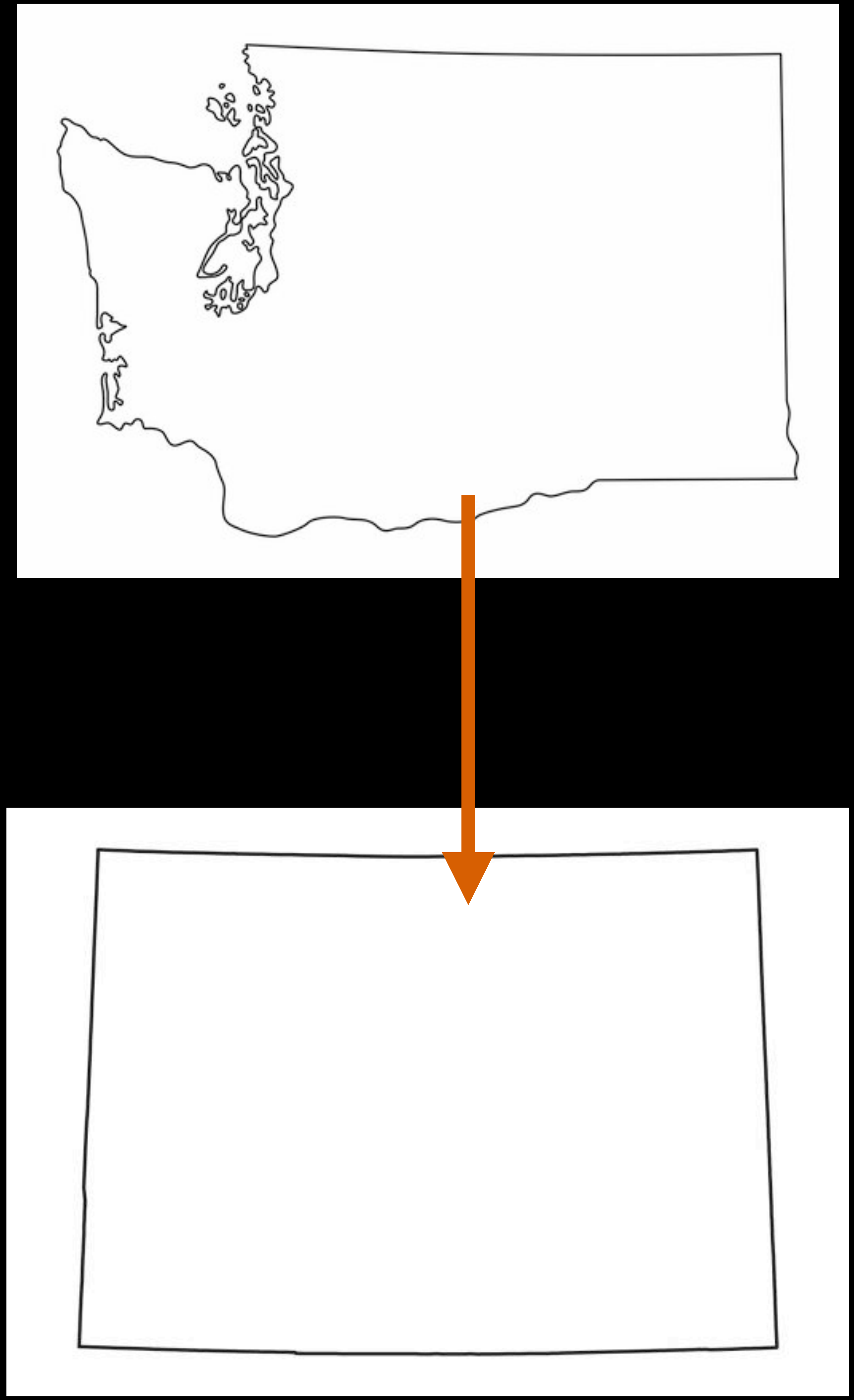
Tuning Foundation Models

CSCI 5922
Spring 2025

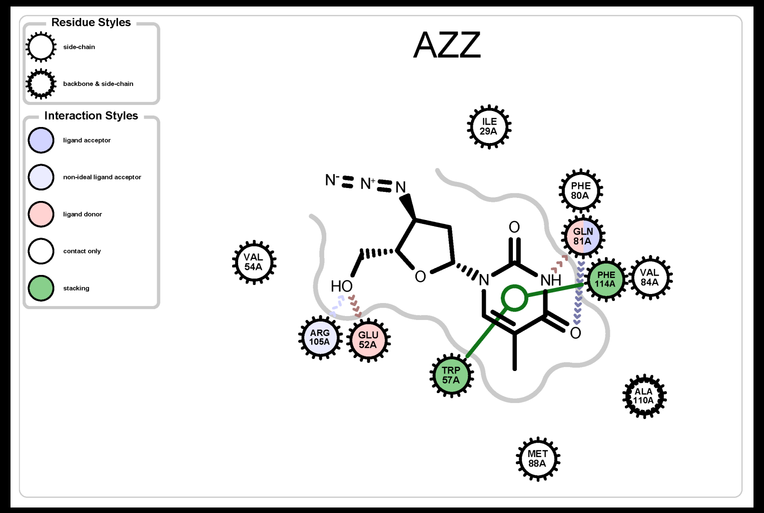
Josh Myers-Dean
3/18/2025

Who am I?



