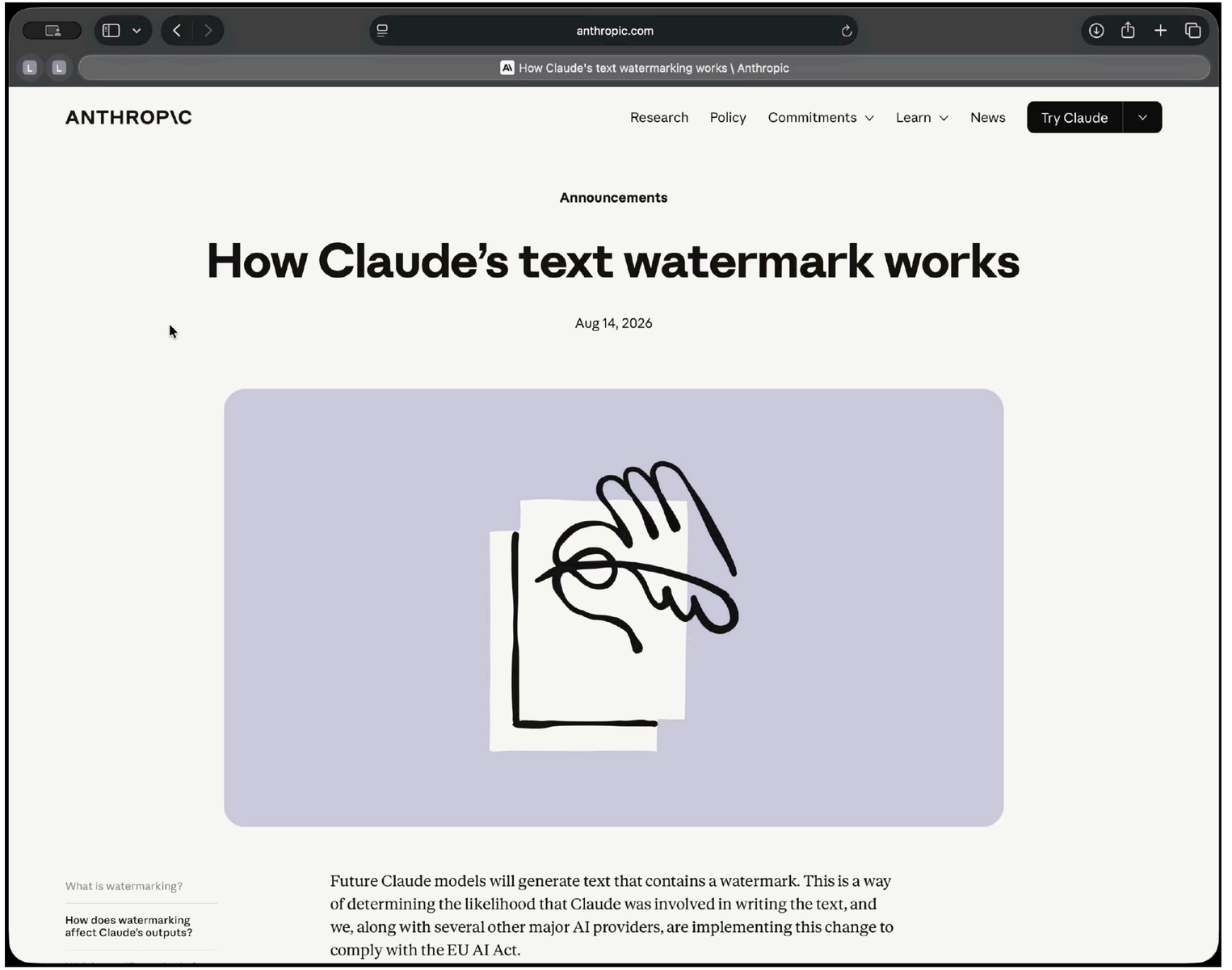


Reasoning LLM
(more expensive to run)

LLM in agent harness

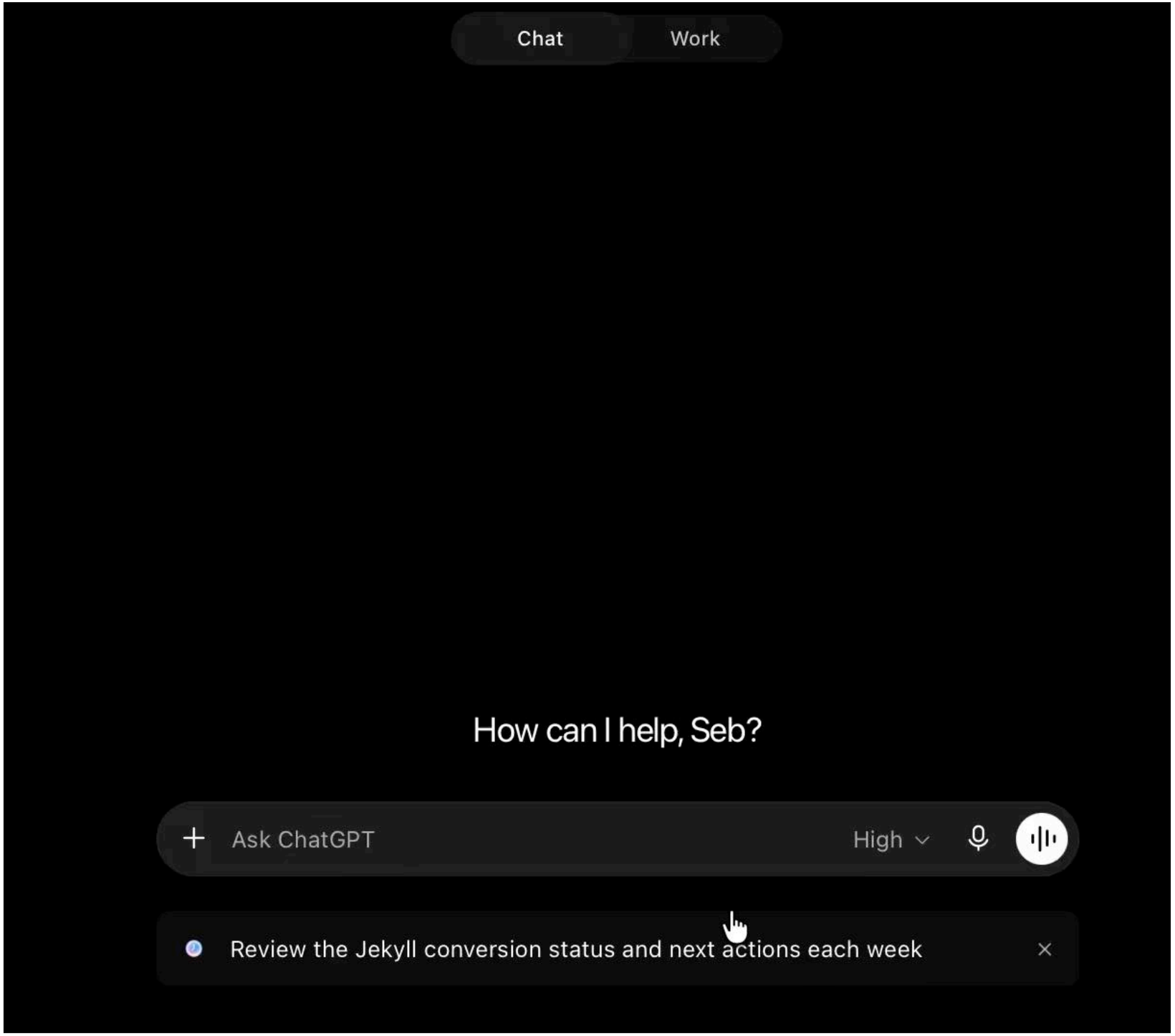
Benefits of “from scratch”

(and how conventional LLMs work…)



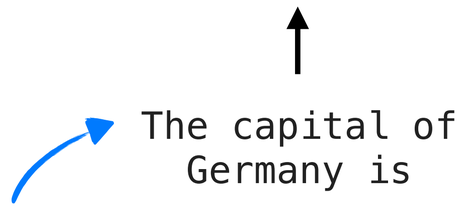
https://www.anthropic.com/news/claude-text-watermark

Prelude: text generation in LLMs


Chat
Work
How can I help, Seb?
Ask ChatGPT
High
Review the Jekyll conversion status and next actions each week

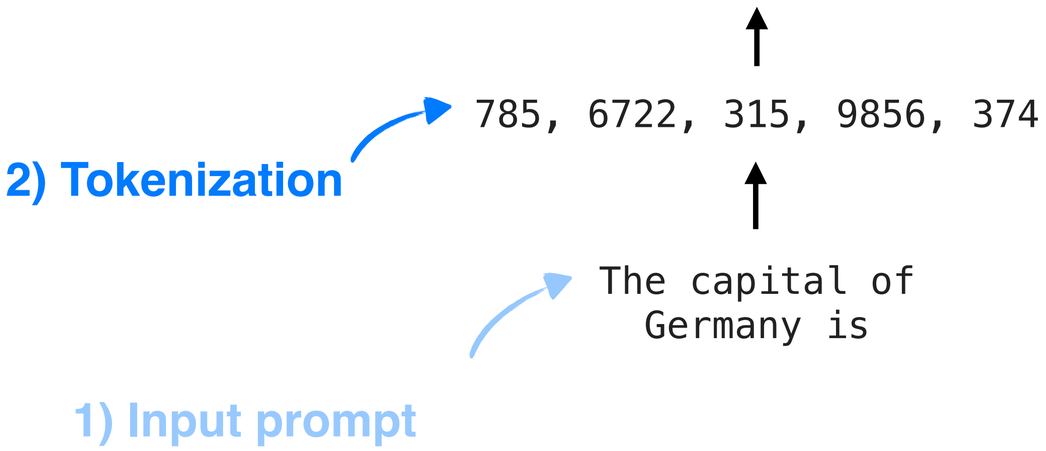
So what happens under the hood when the next token is generated?