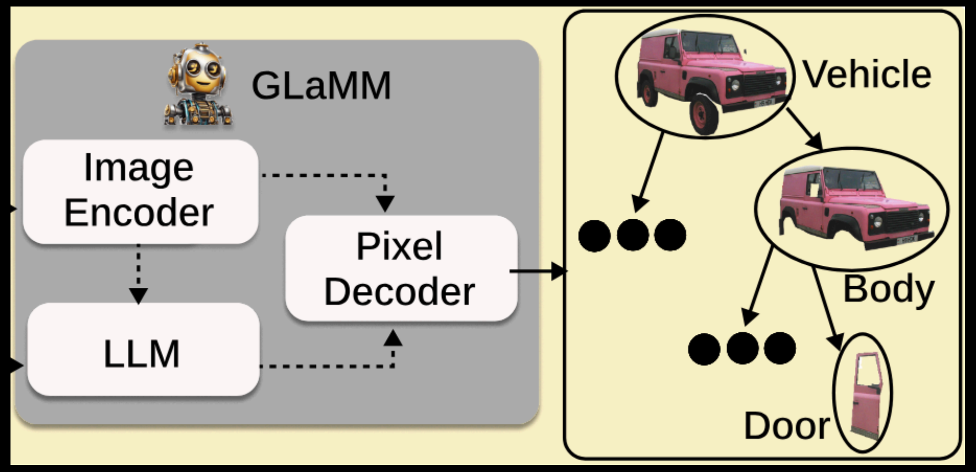
Who am I? — Research Experience

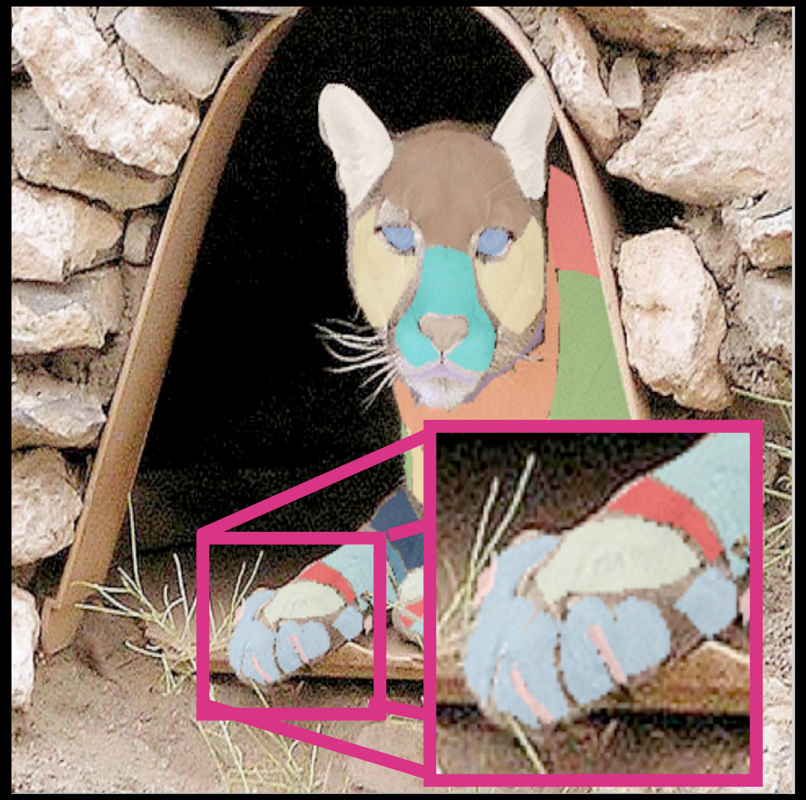
2018-2021



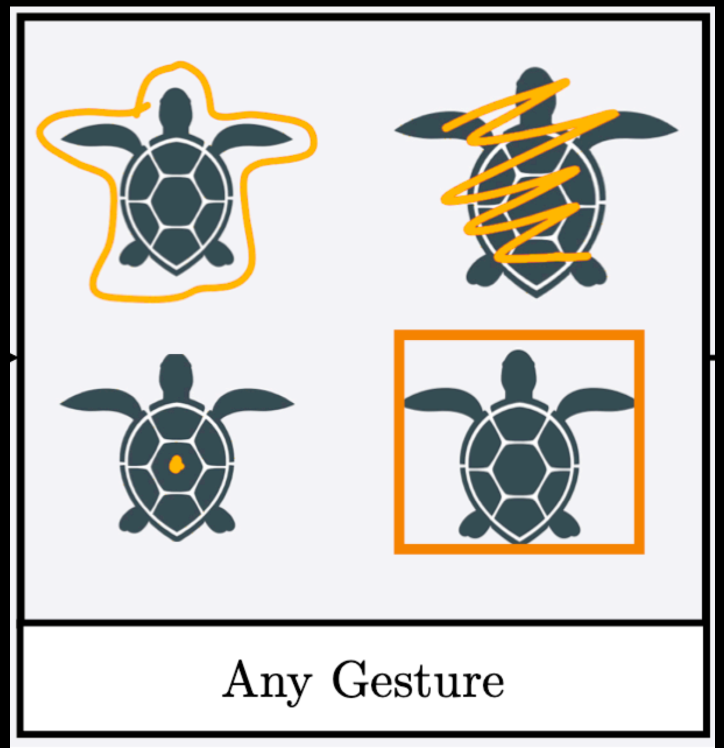
2020-2021

2021-Present

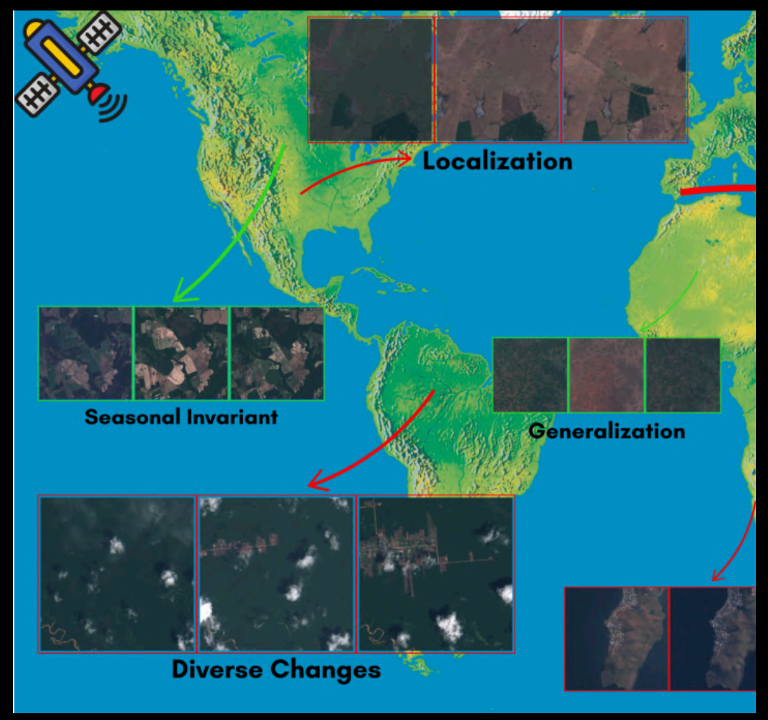




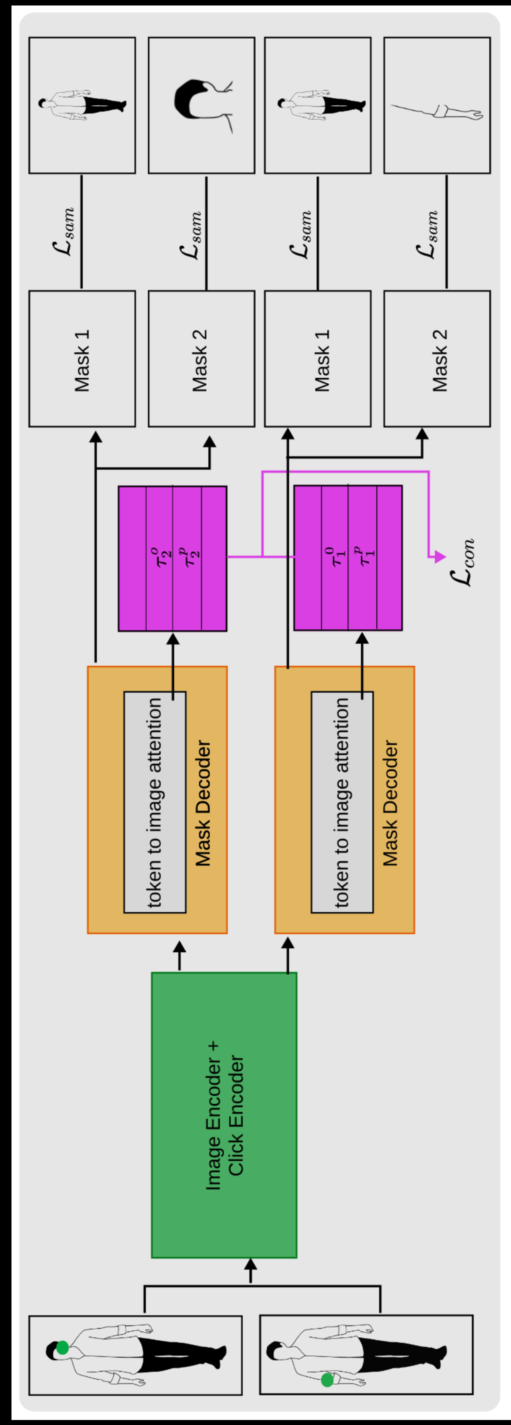
2022



2023



2024



Why Do We Need to Tune Foundation Models?

Tuning — Motivation

- Generalist vs. Specialist

BSc in CS Reqs.

- Computer Networking
- Data Structures
- Algorithms
- Software Engineering
- Computer Architecture

PhD in CS Reqs.
Focus: Machine Learning

- Bayesian Statistics
- NLP
- Reinforcement Learning
- Independent Study
- Dynamical Systems

BSc emphasizes **general** knowledge

PhD emphasizes **specific** knowledge

Tuning — Foundation Models

- Large scale foundation model training emphasizes **breadth** of knowledge
- Fine-tuning allows for **specialized** use cases

What are the best API's to use to leverage images?

When working with images through APIs, there are several robust options available depending on your needs, such as processing, analyzing, or generating images. Here are some of the top APIs you might consider:

1. **Google Cloud Vision API**: This API offers powerful image recognition capabilities, allowing you to understand the content of an image, including tagging, facial detection, and landmark recognition. It can also perform Optical Character Recognition (OCR).
2. **Amazon Rekognition**: Part of AWS, this API provides similar features to Google Cloud Vision but also includes the ability to analyze video content. It can detect objects, scenes, and faces in images and videos.
3. **Microsoft Azure Computer Vision API**: This service from Microsoft offers features like image tagging, face detection, OCR, and even generates smart thumbnails by analyzing the content of images to produce cropped and framed thumbnails focusing on important elements.
4. **IBM Watson Visual Recognition**: This API provides a suite of image recognition services that include classifying images into thousands of categories, detecting objects and faces, and training custom classifiers on visual content.

Base ChatGPT-4

What are the best API's to use to leverage images?

There are several APIs available to access and analyze remote sensing imagery. The best API depends on your use case, whether it's retrieving satellite images, processing geospatial data, or analyzing land cover and vegetation. Below are some of the top APIs for leveraging remote sensing images:

1. Google Earth Engine (GEE) API

- **Best for:** Large-scale geospatial analysis, machine learning, and cloud-based processing.
- **Features:**
  - Free access to a vast archive of satellite imagery (Landsat, Sentinel, MODIS, etc.).
  - Built-in tools for spectral analysis, classification, and time-series analysis.
  - Python and JavaScript support.
- **Access:** Google Earth Engine

2. Sentinel Hub API

- **Best for:** Efficient access to Sentinel-1, Sentinel-2, Landsat, MODIS, and commercial datasets.

Remote Sensing Fine-tuned GPT

Overview

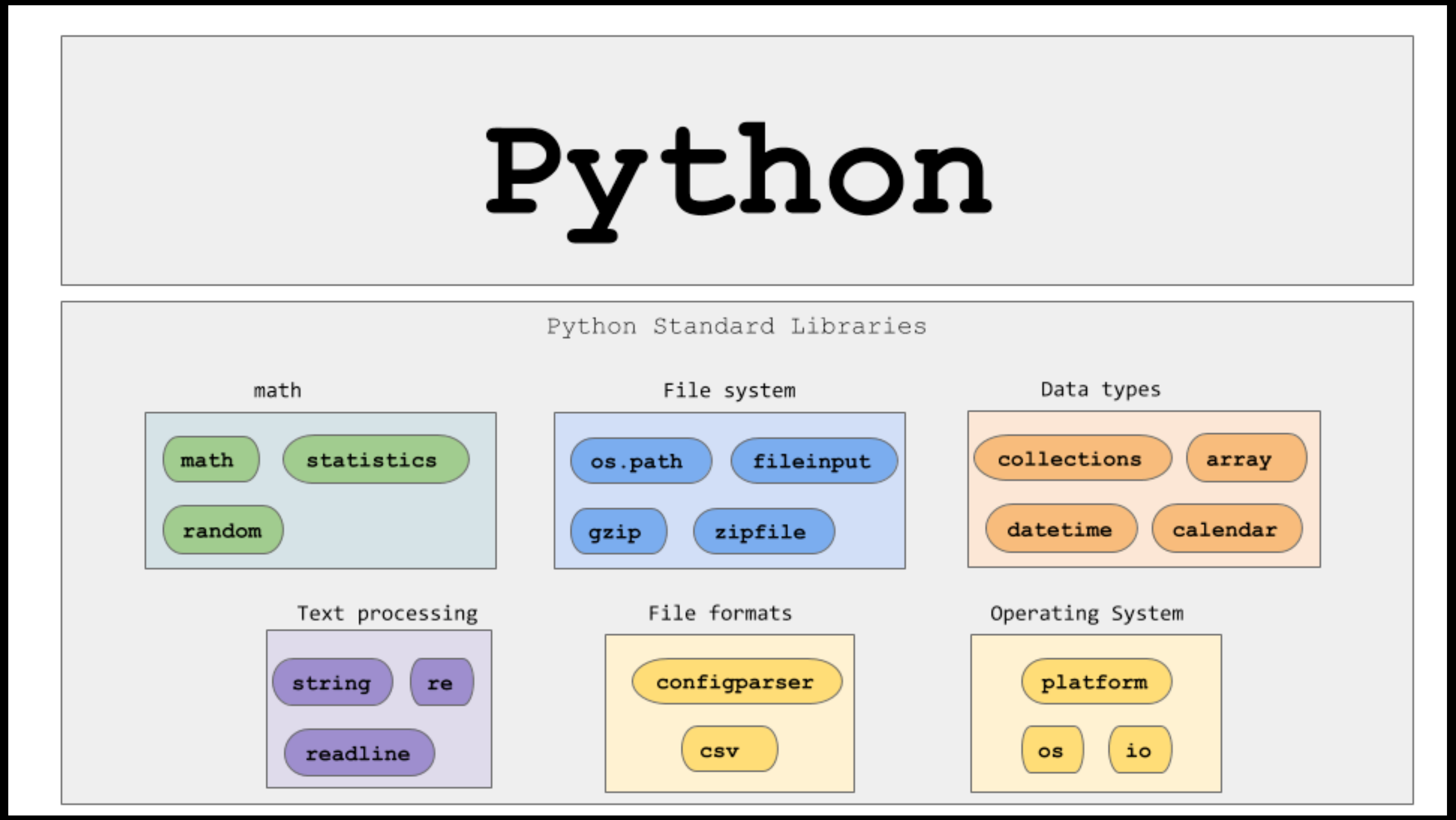
- Tool use
- Instruction tuning
- RLHF
- Parameter-efficient methods for fine-tuning (PEFT)
- Demo

Overview

- Tool use
- Instruction tuning
- RLHF
- Parameter-efficient methods for fine-tuning (PEFT)
- Demo

Teaching Foundation Models to Use Tools

- **Core idea:** Similar to how we use “tools”, let’s teach AI to do the same!



Recap of Prompting + ICL

- **Prompting:** Given an instruction, provide a response
- **In-context learning:** Given **k** demonstrations, provide a response that matches the structure of the demonstrations

Zero-Shot Prompt

```
Classify the sentiment of the following text as positive, negative, or neutral.
Text: I think the vacation was okay.
Sentiment:
```

AI Output

```
Neutral
```

Few-Shot Prompt

```
Classify the sentiment of the following text as positive, negative, or neutral.
Text: The product is terrible. Sentiment: Negative
Text: Super helpful, worth it Sentiment: Positive
Text: It doesnt work! Sentiment:
```