Generating the next token

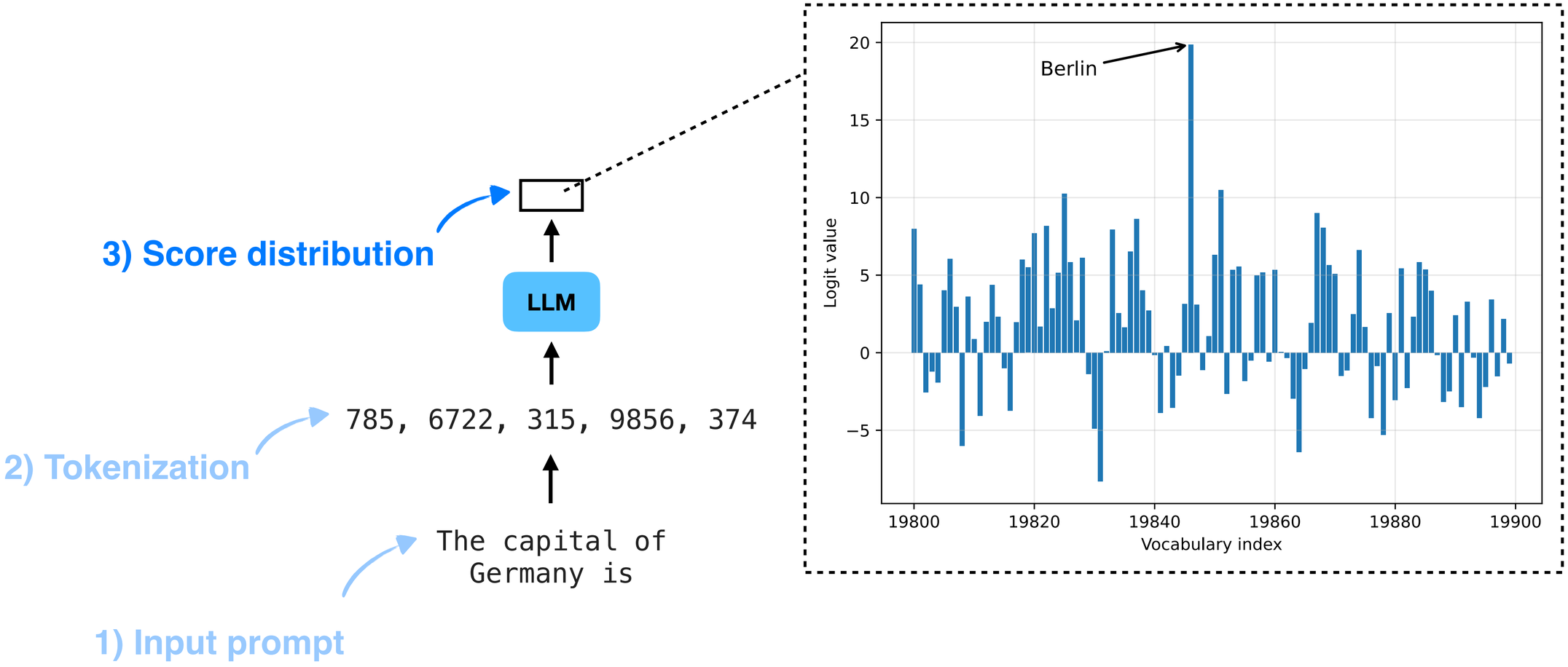


1) Input prompt

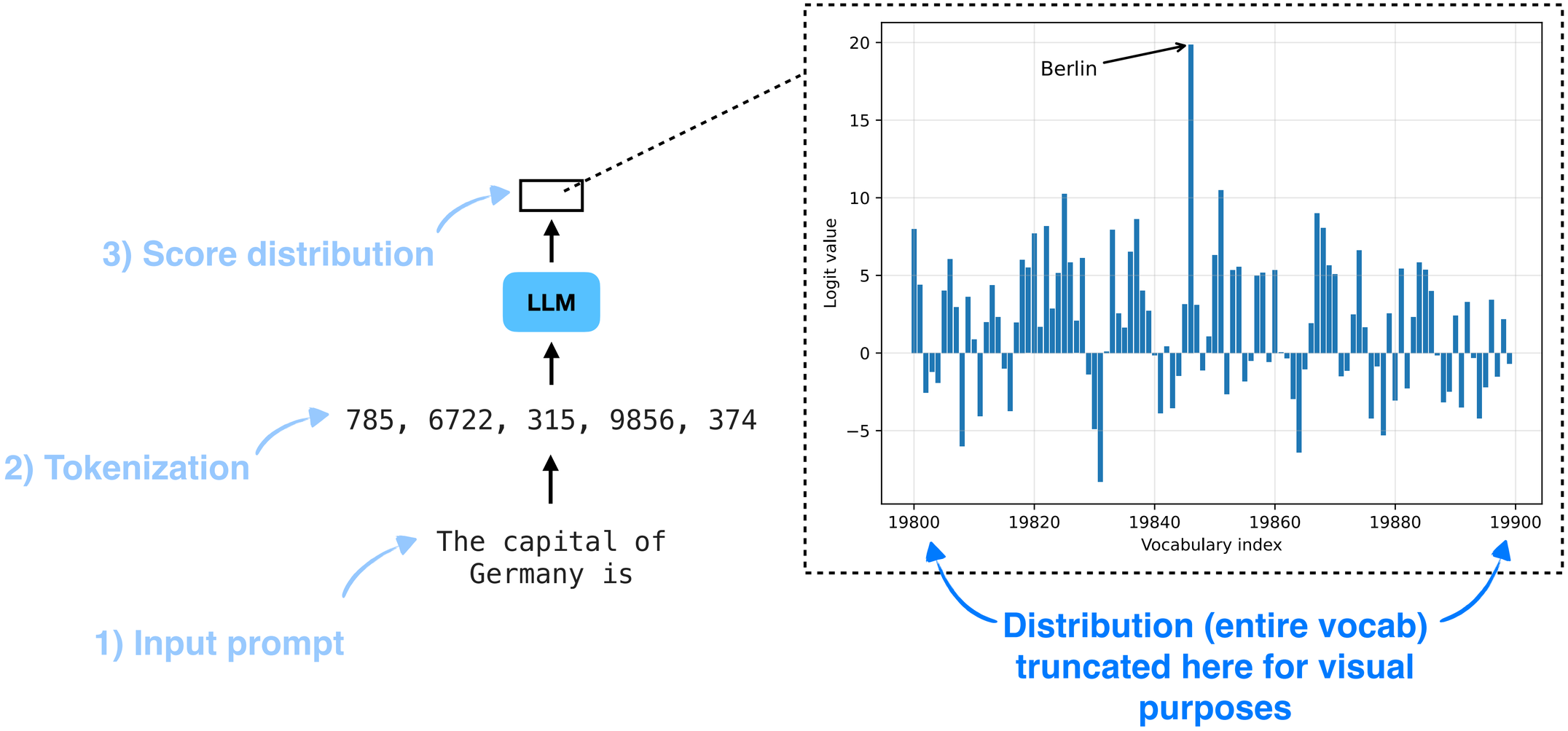
Generating the next token



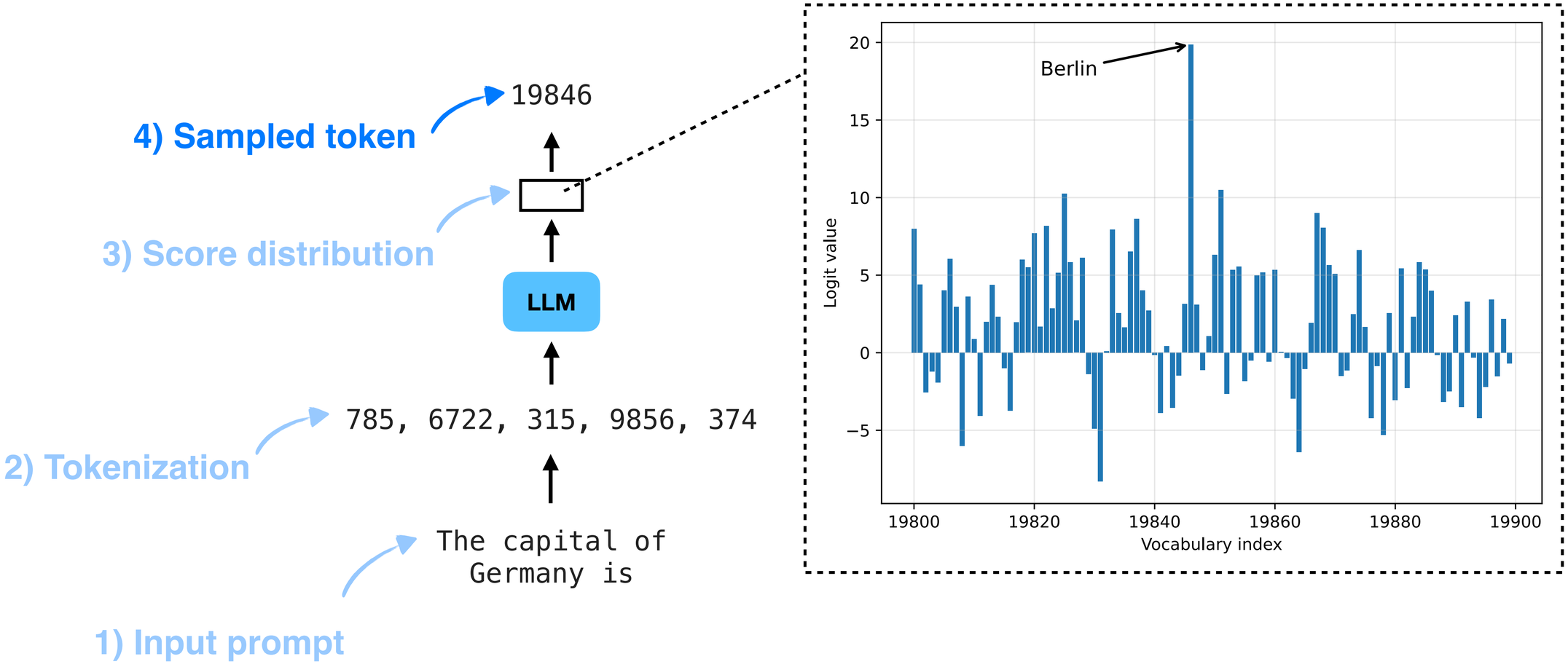
Generating the next token



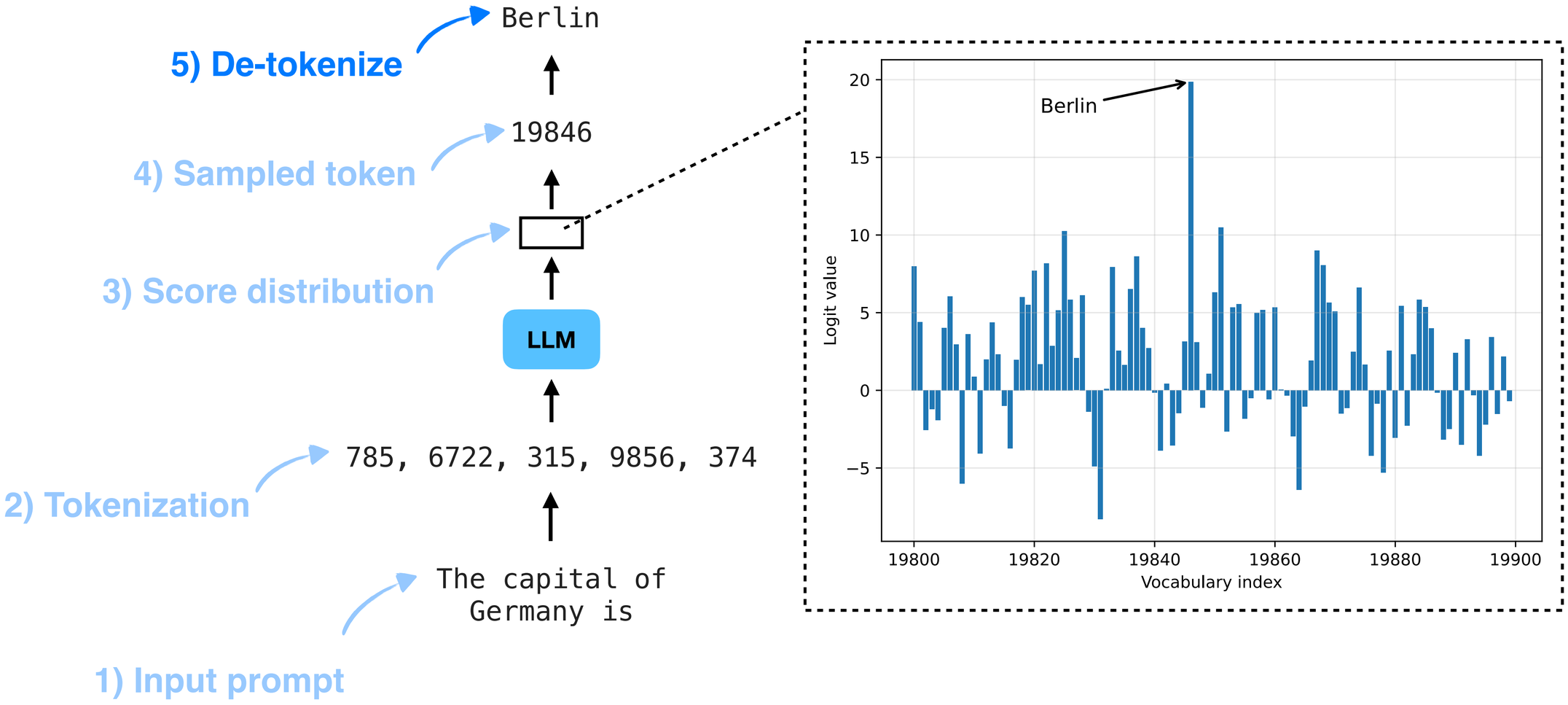
Generating the next token



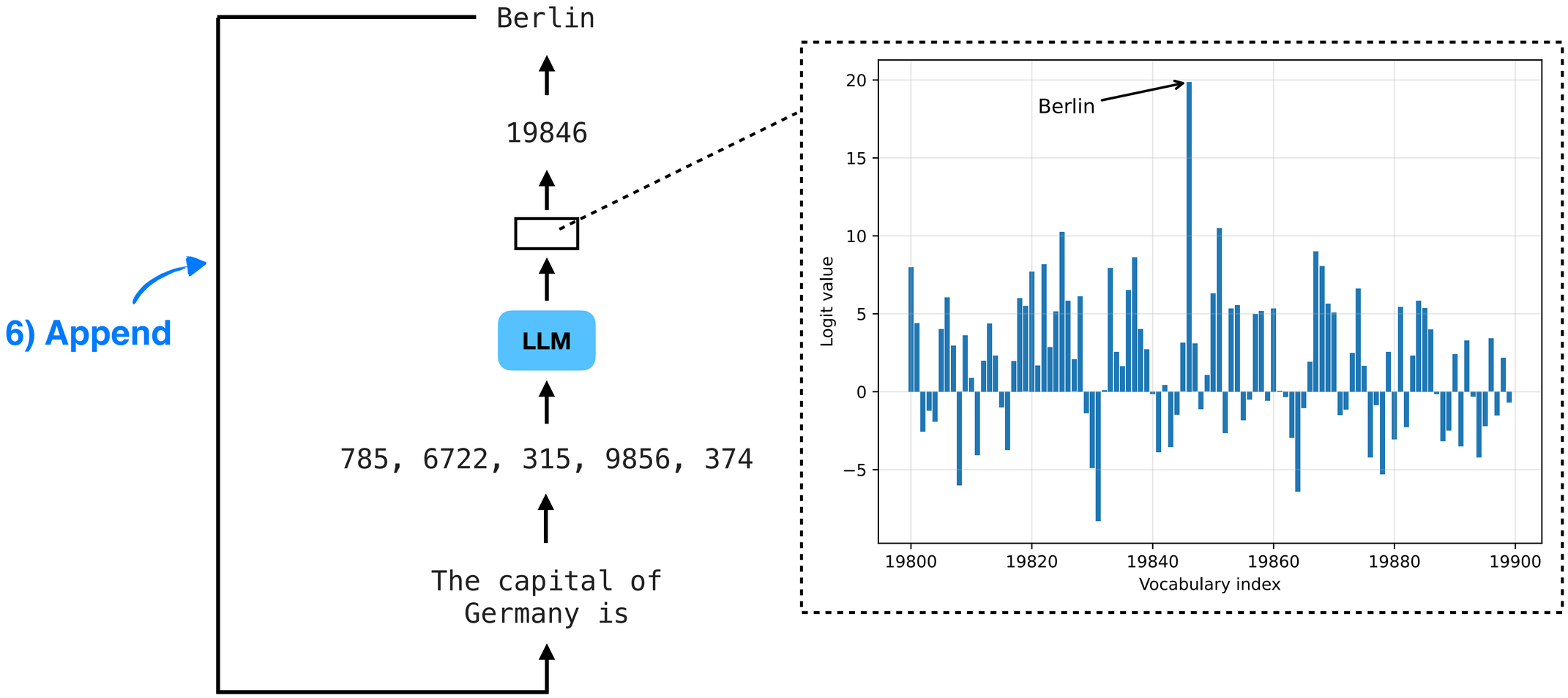
Generating the next token



Generating the next token

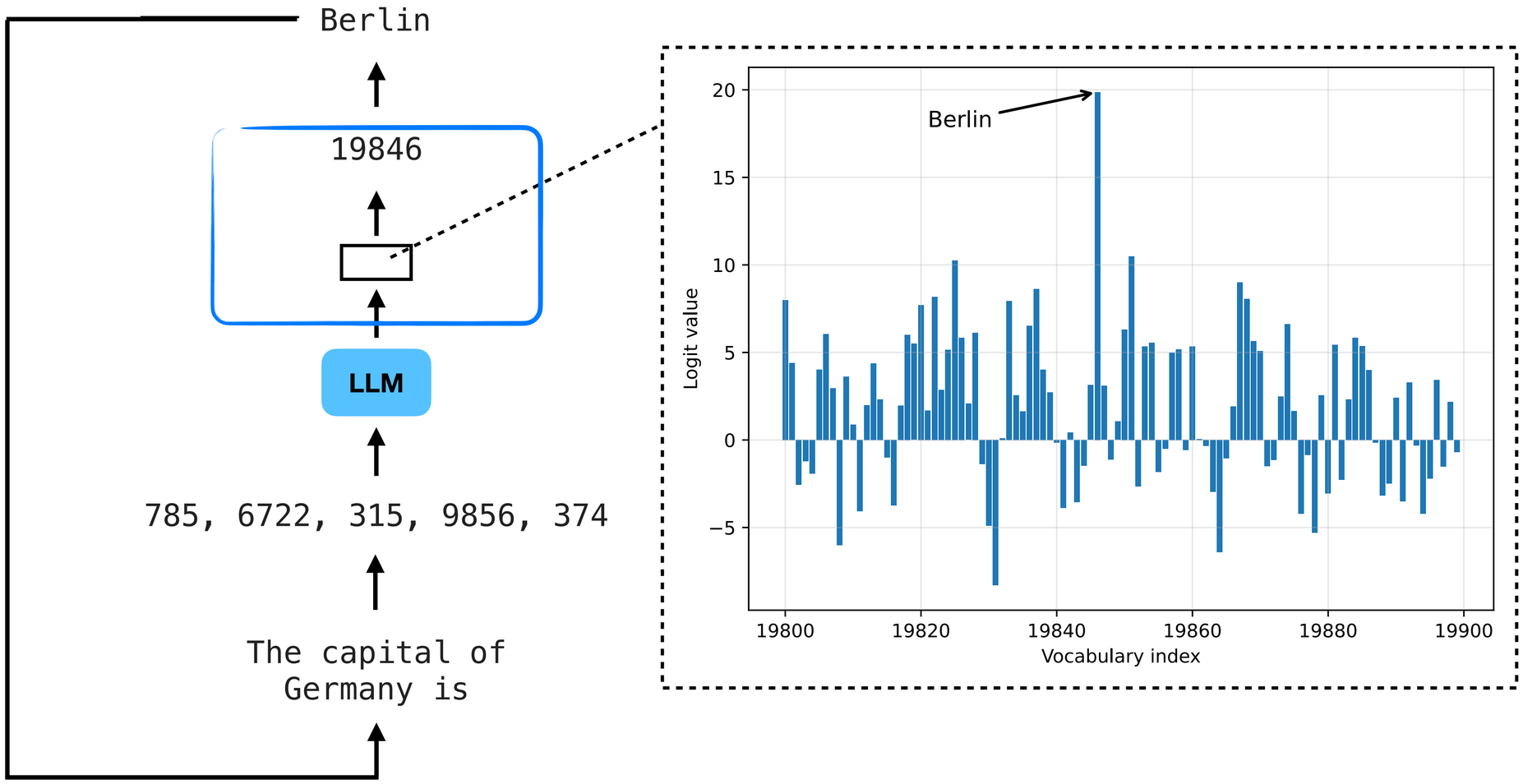


Generating the next token

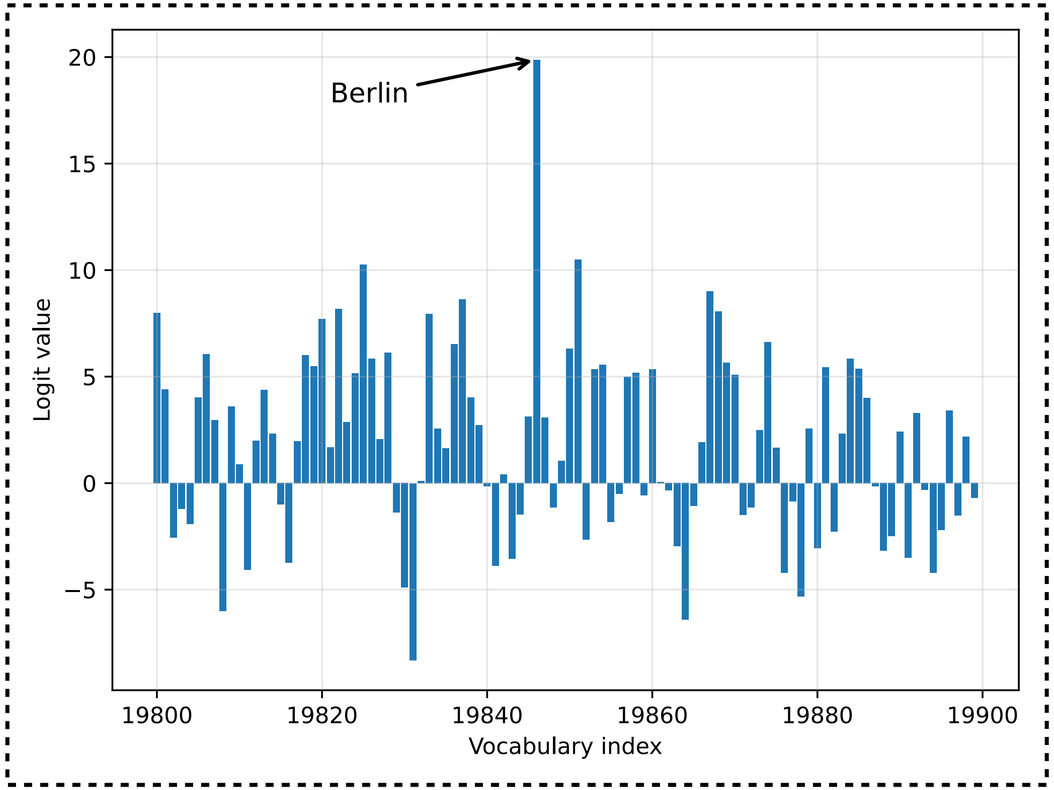


Generating the next token

How do we sample the next token?