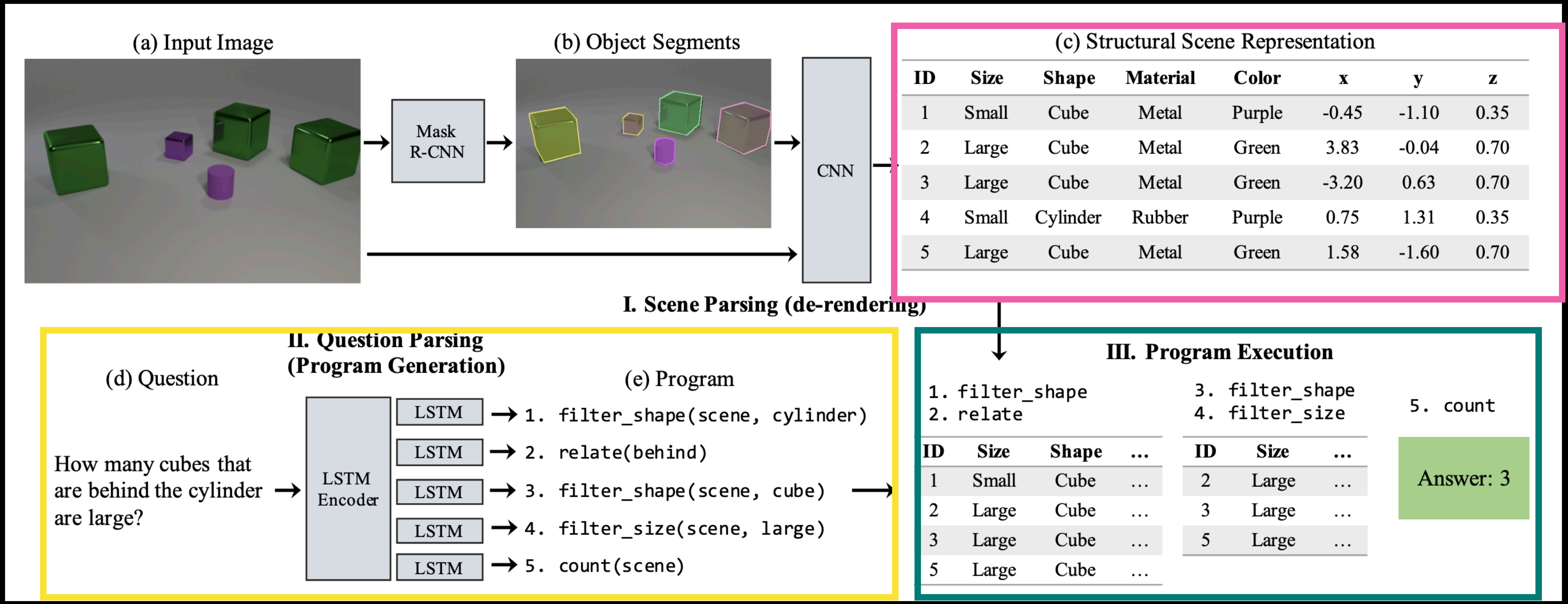
AI Output

```
Negative
```

https://learnprompting.org/docs/basics/few_shot

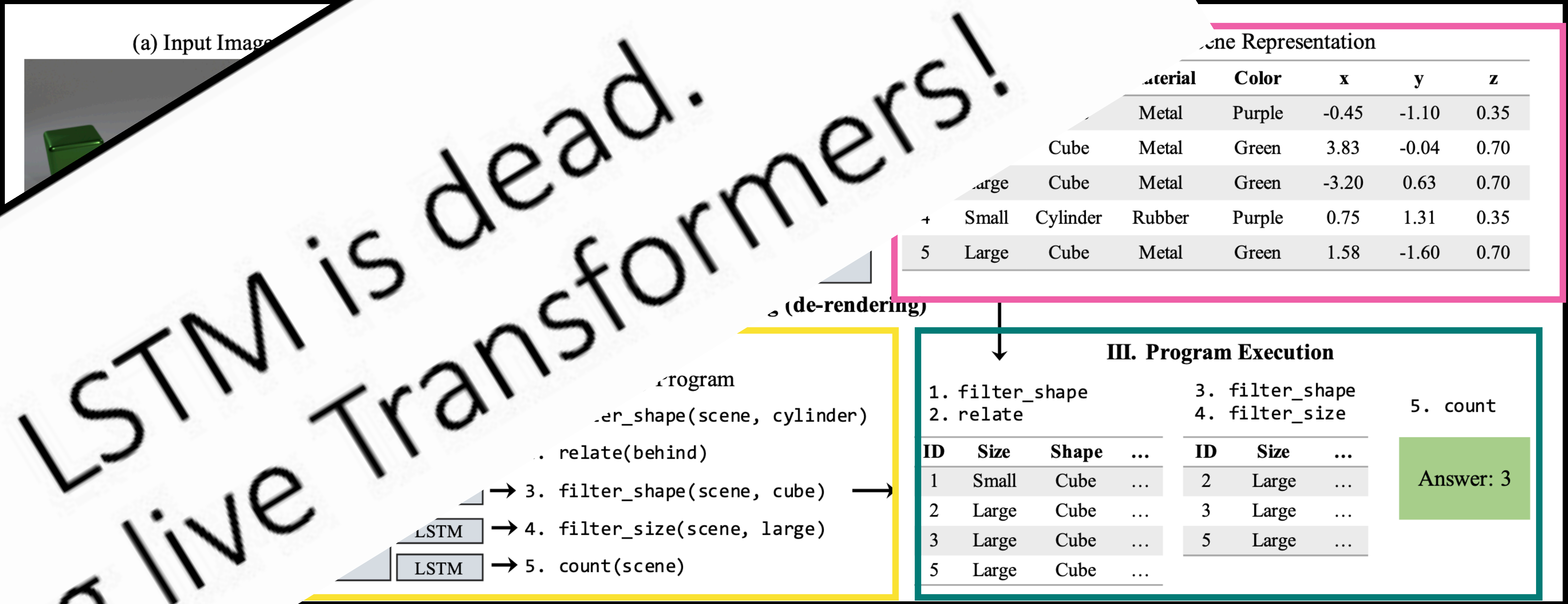
Teaching Foundation Models to Use Tools

- Rewind to pre-foundation model era: NS-VQA [Yi, NeurIPS18]
- **Key Idea:** Use NNs to extract attributes of a scene, generate programs & answer visual questions



Teaching Foundation Models t... Tools

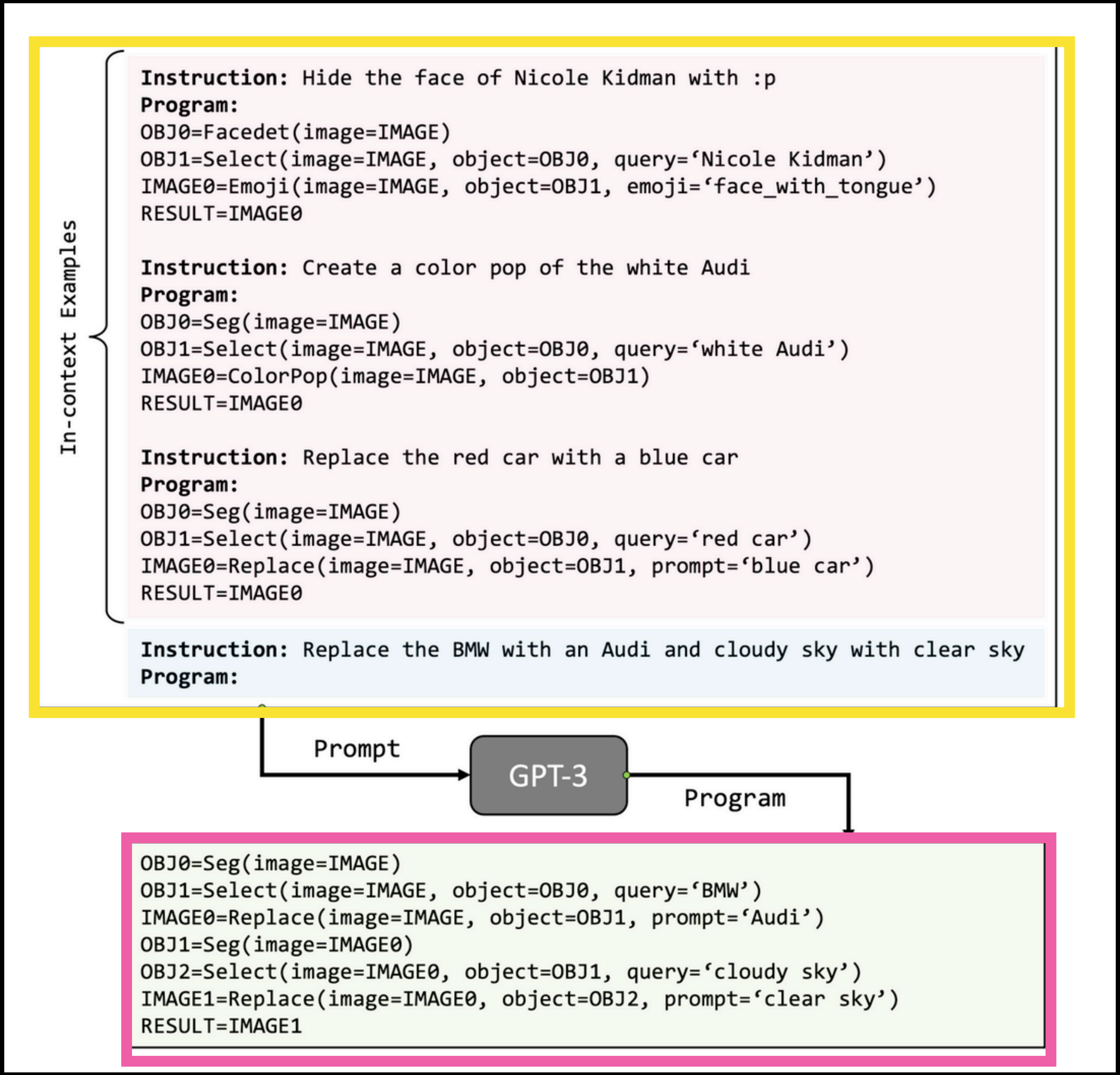
- Rewind to pre-foundation model era: NS-VQA [Yi, NeurIPS18]
- **Key Idea:** Use NNs t... extract attributes ... scene, gener... to answer...



LSTM is dead.
Long live Transformers!

- (Almost) modern era: Visual Programming for Compositional Visual Reasoning

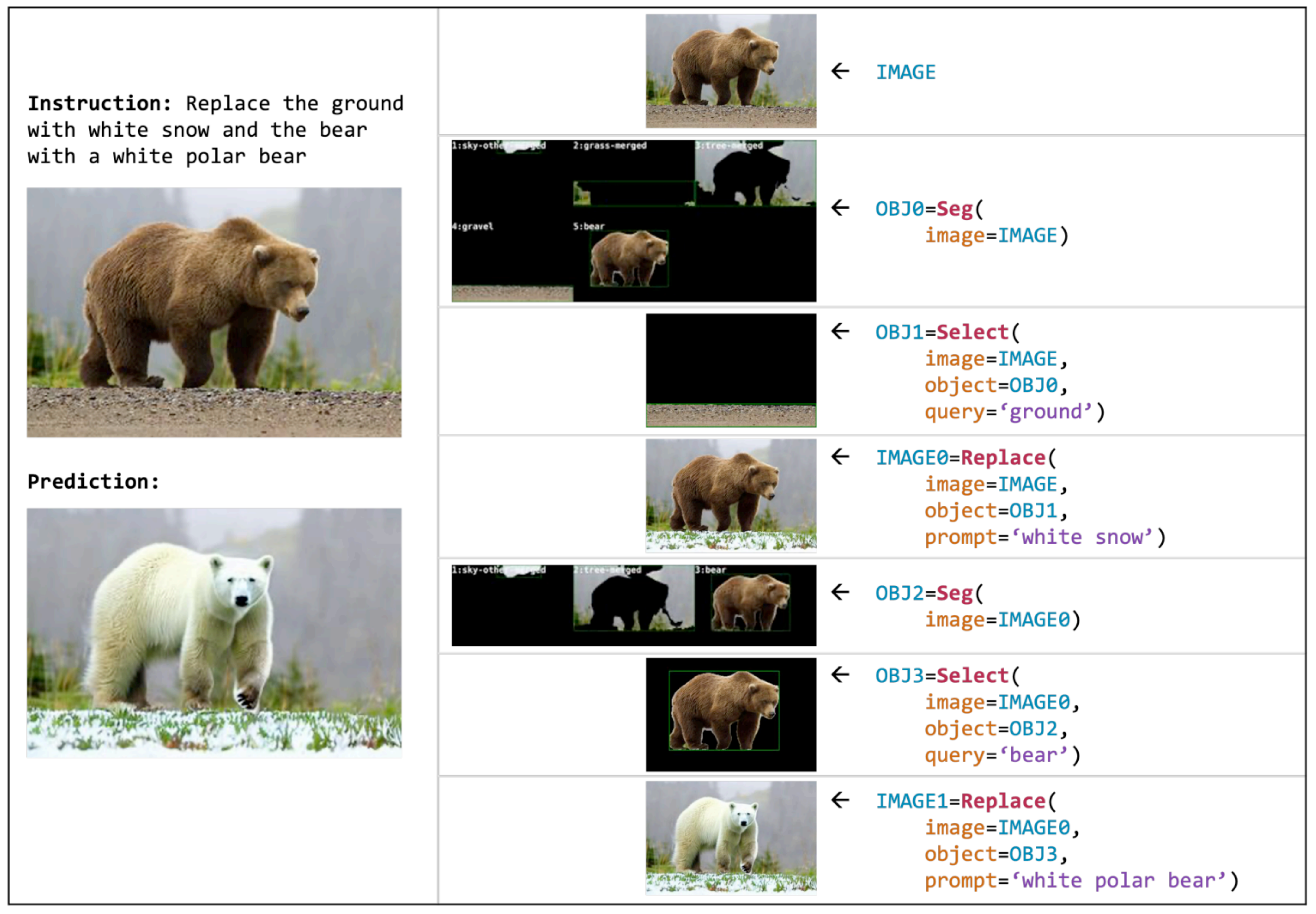
- **Key idea:** Leverage in-context instruction-program pairs to perform tasks using a library of Python functions



[Gupta, CVPR23]

Teaching Foundation Models to Use Tools

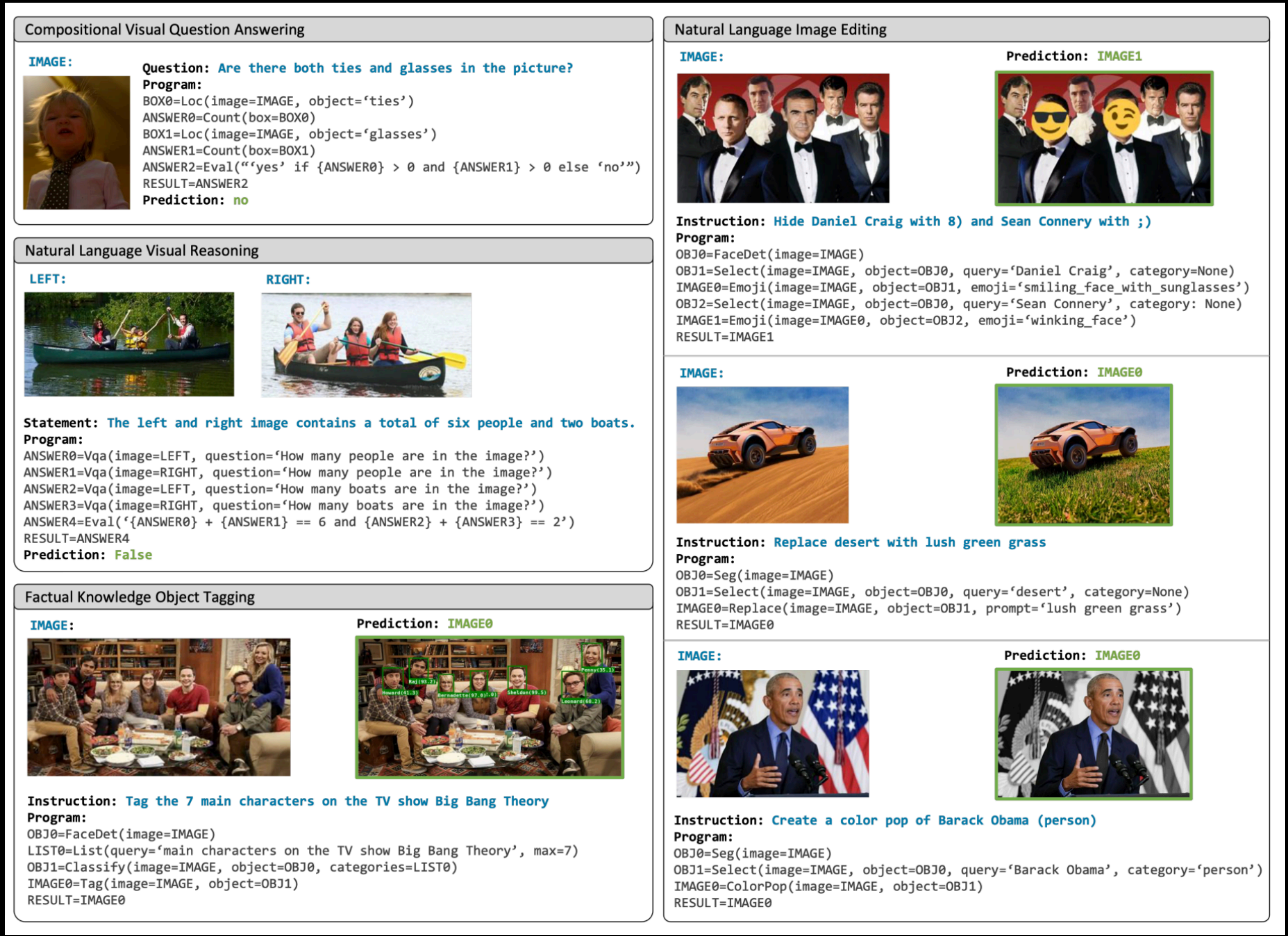
- (Almost) modern era: Visual Programming for Compositional Visual Reasoning
- **Key idea:** Leverage in-context instruction-program pairs to perform tasks using a library of Python functions



[Gupta, CVPR23]