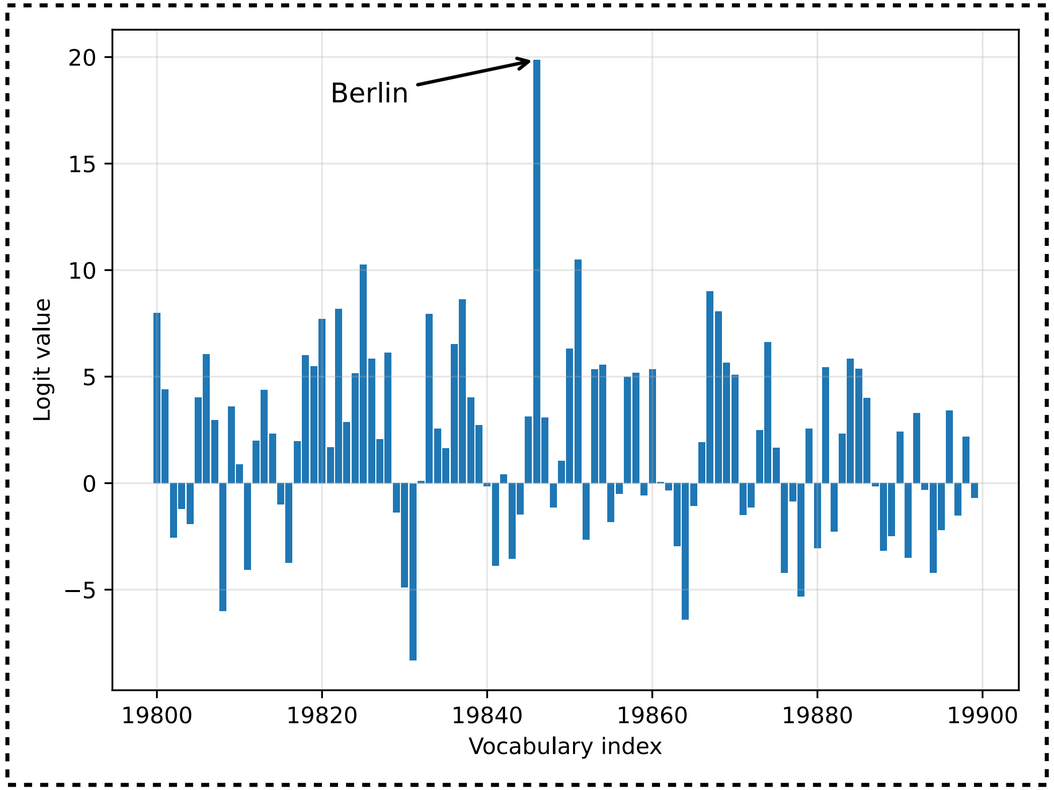
Token sampling




```python
import numpy as np

scores = np.array([
     7.9,  4.4, -2.6, -1.2, -1.9,  4.0,  6.0,  3.0, -6.0,  3.6,
     0.8, -4.0,  2.0,  4.4,  2.3, -1.0, -3.7,  1.9,  6.0,  5.5,
     7.7,  1.7,  8.2,  2.8,  5.1, 10.2,  5.8,  2.1,  6.1, -1.3,
    -4.9, -8.3,  0.1,  7.9,  2.6,  1.7,  6.5,  8.6,  4.0,  2.7,
    -0.2, -3.9,  0.4, -3.6, -1.5,  3.1, 19.8,  3.1, -1.1,  1.0,
     6.3, 10.5, -2.6,  5.3,  5.5, -1.8, -0.5,  5.0,  5.1, -0.6,
     5.3,  0.0, -0.3, -3.0, -6.4, -1.1,  1.9,  9.0,  8.0,  5.6,
     5.0, -1.5, -1.1,  2.5,  6.6,  1.7, -4.2, -0.8, -5.3,  2.6,
    -3.0,  5.4, -2.2,  2.3,  5.8,  5.4,  4.0, -0.2, -3.1, -2.5,
     2.4, -3.5,  3.3, -0.3, -4.2, -2.1,  3.4, -1.5,  2.1, -0.7
])

vocab_indices = np.arange(19800, 19900)

# Softmax
probs = np.exp(scores) / np.exp(scores).sum()
```

Convert to probabilities (all values sum to 1)

Token sampling

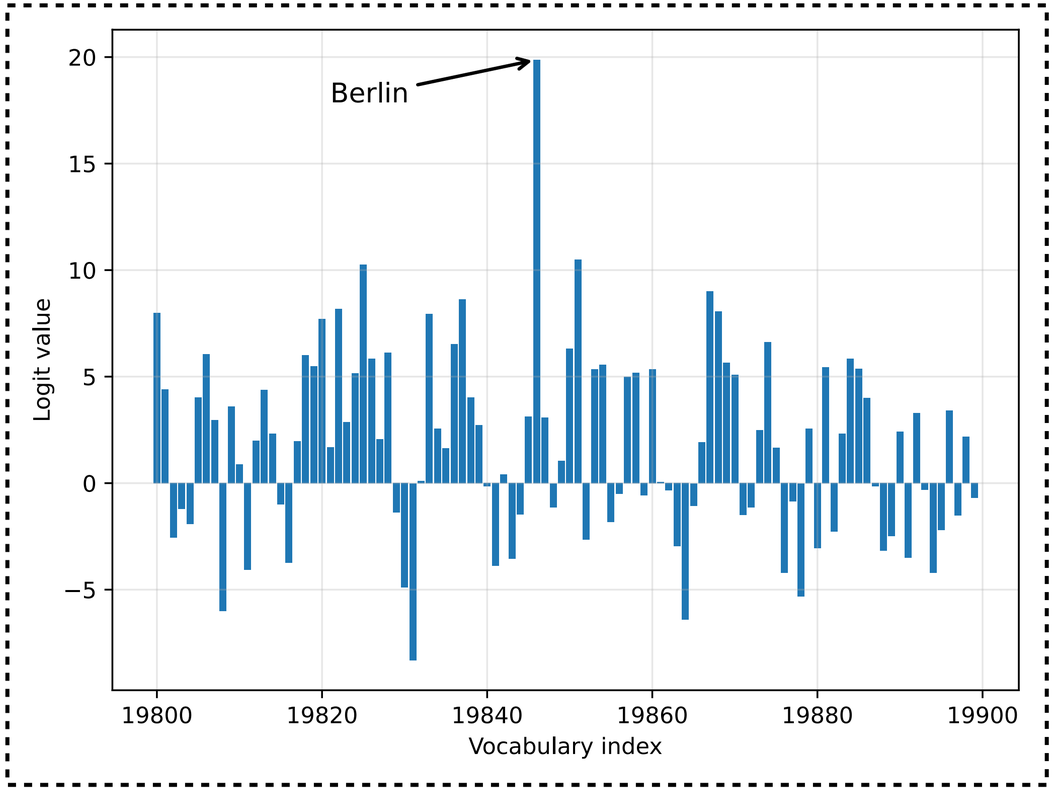



```python
import numpy as np

scores = np.array([
     7.9,  4.4, -2.6, -1.2, -1.9,  4.0,  6.0,  3.0, -6.0,  3.6,
     0.8, -4.0,  2.0,  4.4,  2.3, -1.0, -3.7,  1.9,  6.0,  5.5,
     7.7,  1.7,  8.2,  2.8,  5.1, 10.2,  5.8,  2.1,  6.1, -1.3,
    -4.9, -8.3,  0.1,  7.9,  2.6,  1.7,  6.5,  8.6,  4.0,  2.7,
    -0.2, -3.9,  0.4, -3.6, -1.5,  3.1, 19.8,  3.1, -1.1,  1.0,
     6.3, 10.5, -2.6,  5.3,  5.5, -1.8, -0.5,  5.0,  5.1, -0.6,
     5.3,  0.0, -0.3, -3.0, -6.4, -1.1,  1.9,  9.0,  8.0,  5.6,
     5.0, -1.5, -1.1,  2.5,  6.6,  1.7, -4.2, -0.8, -5.3,  2.6,
    -3.0,  5.4, -2.2,  2.3,  5.8,  5.4,  4.0, -0.2, -3.1, -2.5,
     2.4, -3.5,  3.3, -0.3, -4.2, -2.1,  3.4, -1.5,  2.1, -0.7
])

vocab_indices = np.arange(19800, 19900)

# Softmax
probs = np.exp(scores) / np.exp(scores).sum()

rng = np.random.default_rng(123)
sample = rng.choice(vocab_indices, p=probs)
print(f"{sample} -> {tokenizer.decode([sample])}")
```

```
19846 ->  Berlin
```

Sample token from probability distribution (probability = “contribution weights”)

Token sampling




```python
samples = rng.choice(vocab_indices, size=10_000, replace=True, p=probs)

tokens, counts = np.unique(samples, return_counts=True)

top5 = np.argsort(counts)[-5:][::-1]

for sample, count in zip(tokens[top5], counts[top5]):
    print(f"{sample} -> {tokenizer.decode([sample])}: {count}x")
```

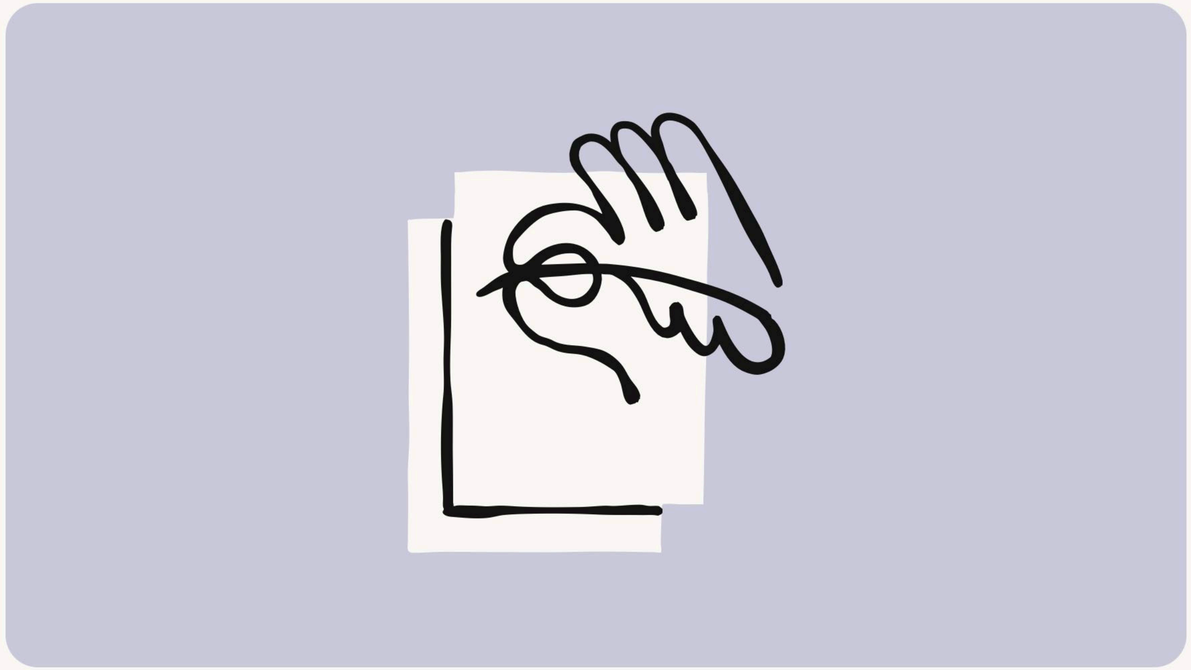
```
19846 ->  Berlin: 9997x
19851 ->  Hal: 2x
19822 ->  Moh: 1x
```

E.g., if we sample 10k times, “Berlin” would be selected most of the time

Announcements

How Claude’s text watermark works

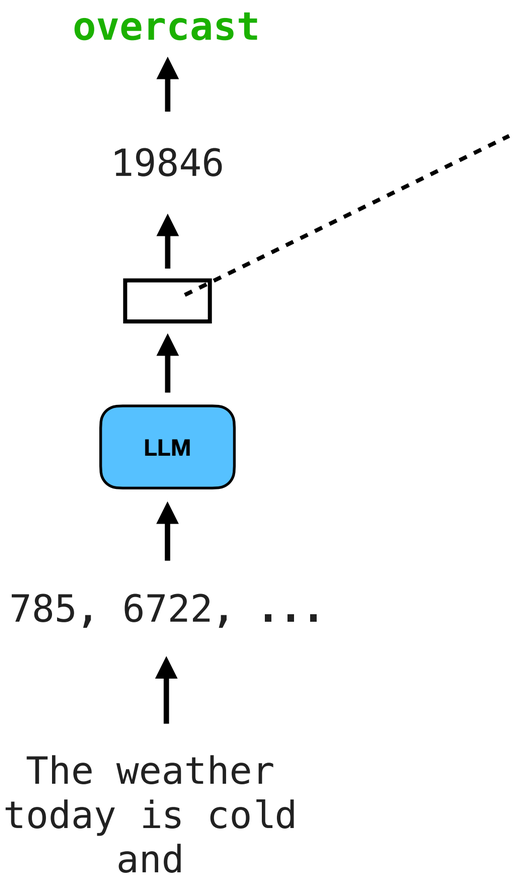
Aug 14, 2026

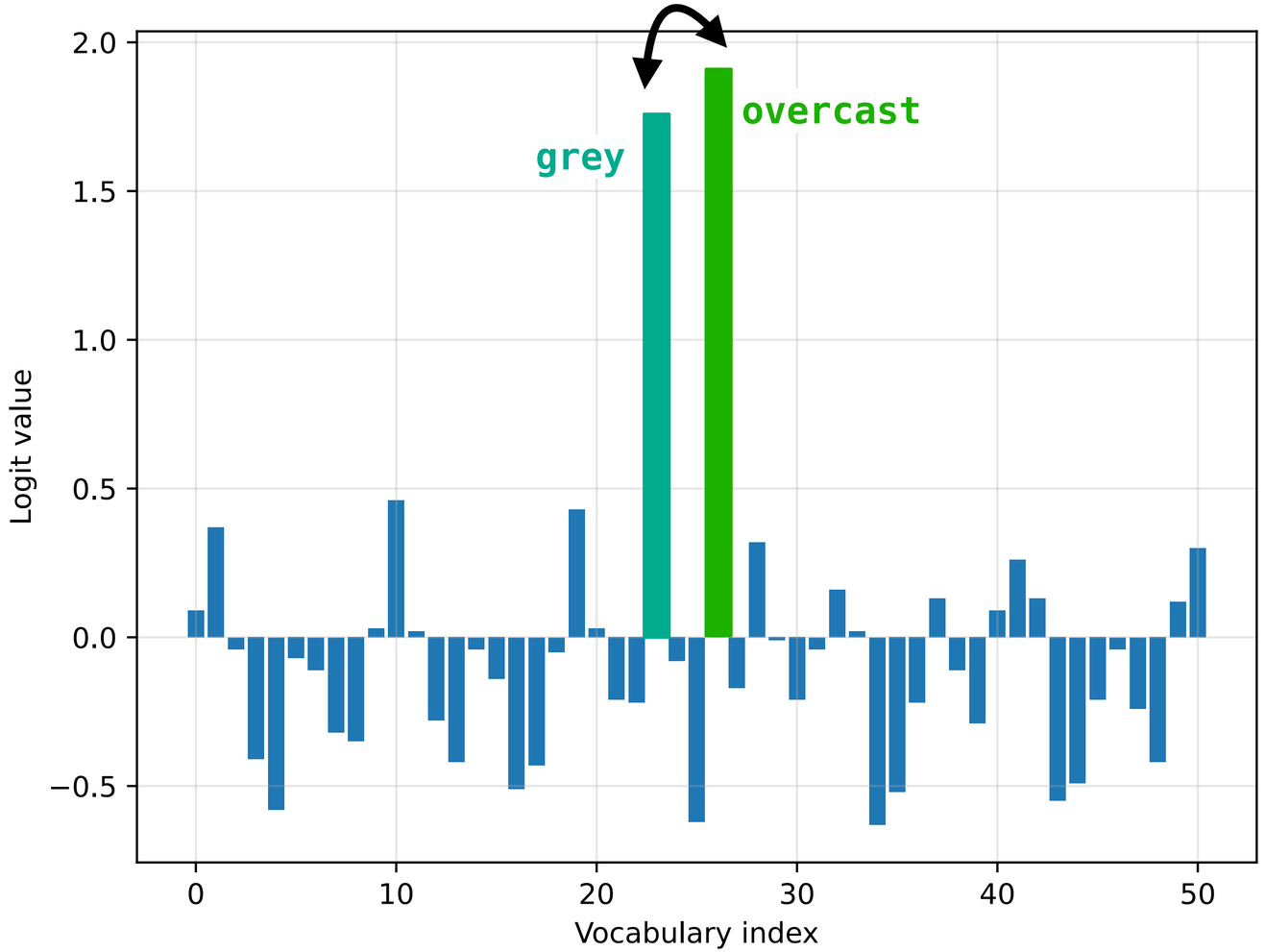


https://www.anthropic.com/news/claude-text-watermark

Without watermarking:

Due to random sampling we select either of these likely tokens





Side note: random number generation

```
[ ]: import numpy as np


     for i in range(5):
         print(np.random.randint(100))
```

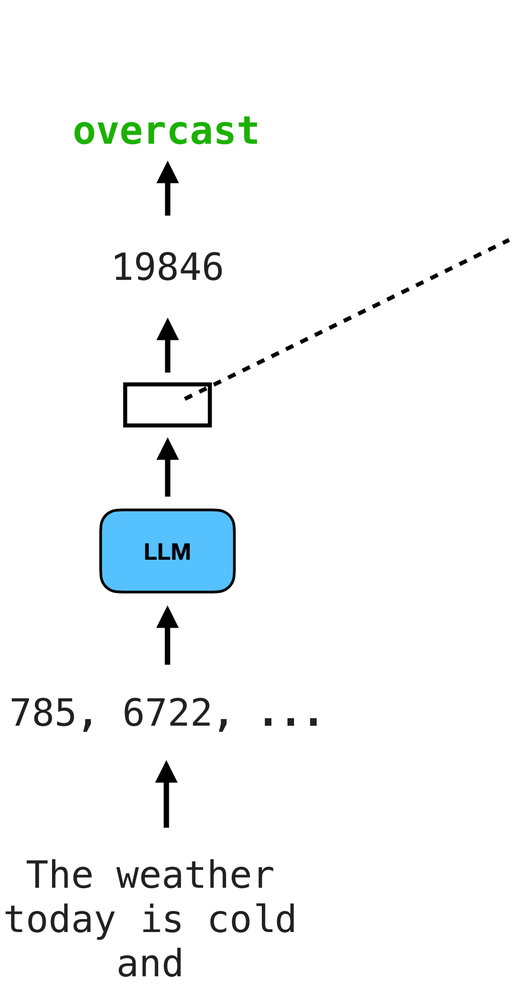
```
[ ]:
```

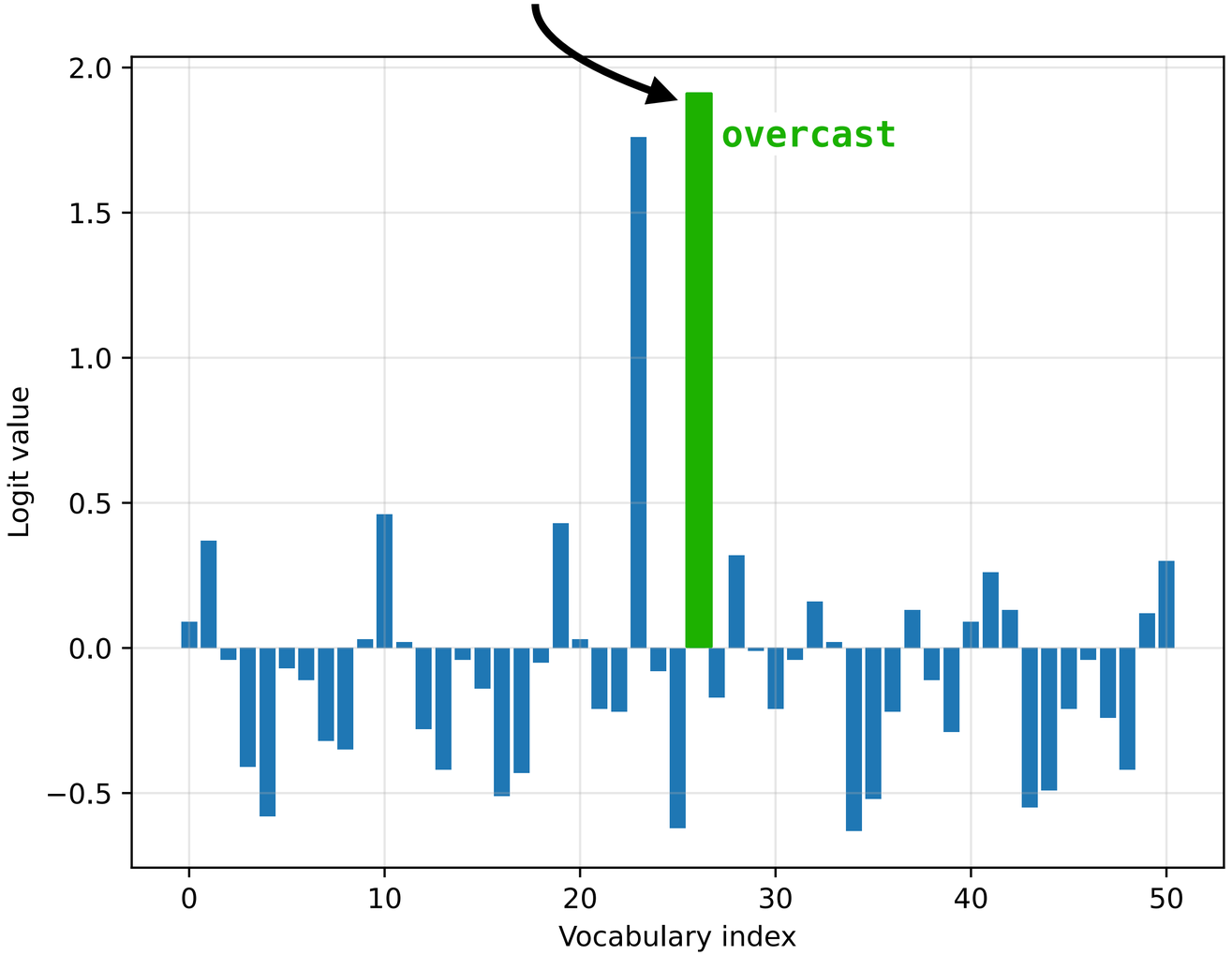
Side note: random seeds add reproducibility