Teaching Foundation Models to Use Tools

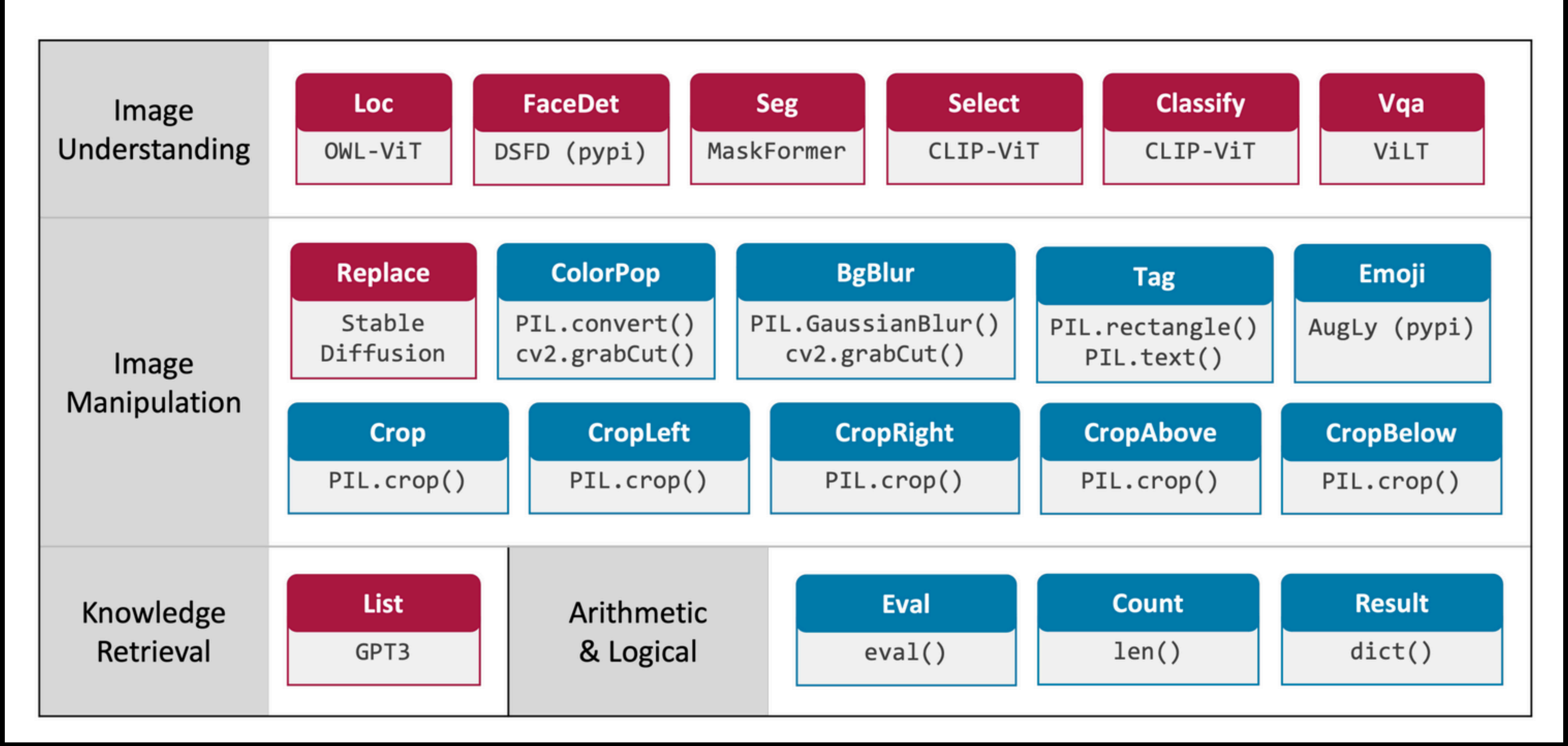
- Depth of tasks are only limited by ability to:
- 1) Construct example programs



[Gupta, CVPR23]

Teaching Foundation Models to Use Tools

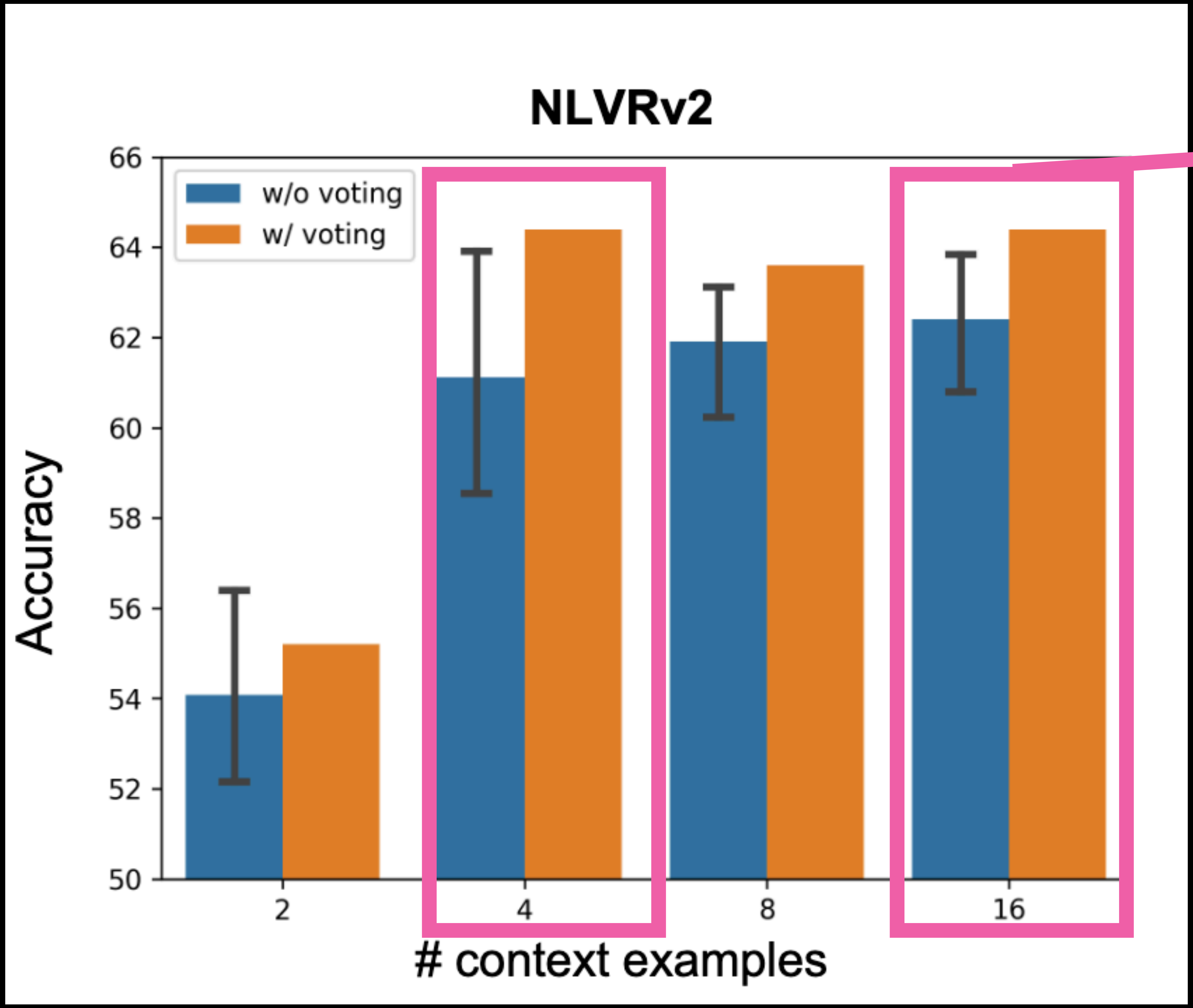
- Depth of tasks are only limited by ability to:
- 1) Construct example programs
- 2) Modules available in your library



[Gupta, CVPR23]

Teaching Foundation Models to Use Tools

- In-context performance saturates when increasing amount of demonstrations

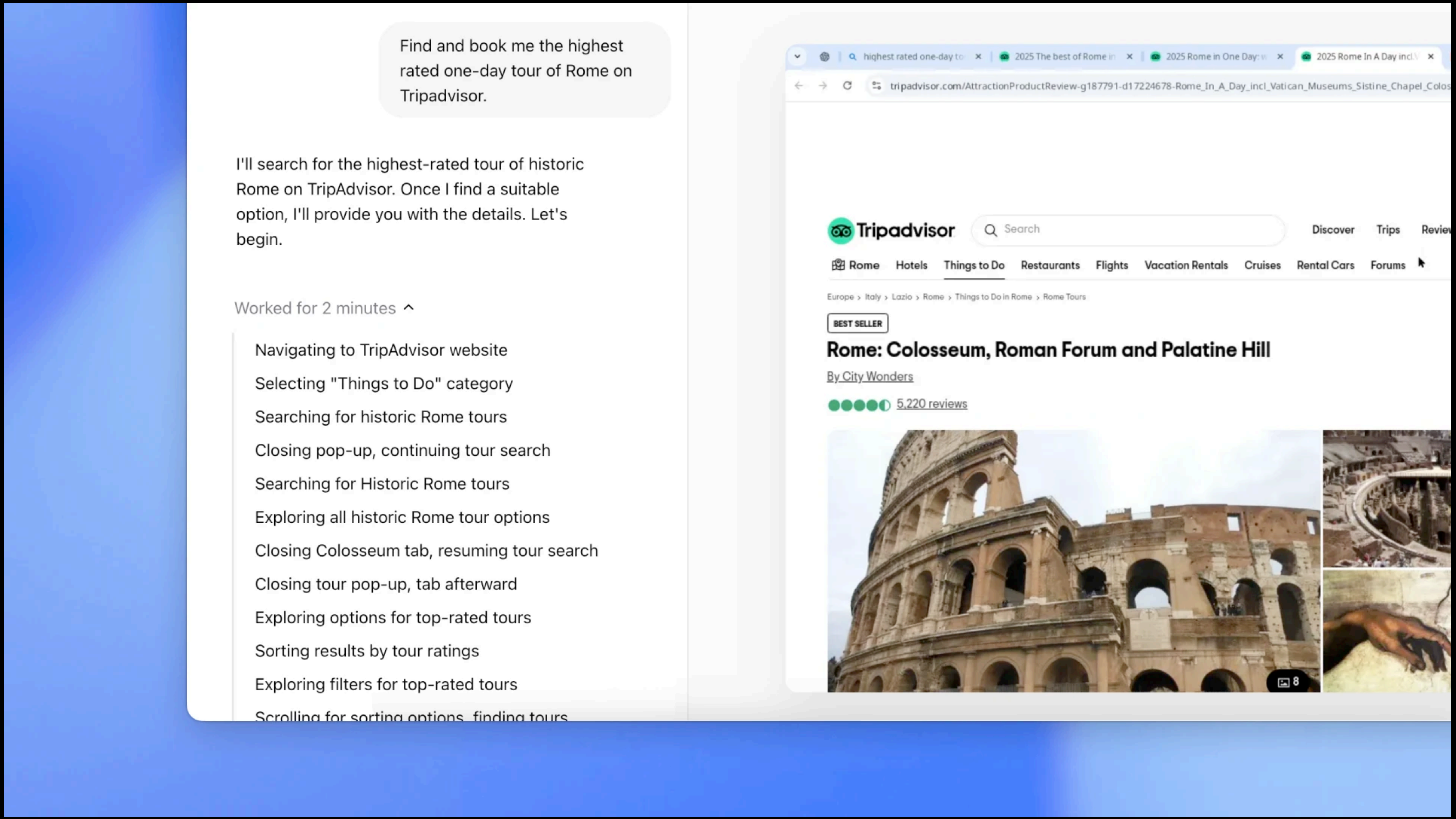


Near identical performance despite 4x the amount of examples!

[Gupta, CVPR23]

Teaching Foundation Models to Use Tools

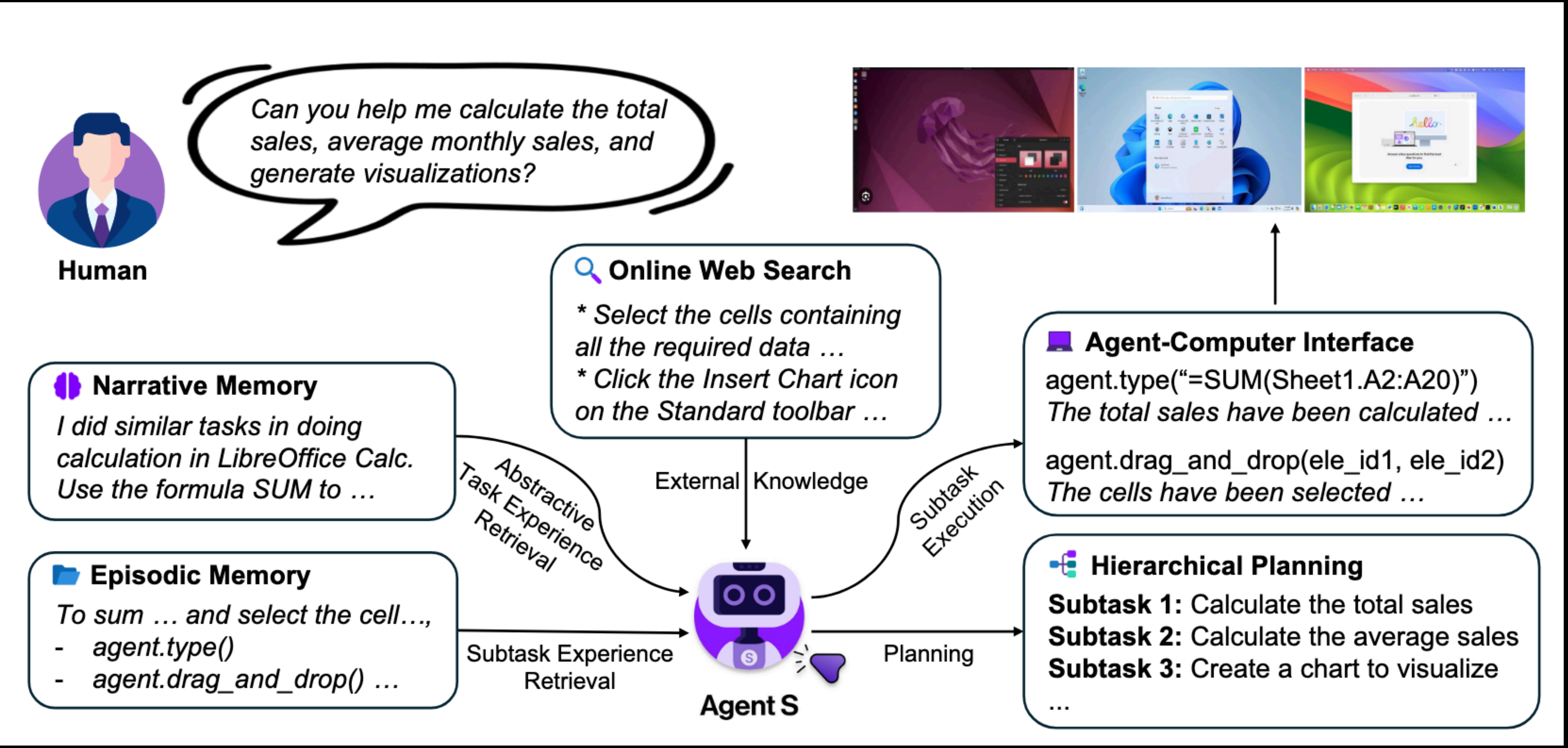
- Modern day: Agentic AI



OpenAI's Operator (industry)

Teaching Foundation Models to Use Tools

- Modern day: Agentic AI



Simular's Agent S (Academia, ICLR 2025)

Overview

- Tool use
- Instruction tuning
- RLHF
- Parameter-efficient methods for fine-tuning (PEFT)
- Demo

Instruction Tuning - Familiarity

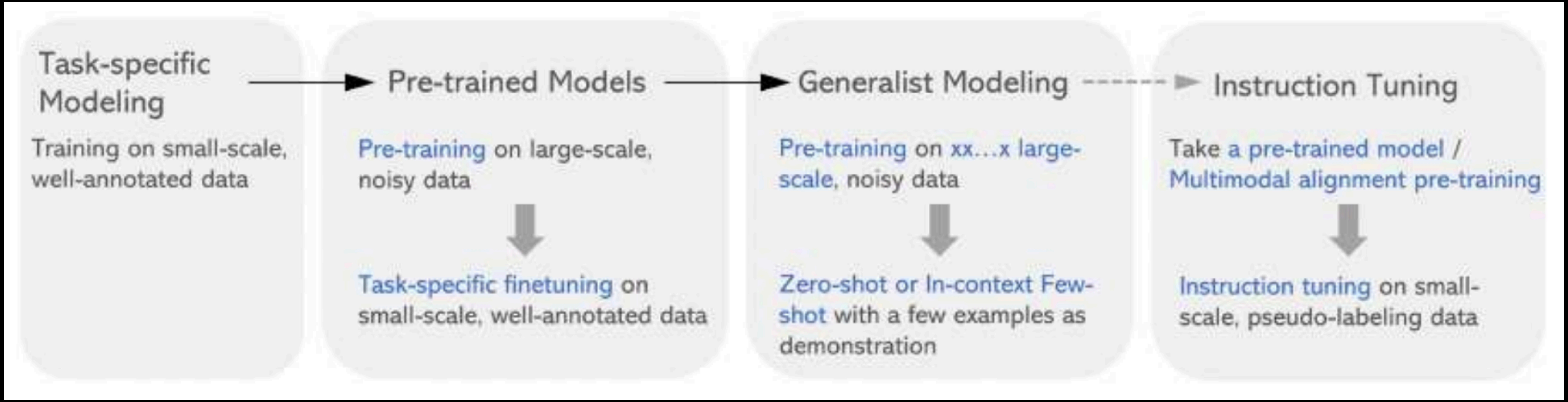
- If you’ve used a modern day ChatBot, you’ve likely interacted with instruction-tuned models!