[5]:
```
import numpy as np


for i in range(5):
    print(np.random.randint(100))
```

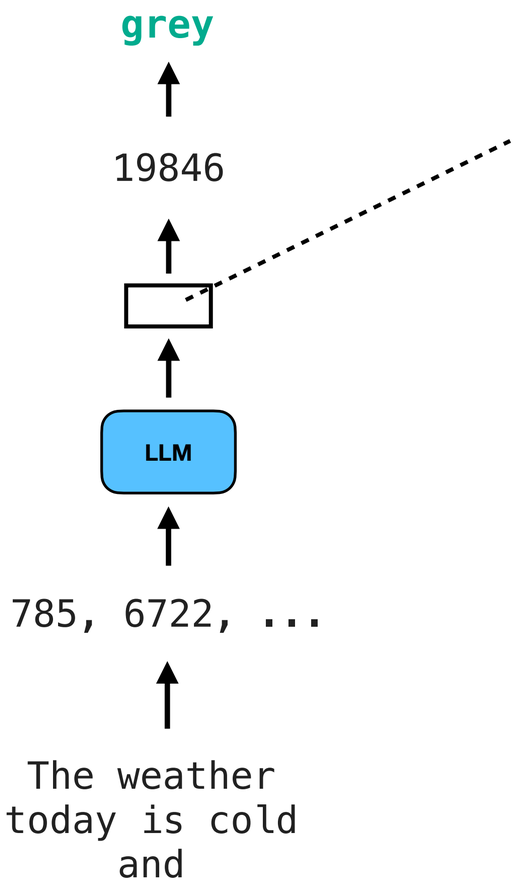
```
3
65
9
64
1
```

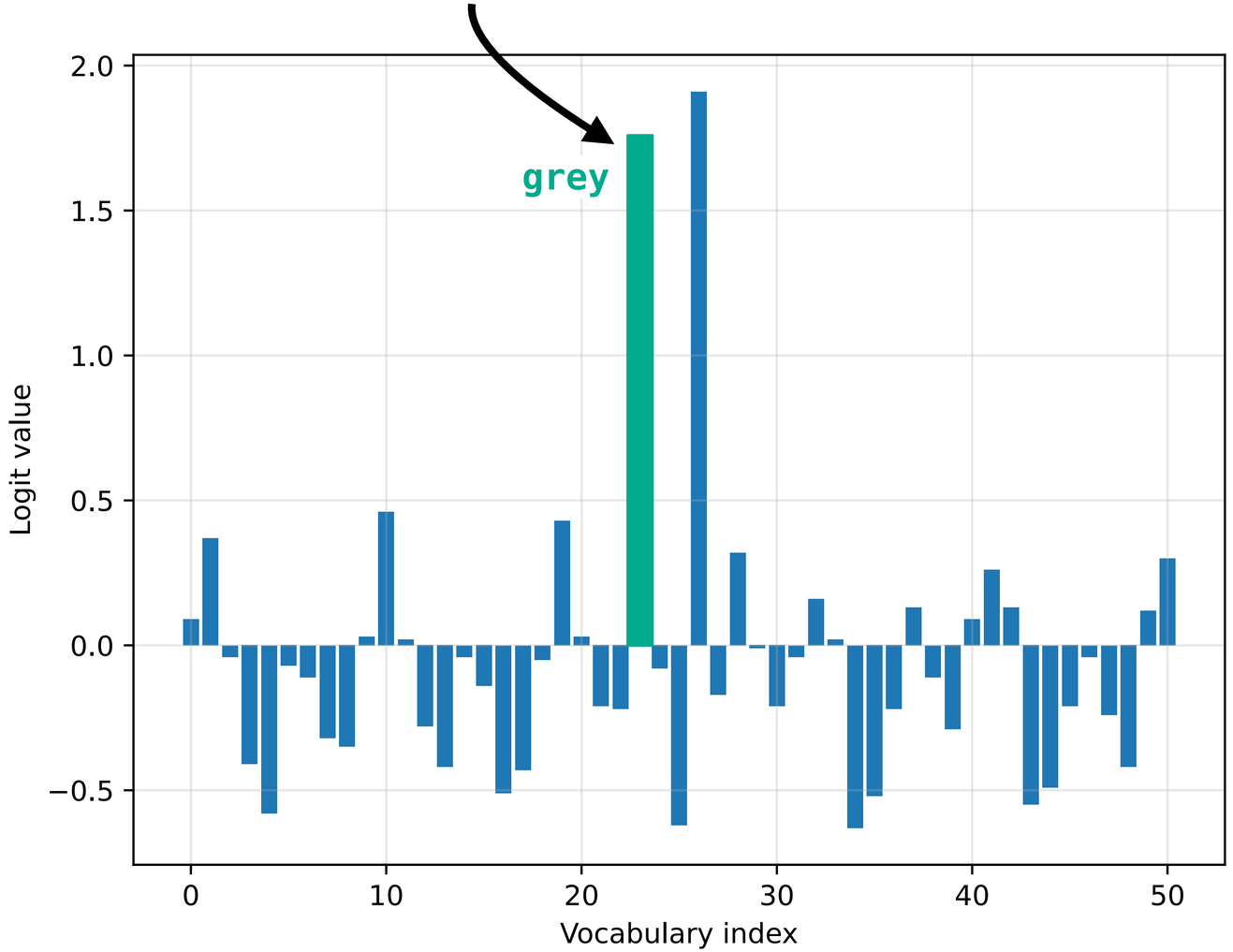
With random seed 42 we always select this



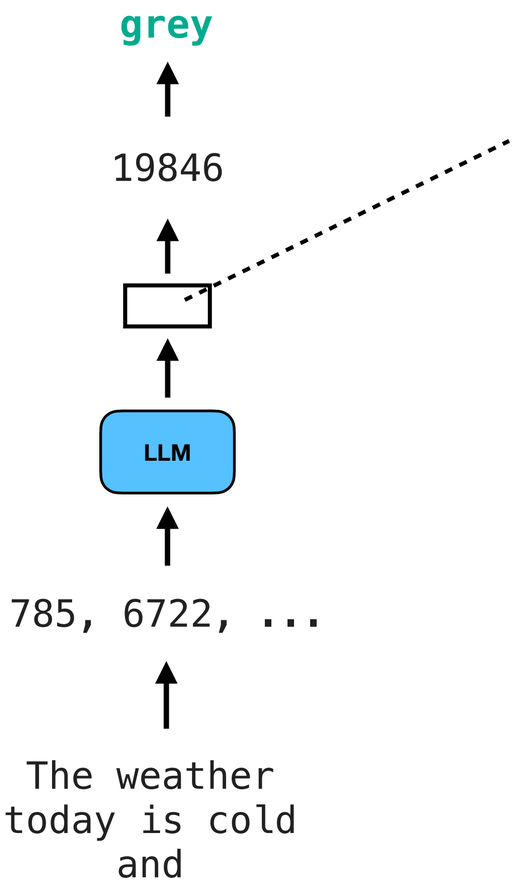


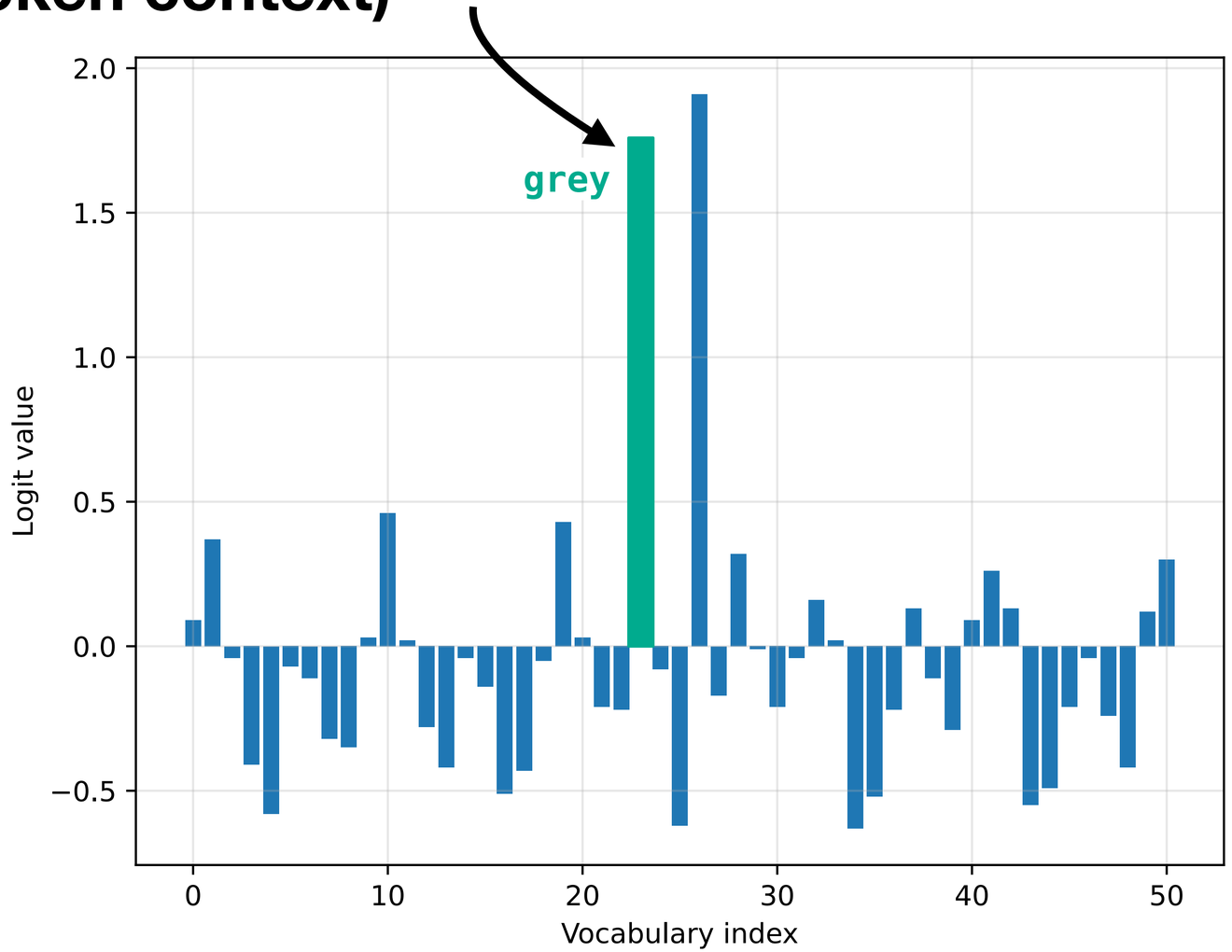
With random seed 99 we always select this





Watermarking sets random seed based on secret key (and preceding token context)

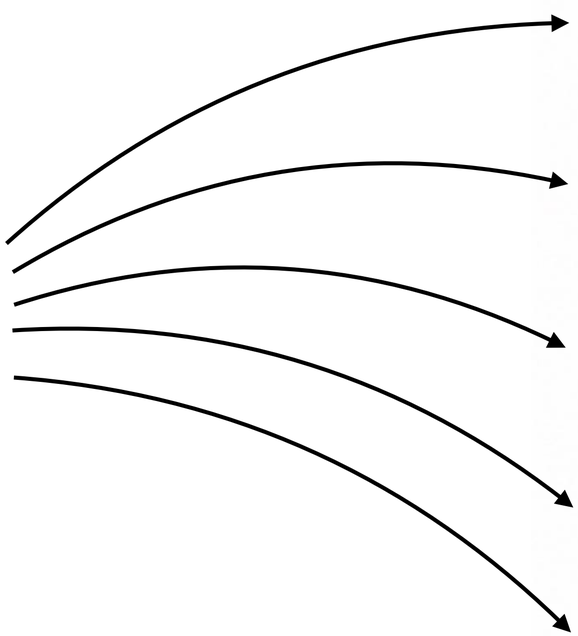




Without watermarking might get any of these texts

Plausible texts

The weather today is cold and **overcast** / **grey**. A **light** / **gentle** breeze is **moving** / **blowing** through the trees, and the streets seem **quiet** / **still**. I think I'll stay **home** / **inside** and read a **book** / **novel** with a **cup** / **mug** of tea.



1. The weather today is cold and **overcast**. A **light** breeze is **moving** through the trees, and the streets seem **quiet**. I think I'll stay **home** and read a **book** with a **cup** of tea.
2. The weather today is cold and **grey**. A **gentle** breeze is **blowing** through the trees, and the streets seem **still**. I think I'll stay **inside** and read a **novel** with a **mug** of tea.
3. The weather today is cold and **grey**. A **light** breeze is **blowing** through the trees, and the streets seem **quiet**. I think I'll stay **home** and read a **novel** with a **cup** of tea.
4. The weather today is cold and **overcast**. A **gentle** breeze is **moving** through the trees, and the streets seem **still**. I think I'll stay **inside** and read a **book** with a **mug** of tea.
5. The weather today is cold and **overcast**. A **gentle** breeze is **blowing** through the trees, and the streets seem **quiet**. I think I'll stay **inside** and read a **book** with a **cup** of tea.

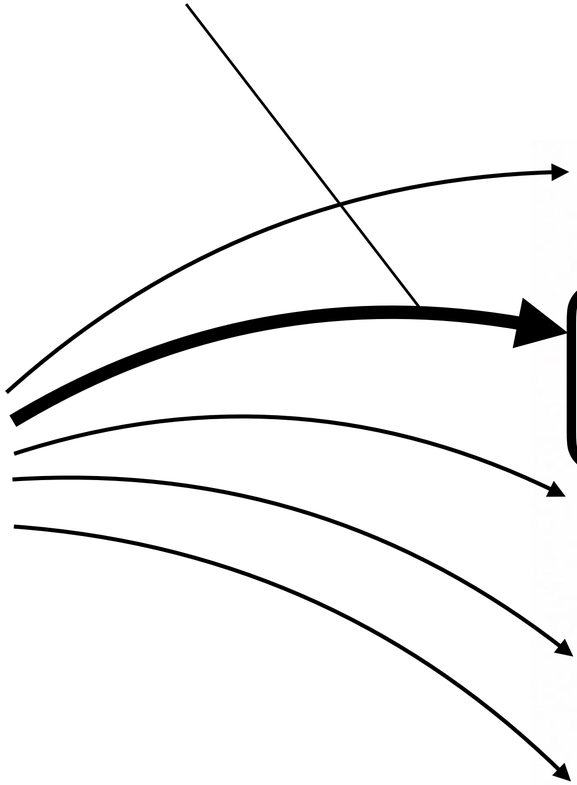
. . .

128. . . .

With fixed random seed 99 we may always get this text

Plausible texts

The weather today is cold and **overcast** / **grey**. A **light** / **gentle** breeze is **moving** / **blowing** through the trees, and the streets seem **quiet** / **still**. I think I'll stay **home** / **inside** and read a **book** / **novel** with a **cup** / **mug** of tea.

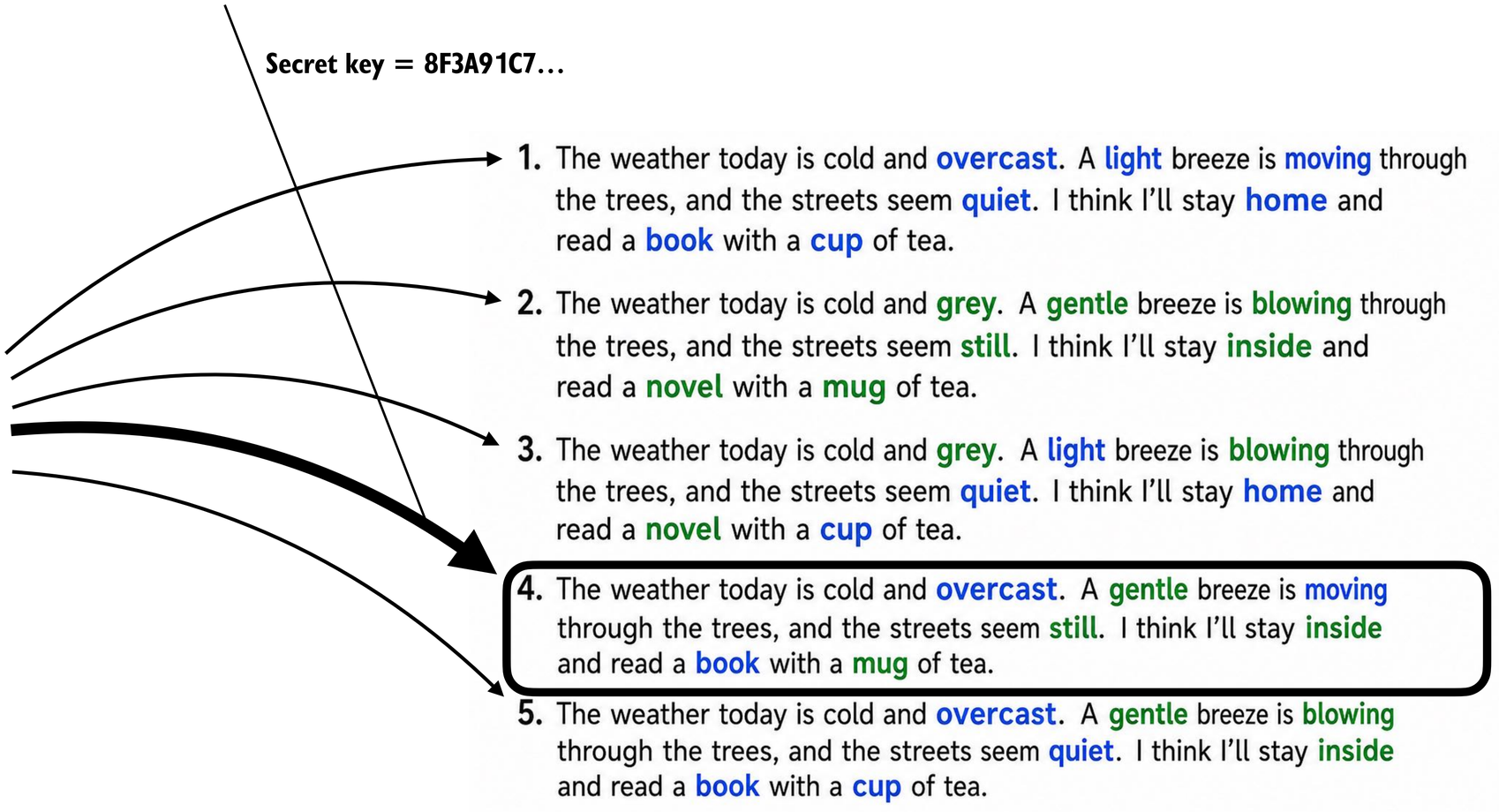


1. The weather today is cold and **overcast**. A **light** breeze is **moving** through the trees, and the streets seem **quiet**. I think I'll stay **home** and read a **book** with a **cup** of tea.
2. The weather today is cold and **grey**. A **gentle** breeze is **blowing** through the trees, and the streets seem **still**. I think I'll stay **inside** and read a **novel** with a **mug** of tea.
3. The weather today is cold and **grey**. A **light** breeze is **blowing** through the trees, and the streets seem **quiet**. I think I'll stay **home** and read a **novel** with a **cup** of tea.
4. The weather today is cold and **overcast**. A **gentle** breeze is **moving** through the trees, and the streets seem **still**. I think I'll stay **inside** and read a **book** with a **mug** of tea.
5. The weather today is cold and **overcast**. A **gentle** breeze is **blowing** through the trees, and the streets seem **quiet**. I think I'll stay **inside** and read a **book** with a **cup** of tea.

. . .

128. . . .

With watermarking we might get this text

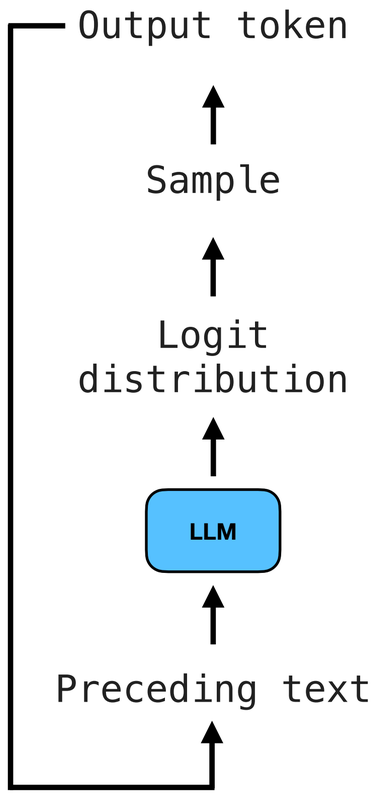


Plausible texts

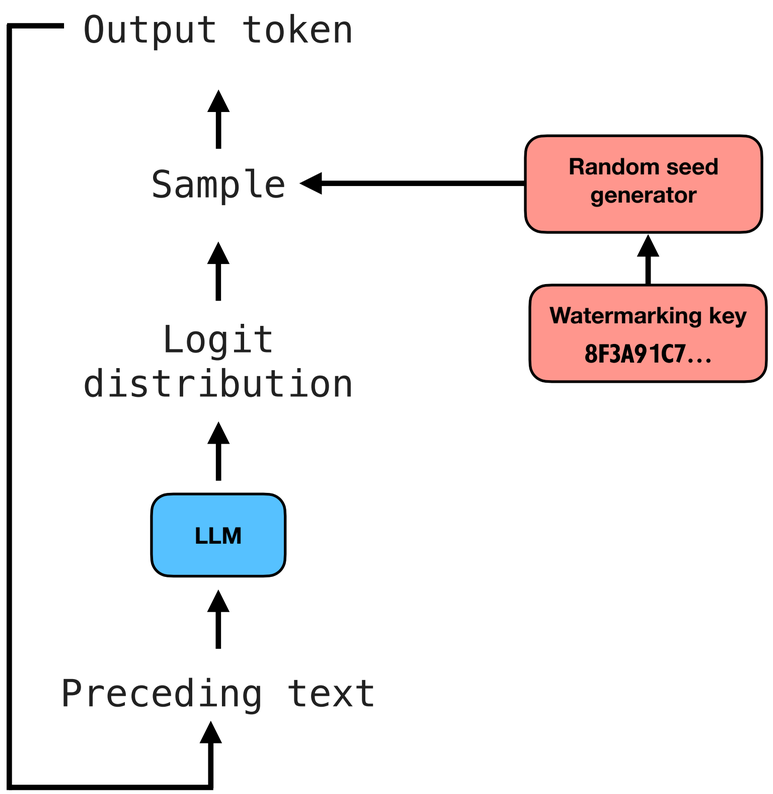
The weather today is cold and overcast / grey. A light / gentle breeze is moving / blowing through the trees, and the streets seem quiet / still. I think I'll stay home / inside and read a book / novel with a cup / mug of tea.

Summary

No watermarking



With watermarking

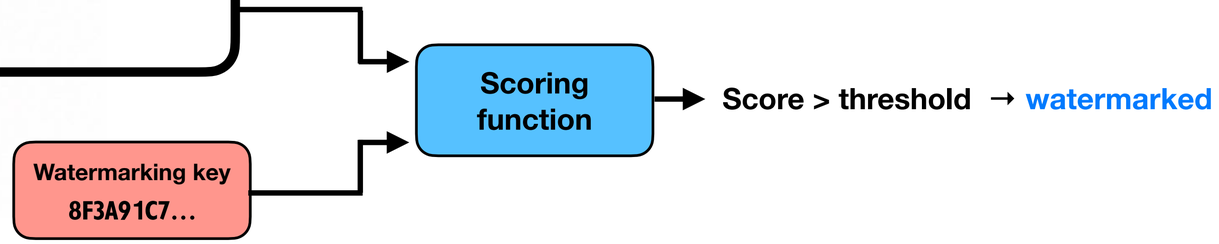


Detect watermark: Only possible if access to key

1. The weather today is cold and **overcast**. A **light** breeze is **moving** through the trees, and the streets seem **quiet**. I think I'll stay **home** and read a **book** with a **cup** of tea.
2. The weather today is cold and **grey**. A **gentle** breeze is **blowing** through the trees, and the streets seem **still**. I think I'll stay **inside** and read a **novel** with a **mug** of tea.
3. The weather today is cold and **grey**. A **light** breeze is **blowing** through the trees, and the streets seem **quiet**. I think I'll stay **home** and read a **novel** with a **cup** of tea.
4. The weather today is cold and **overcast**. A **gentle** breeze is **moving** through the trees, and the streets seem **still**. I think I'll stay **inside** and read a **book** with a **mug** of tea.
5. The weather today is cold and **overcast**. A **gentle** breeze is **blowing** through the trees, and the streets seem **quiet**. I think I'll stay **inside** and read a **book** with a **cup** of tea.

. . .

128. . . .

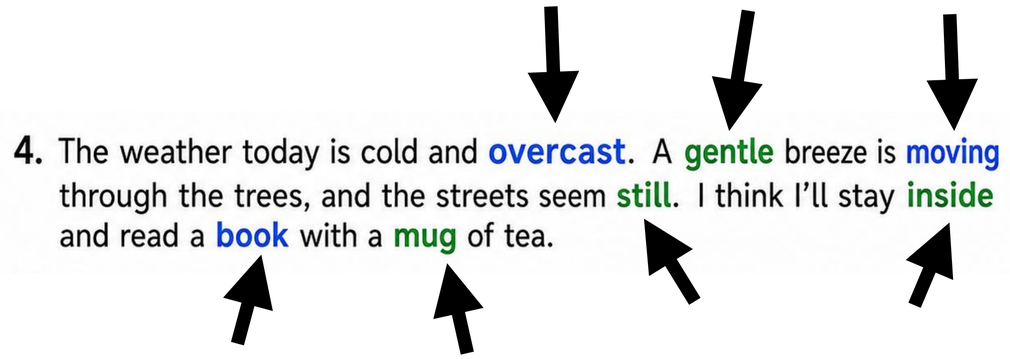


Remove watermark by editing words

Plausible texts

The weather today is cold and **overcast** / **grey**. A **light** / **gentle** breeze is **moving** / **blowing** through the trees, and the streets seem **quiet** / **still**. I think I'll stay **home** / **inside** and read a **book** / **novel** with a **cup** / **mug** of tea.

ideally, edit all watermark positions:

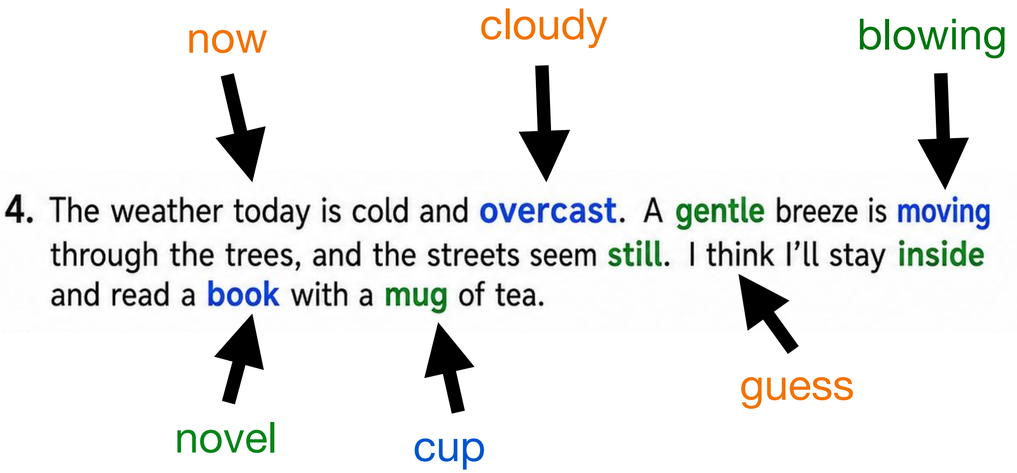


Remove watermark by editing words

Plausible texts

The weather today is cold and **overcast** / **grey**. A **light** / **gentle** breeze is **moving** / **blowing** through the trees, and the streets seem **quiet** / **still**. I think I'll stay **home** / **inside** and read a **book** / **novel** with a **cup** / **mug** of tea.

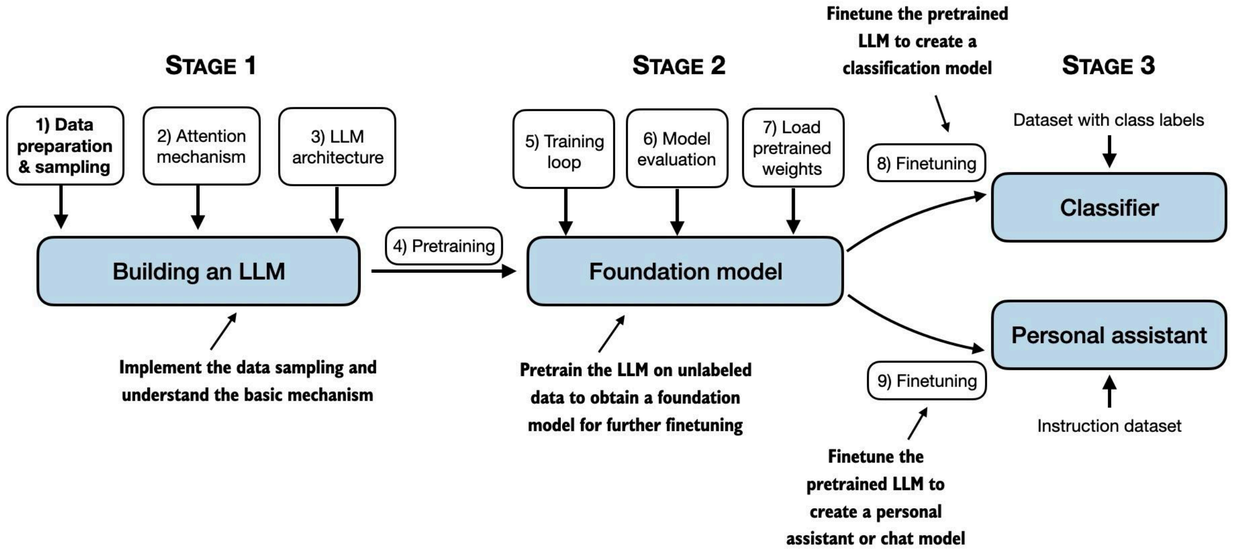
In practice, edit some random positions:



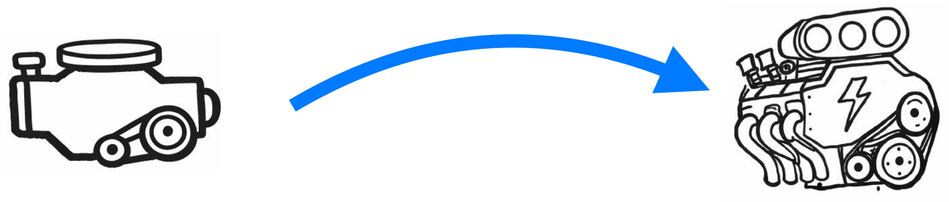
Takeaway: Conventional LLMs generate one word at a time

A “from-scratch” understanding helps us understand other concepts more easily

https://sebastianraschka.com/books/



From conventional LLMs to reasoning models

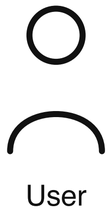


Conventional LLM

Reasoning LLM
(more expensive to run)

Conventional LLMs generate one word at a time

Reasoning models are LLMs that generate “more explanation” (aka reasoning trace)



User

A hiker walks 12 miles north,
then 5 miles east,
then 12 miles south.

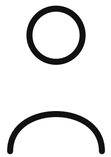
How far is the hiker
from the starting point?

Conventional LLM

The hiker is 5 miles from the starting point.

They walk 12 miles north and then 12 miles south,
which cancel out. The 5 miles east remains, so
they are 5 miles from the starting point.

Reasoning models are “chattier” LLMs



User

A hiker walks 12 miles north,
then 5 miles east,
then 12 miles south.

How far is the hiker
from the starting point?

Conventional LLM

The hiker is 5 miles from the starting point.

They walk 12 miles north and then 12 miles south, which cancel out. The 5 miles east remains, so they are 5 miles from the starting point.

Reasoning LLM

(self-correction & backtracking)

Let me think this through. The hiker walks 12 miles north, then 5 miles east, then 12 miles south.

At first glance, you might add all the distances and get 29 miles, but that would be the total distance traveled, not the distance from the starting point.

Initial idea

Self-correction

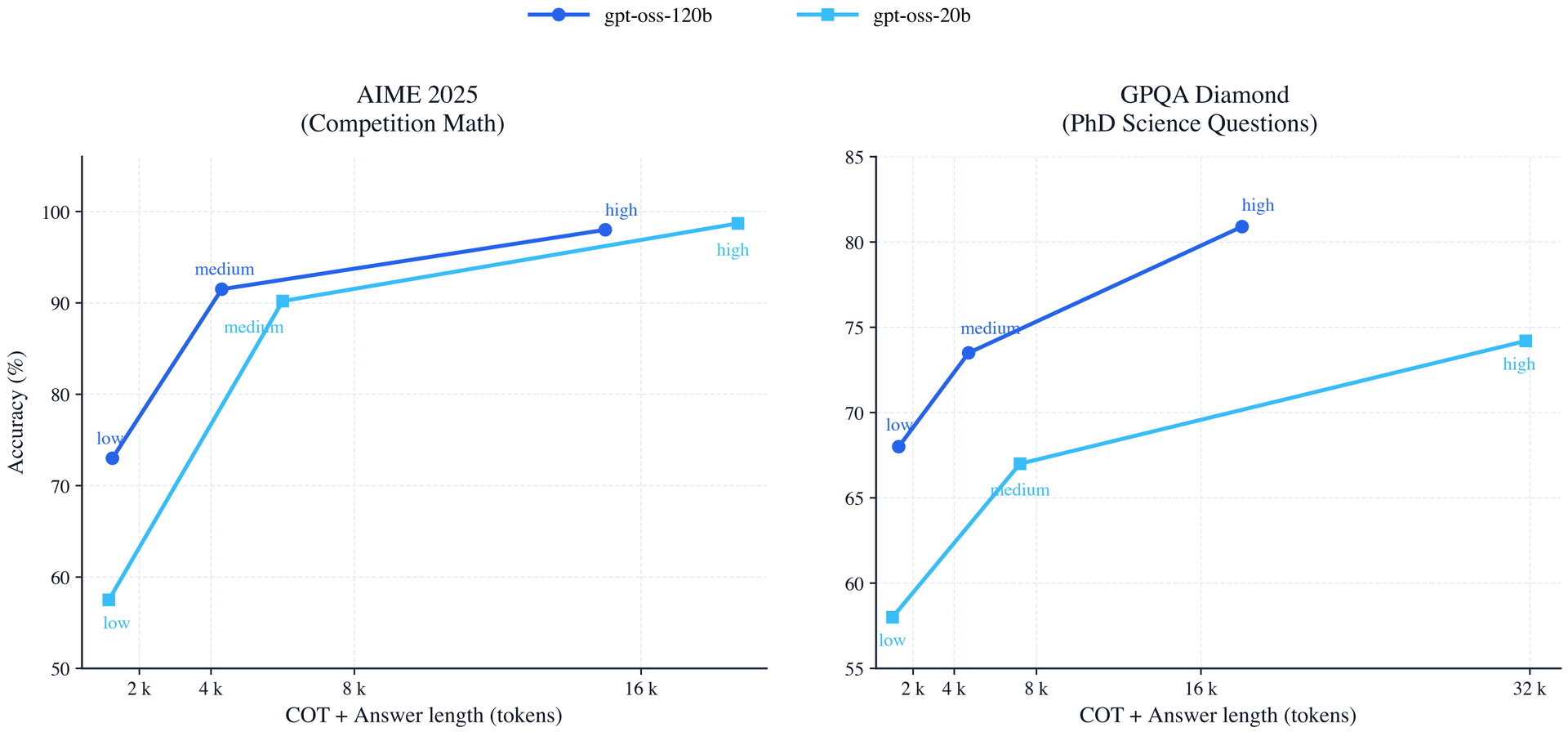
So let me back up and look at the net movement instead. The 12 miles north and 12 miles south cancel each other out, which means only the 5 miles east remains.

Backtracking

Therefore, the hiker is 5 miles from the starting point.

Final verified answer

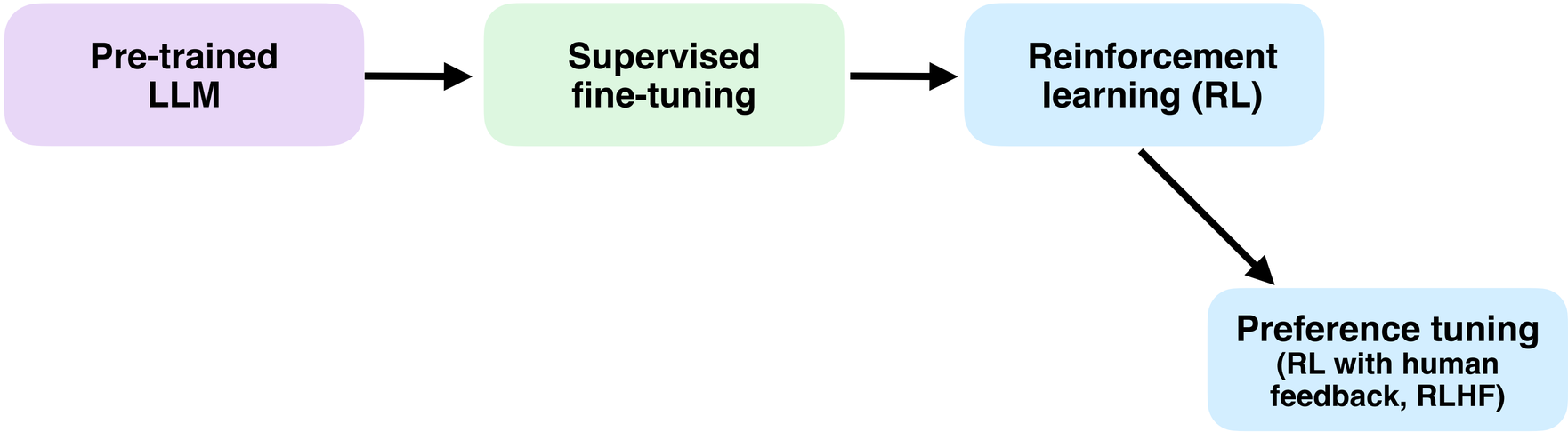
Answer length correlates with accuracy



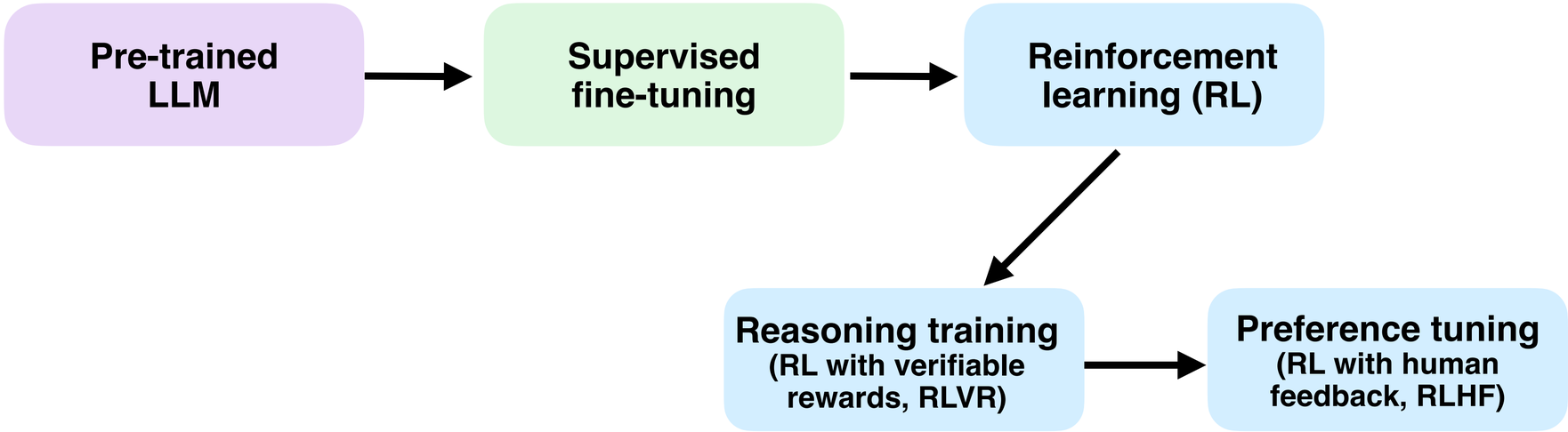
From “gpt-oss-120b & gpt-oss-20b Model Card”, https://arxiv.org/abs/2508.10925

How are reasoning models trained?

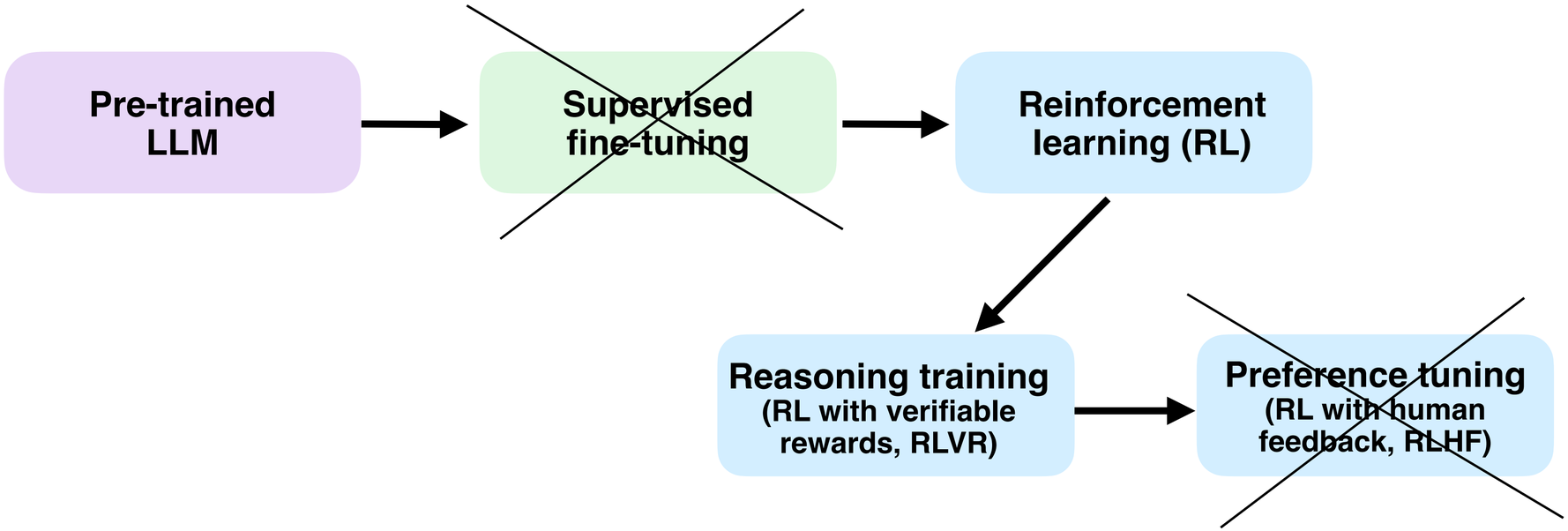
”Regular” LLM training



”Reasoning” LLM training



”Reasoning” LLM training



As proof of concept, even “just” RLVR works

What the rewards in RLVR look like when training reasoning models

Prompt

```
Solve: 17 * 24 = ?
```

Output example 1

```
<think>
Break 24 into 20 + 4.
17 * 20 = 340.
17 * 4 = 68.
340 + 68 = 408.
</think>

<answer>408</answer>
```

Accuracy reward = 1
(final answer is correct)

Format reward = 1
(tags are present)

What the rewards in RLVR look like when training reasoning models

Prompt

```
Solve: 17 * 24 = ?
```

Output example 1

```
<think>
Break 24 into 20 + 4.
17 * 20 = 340.
17 * 4 = 68.
340 + 68 = 408.
</think>

<answer>408</answer>
```

Accuracy reward = 1
(final answer is correct)

Format reward = 1
(tags are present)

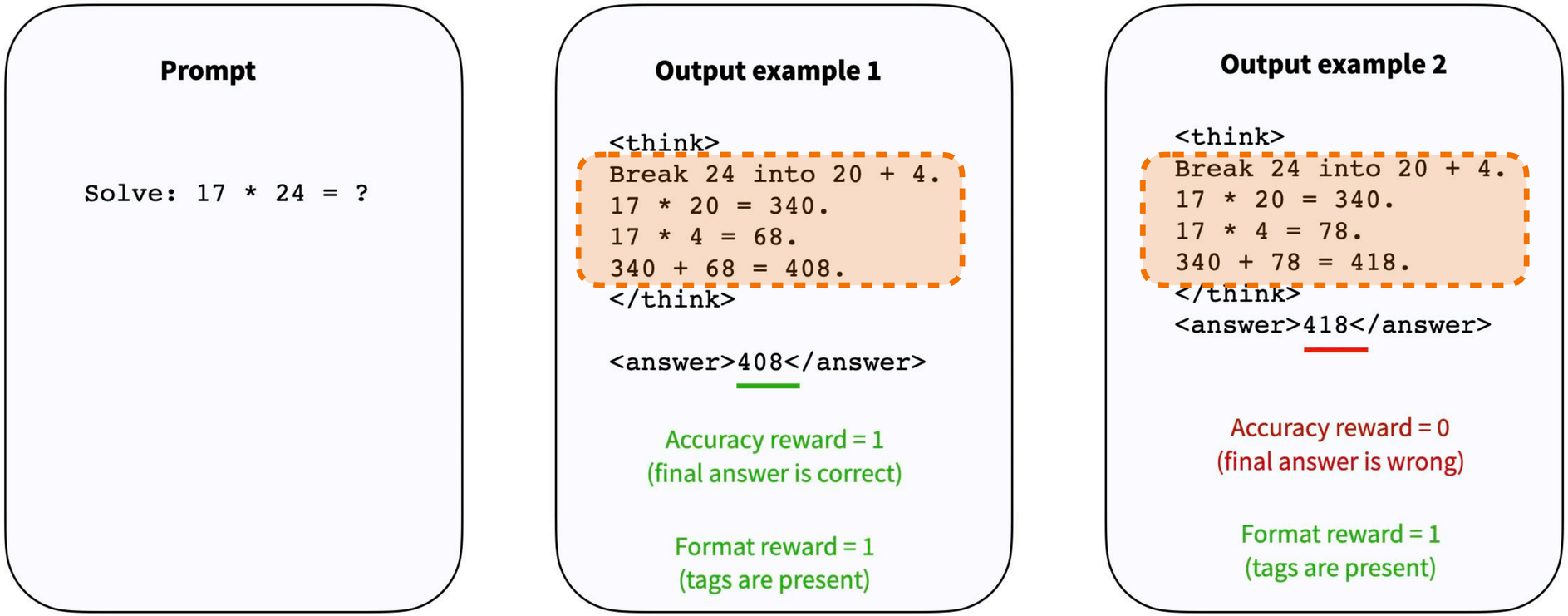
Output example 2

```
<think>
Break 24 into 20 + 4.
17 * 20 = 340.
17 * 4 = 78.
340 + 78 = 418.
</think>
<answer>418</answer>
```

Accuracy reward = 0
(final answer is wrong)

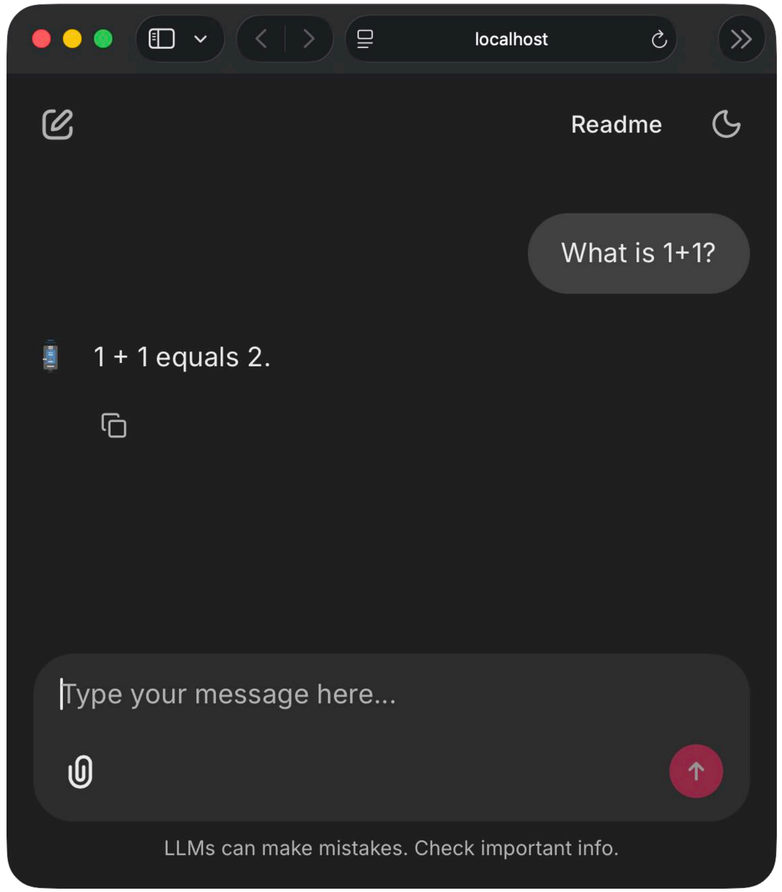
Format reward = 1
(tags are present)

What is ignored in RLVR



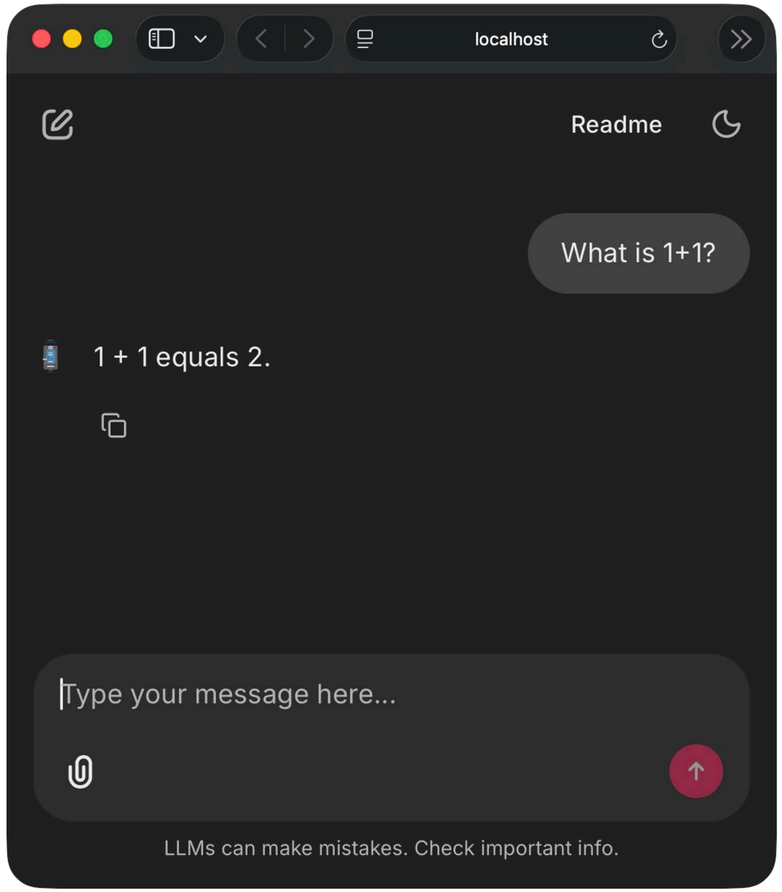
Reasoning may exist as binary (on/off) switch

`thinking=False`

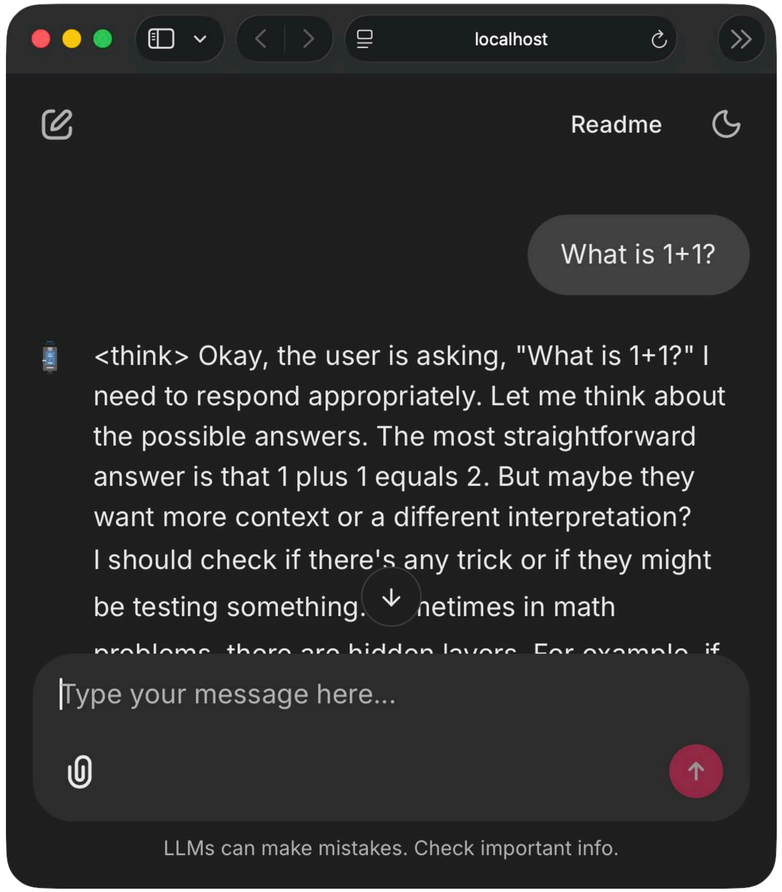


Reasoning may exist as binary (on/off) switch

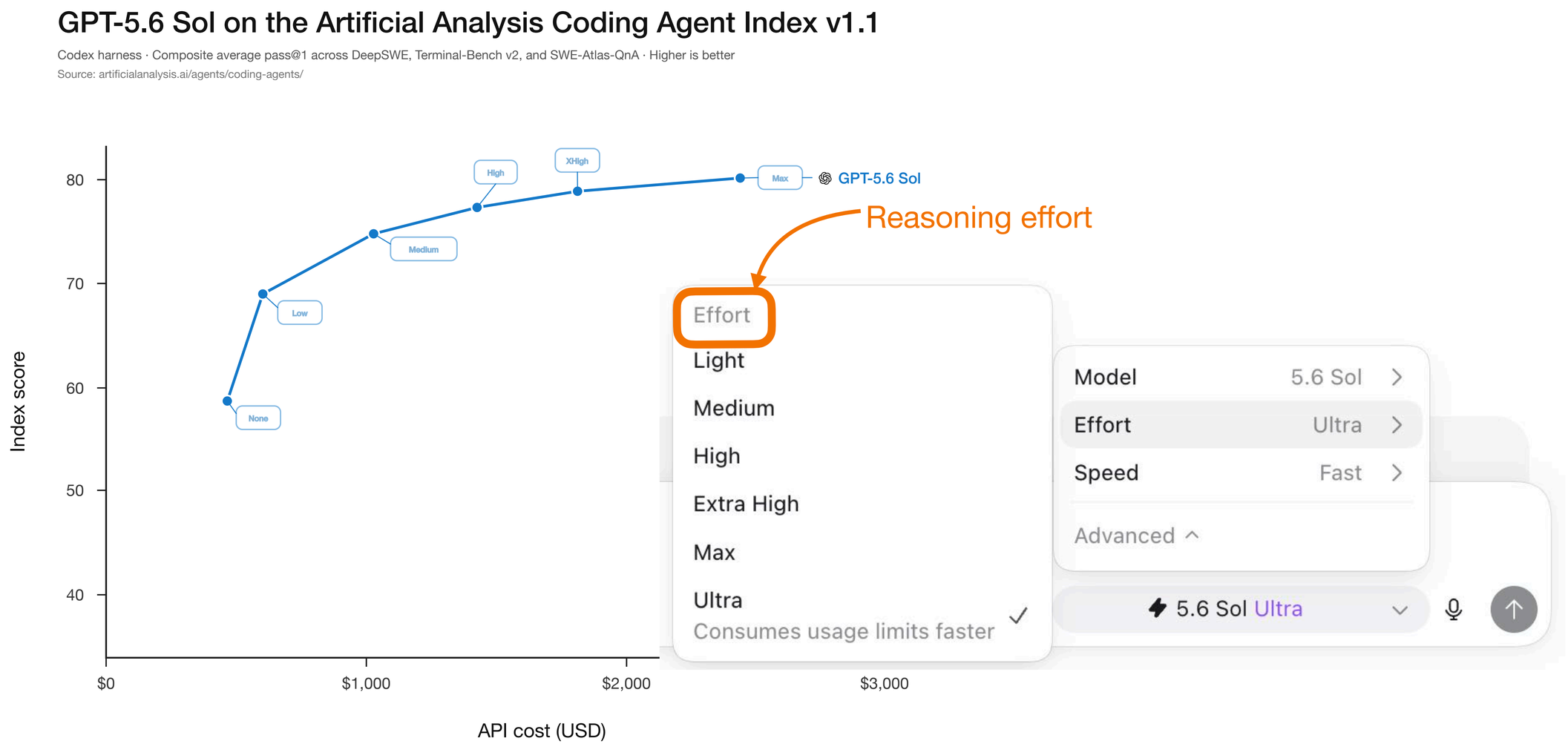
`thinking=False`



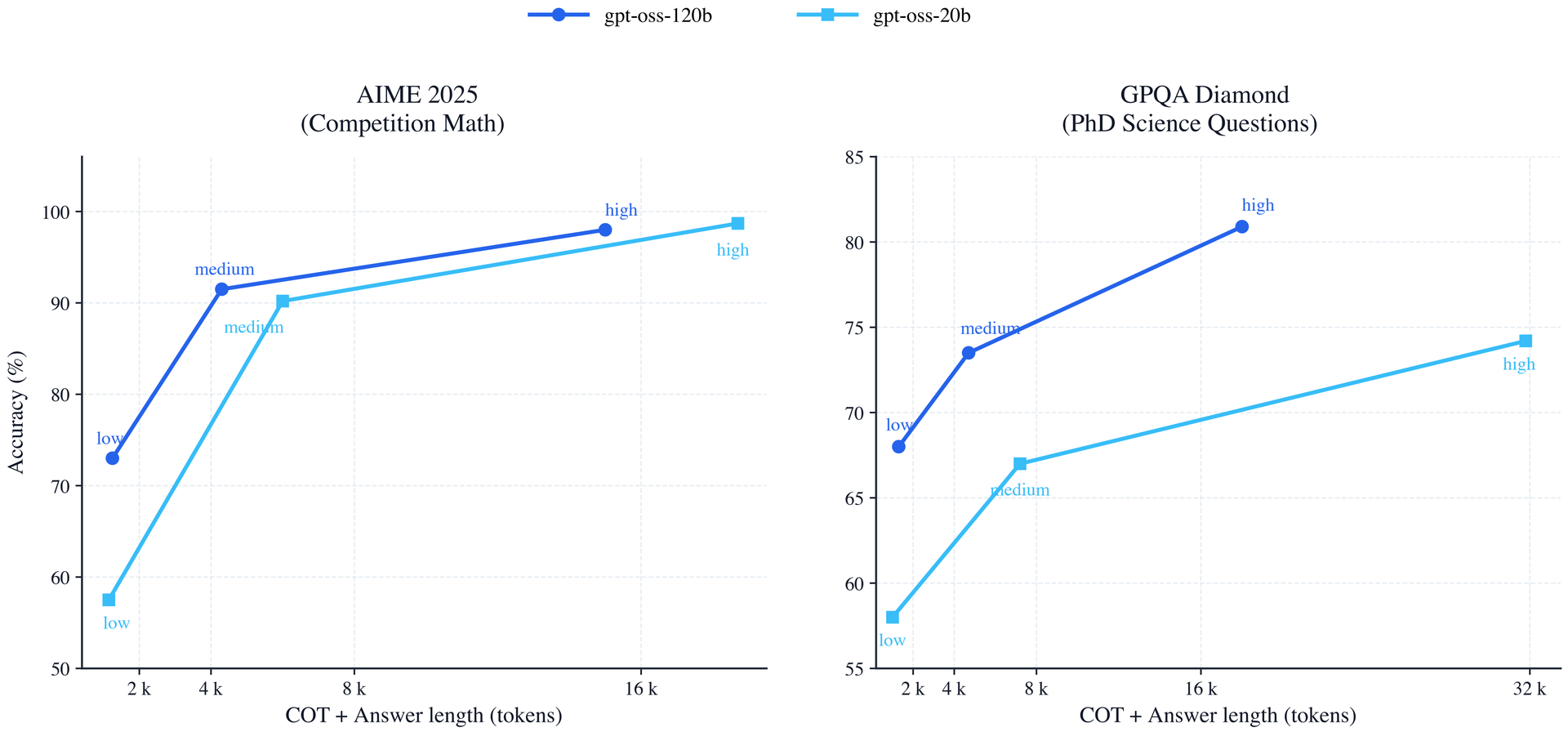
`thinking=True`



Reasoning can be implemented with multiple effort levels

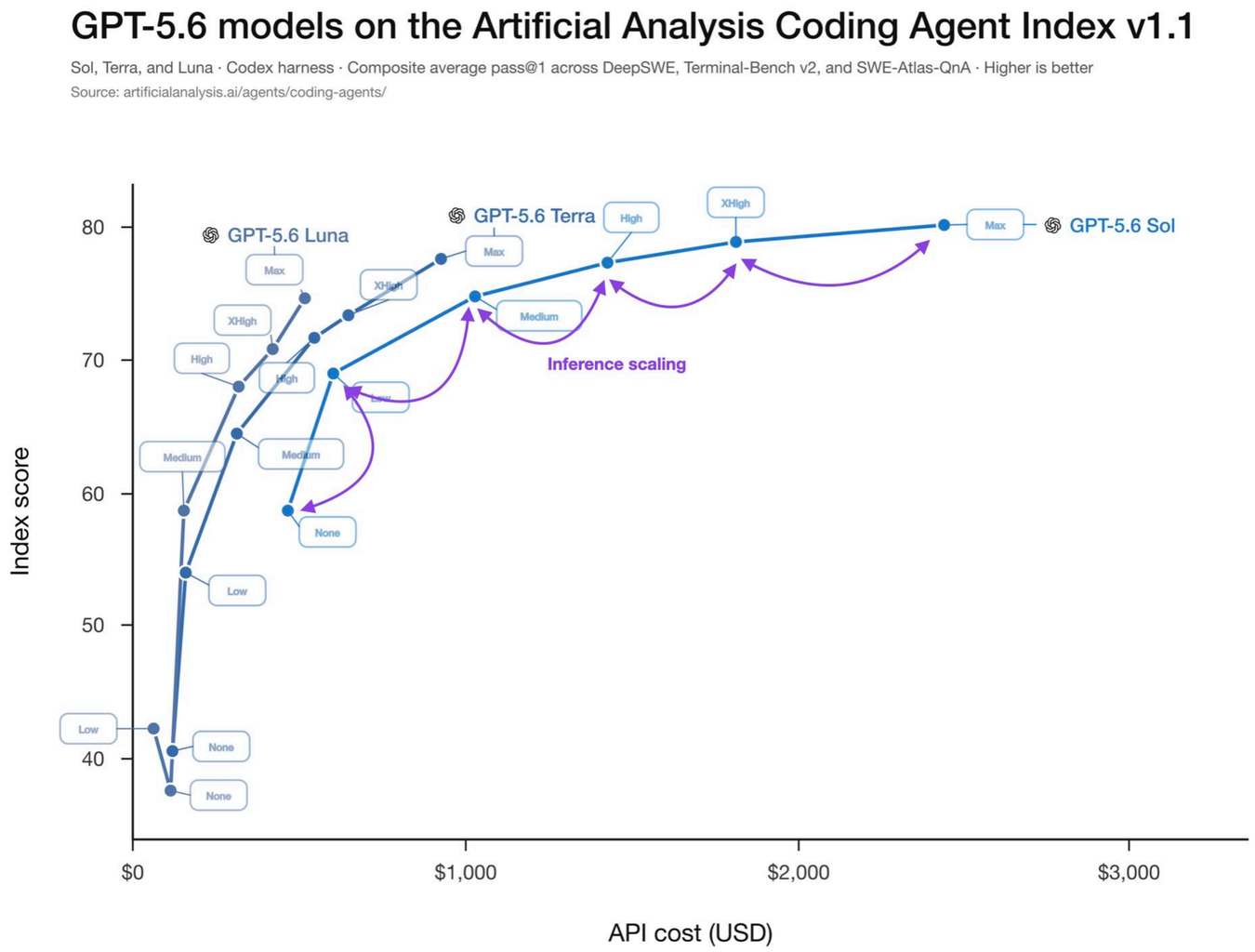


More reasoning effort results in higher token usage

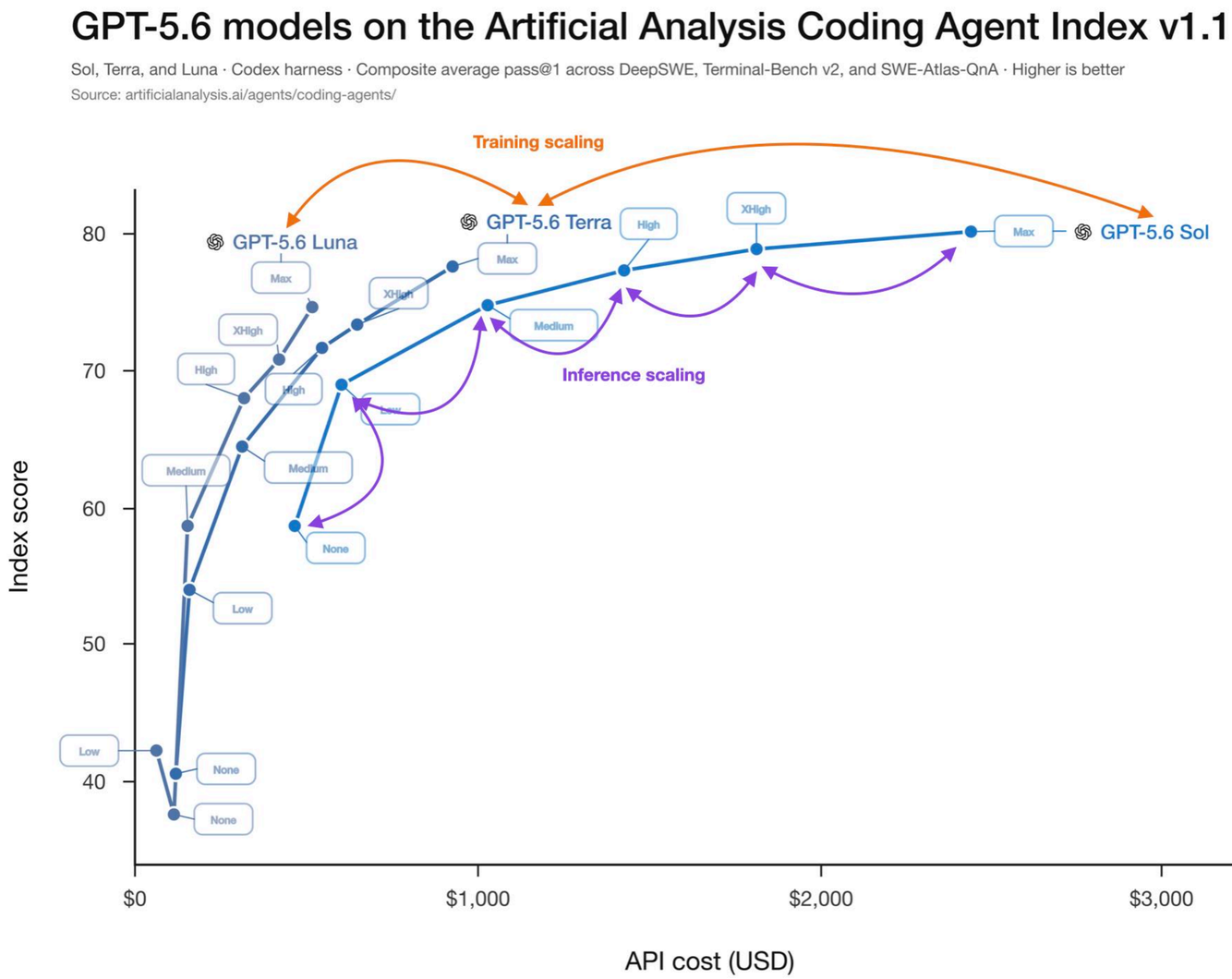


From “gpt-oss-120b & gpt-oss-20b Model Card”, https://arxiv.org/abs/2508.10925

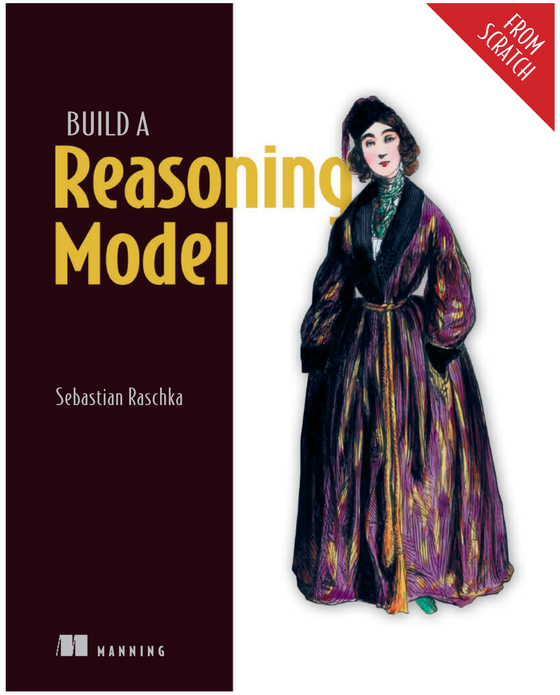
Reasoning effort is a form of inference scaling



Scaling training still matters though

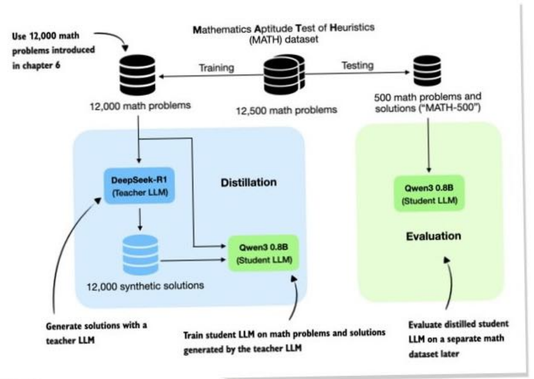


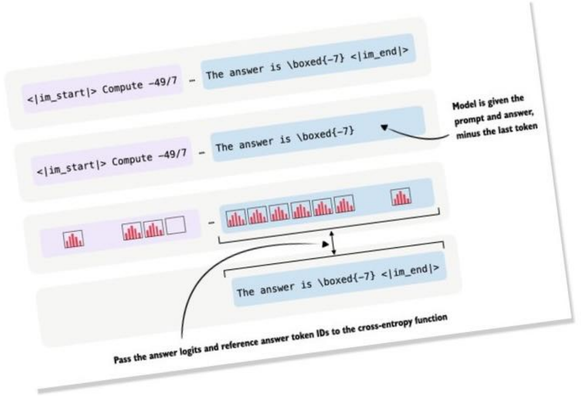
Build A Reasoning Model
(From Scratch)

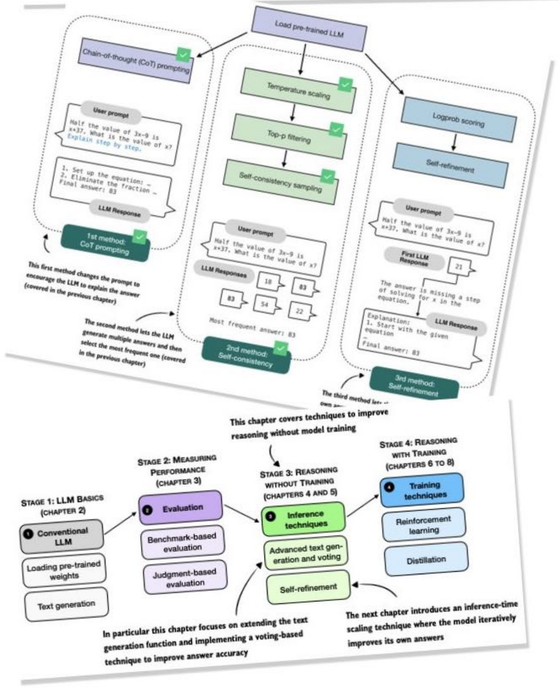


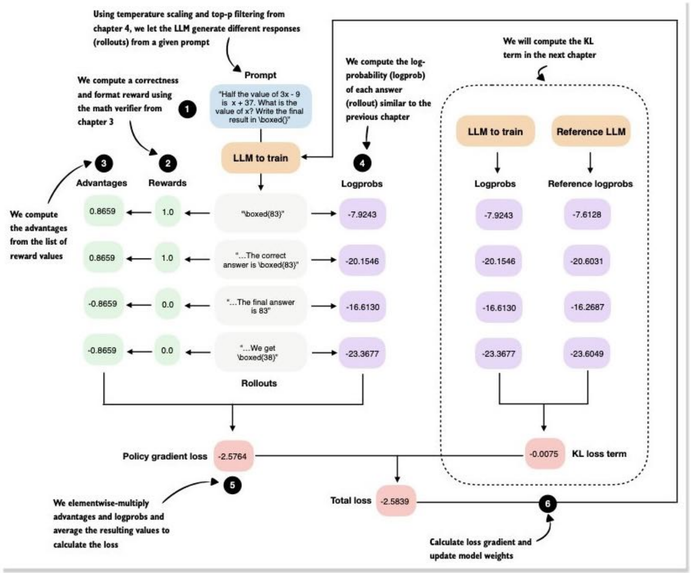
https://amzn.to/4aAKiFY

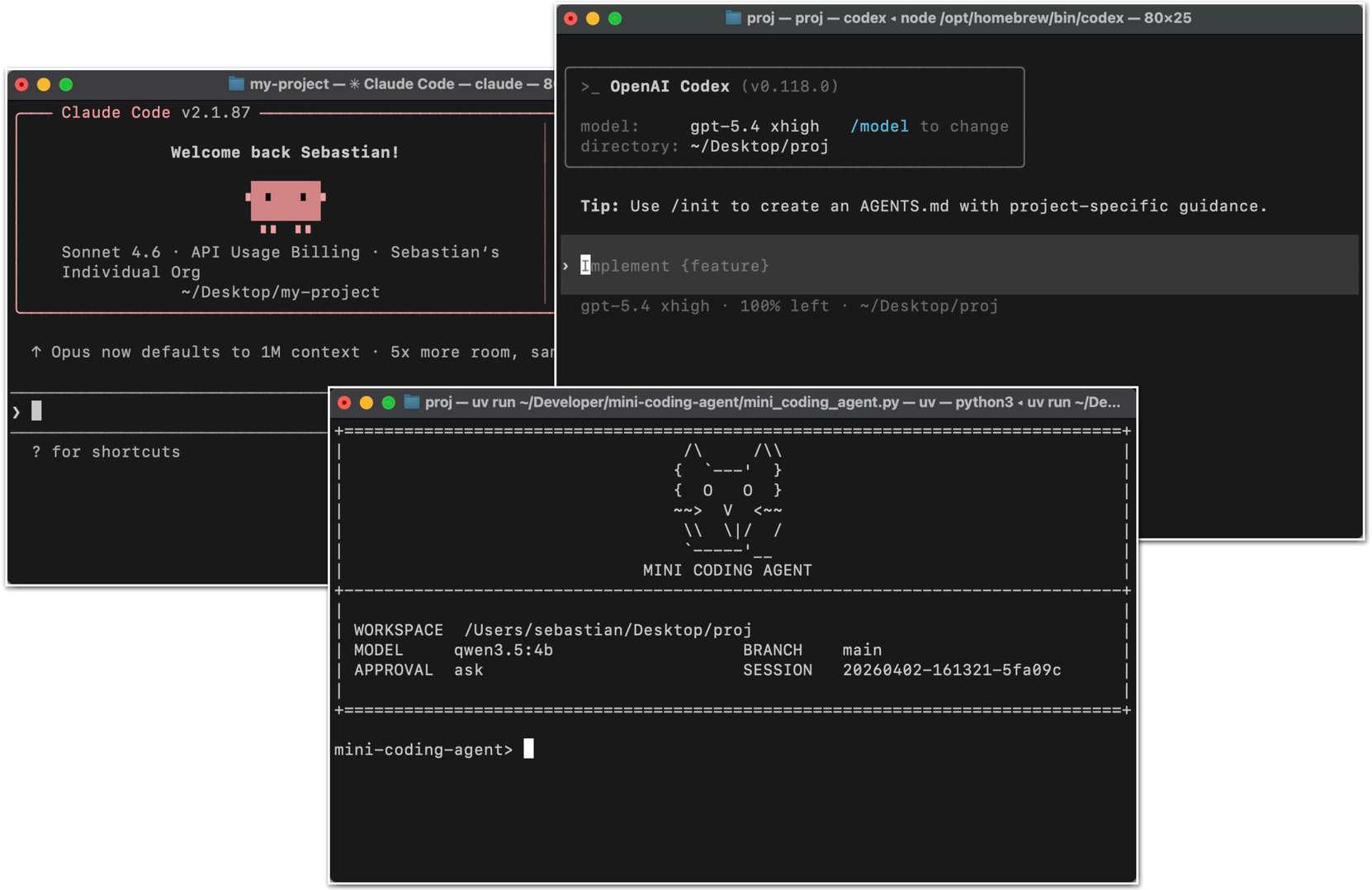
https://mng.bz/Nwr7



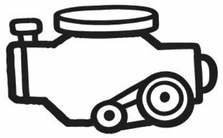




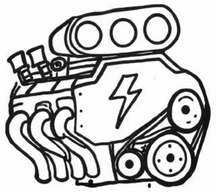



my-project — ✳ Claude Code — claude — 8
Claude Code v2.1.87
Welcome back Sebastian!
Sonnet 4.6 · API Usage Billing · Sebastian's Individual Org
~/Desktop/my-project
↑ Opus now defaults to 1M context · 5x more room, sar
? for shortcuts
proj — proj — codex ◂ node /opt/homebrew/bin/codex — 80×25
>_ OpenAI Codex (v0.118.0)
model: gpt-5.4 xhigh /model to change
directory: ~/Desktop/proj
Tip: Use /init to create an AGENTS.md with project-specific guidance.
› Implement {feature}
gpt-5.4 xhigh · 100% left · ~/Desktop/proj
proj — uv run ~/Developer/mini-coding-agent/mini_coding_agent.py — uv — python3 ◂ uv run ~/De...
MINI CODING AGENT
WORKSPACE /Users/sebastian/Desktop/proj
MODEL qwen3.5:4b
BRANCH main
APPROVAL ask
SESSION 20260402-161321-5fa09c
mini-coding-agent>

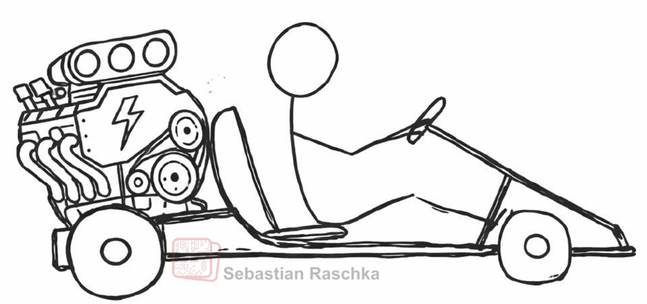
From Conventional LLMs to Reasoning Models to Agents



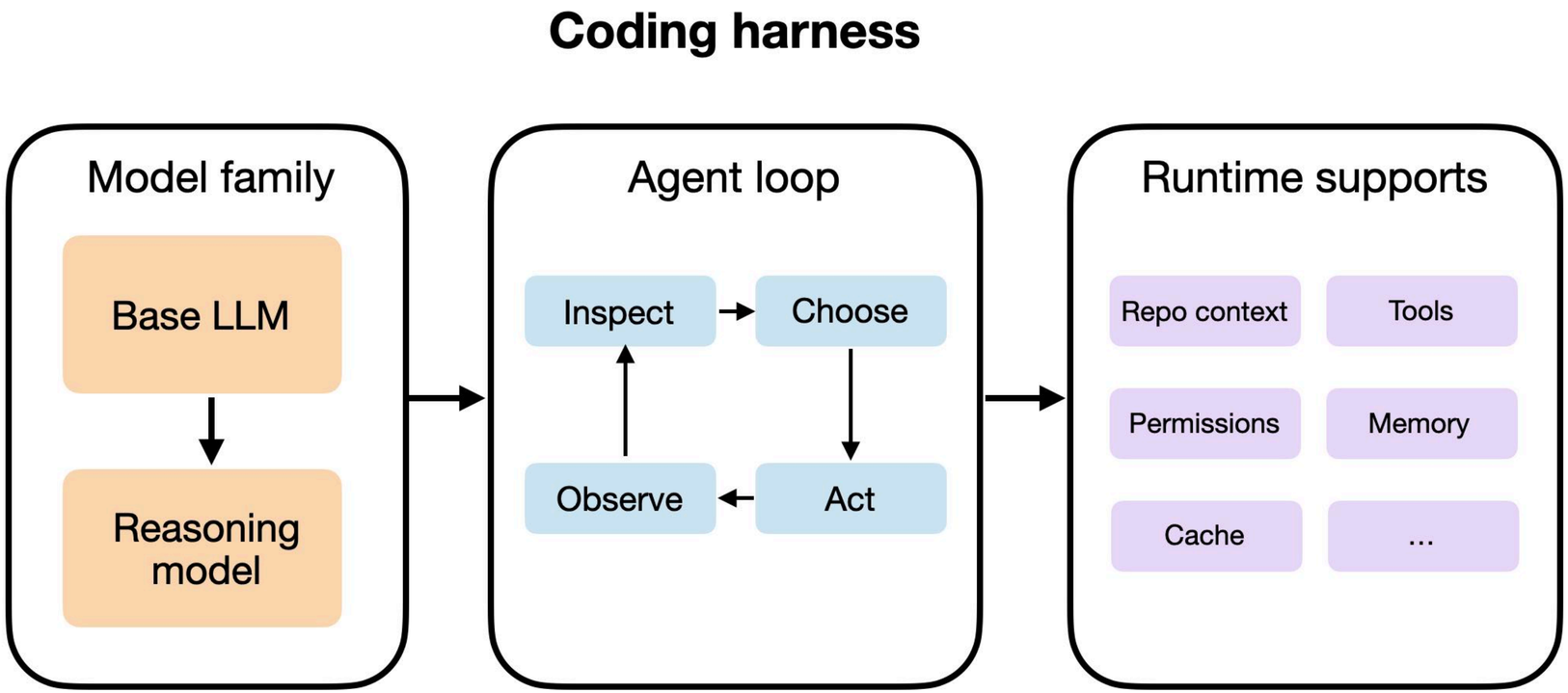
Conventional LLM



Reasoning LLM
(more expensive to run)



LLM in agent harness


Coding harness
Model family
Base LLM
Reasoning model
Agent loop
Inspect
Choose
Observe
Act
Runtime supports
Repo context
Tools
Permissions
Memory
Cache
...

AHEAD OF AI

Sebastian Raschka, PhD

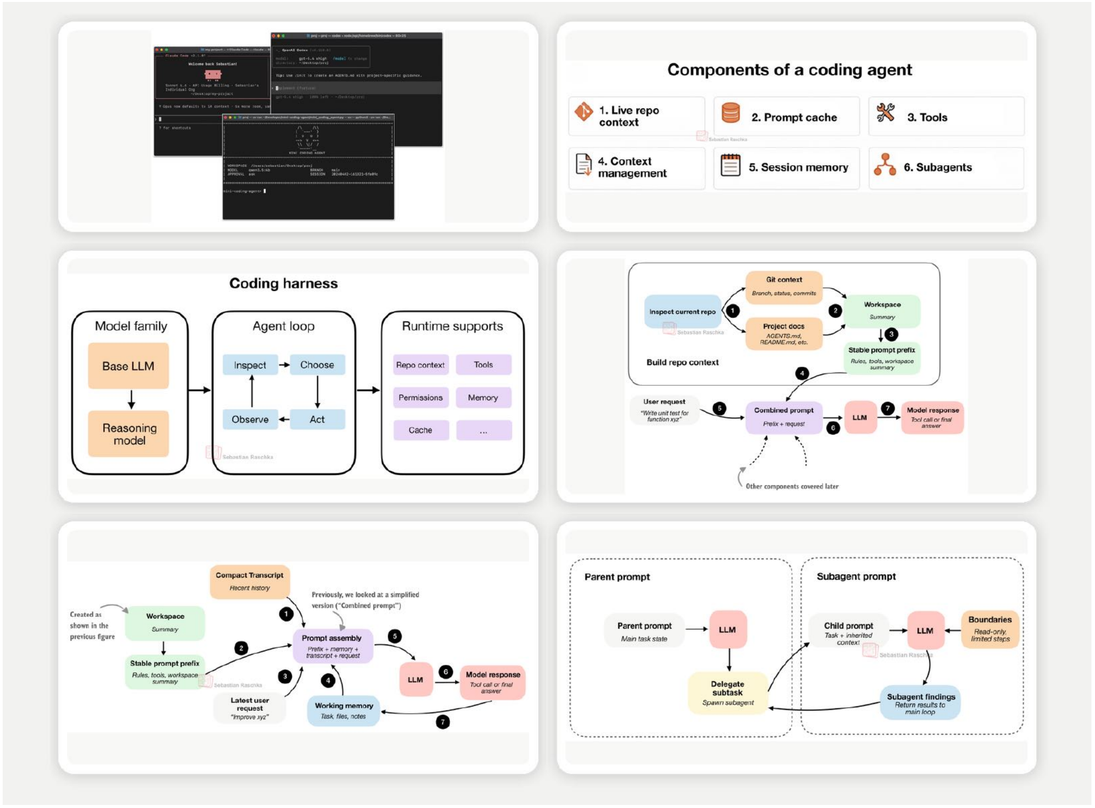
Components of A Coding Agent

How coding agents use tools, memory, and repo context to make LLMs work better in practice

SEBASTIAN RASCHKA, PHD

APR 04, 2026

951 63 97 Share



https://magazine.sebastianraschka.com/

or

https://aheadofai.com