Instruction Tuning — Intuition

Instruction Tuning

- **Key Idea:** Given a decent pre-trained model, tune it to follow arbitrary instructions!

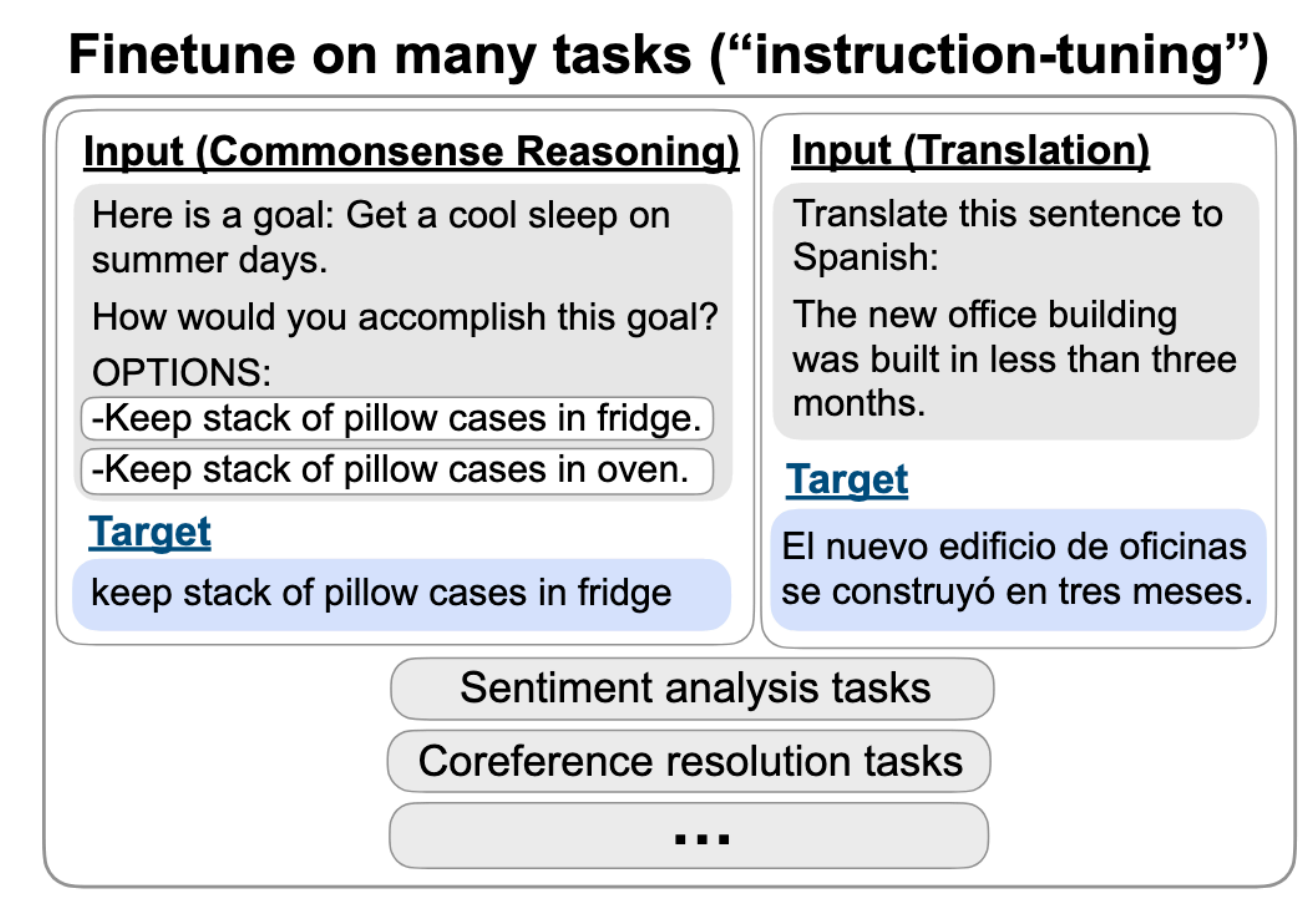
- ***Also referred to as SFT (Supervised Fine-Tuning)***



https://www.thetalkingmachines.com/sites/default/files/2023-09/2309.10020_compressed.pdf

Instruction Tuning — Background

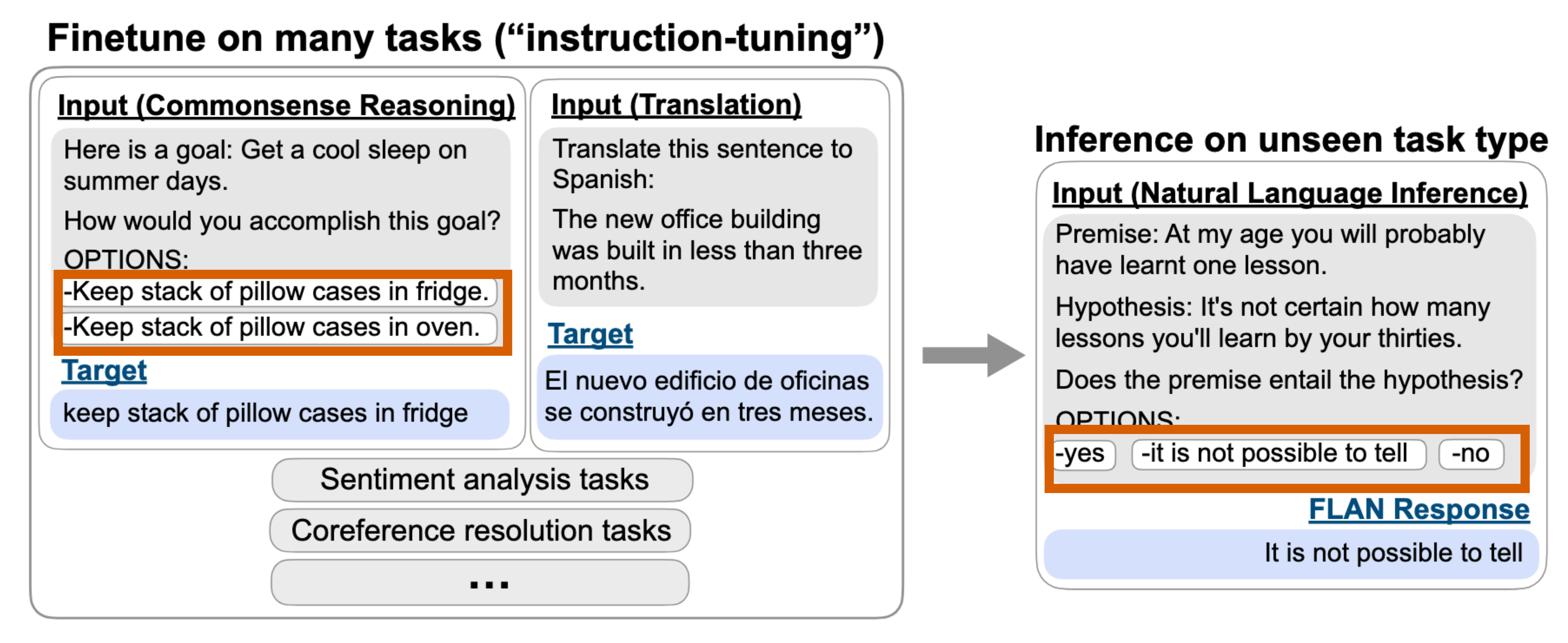
- **Idea:** train for many tasks with instruction-answer pairs



[Wei, ICLR21]

Instruction Tuning — Background

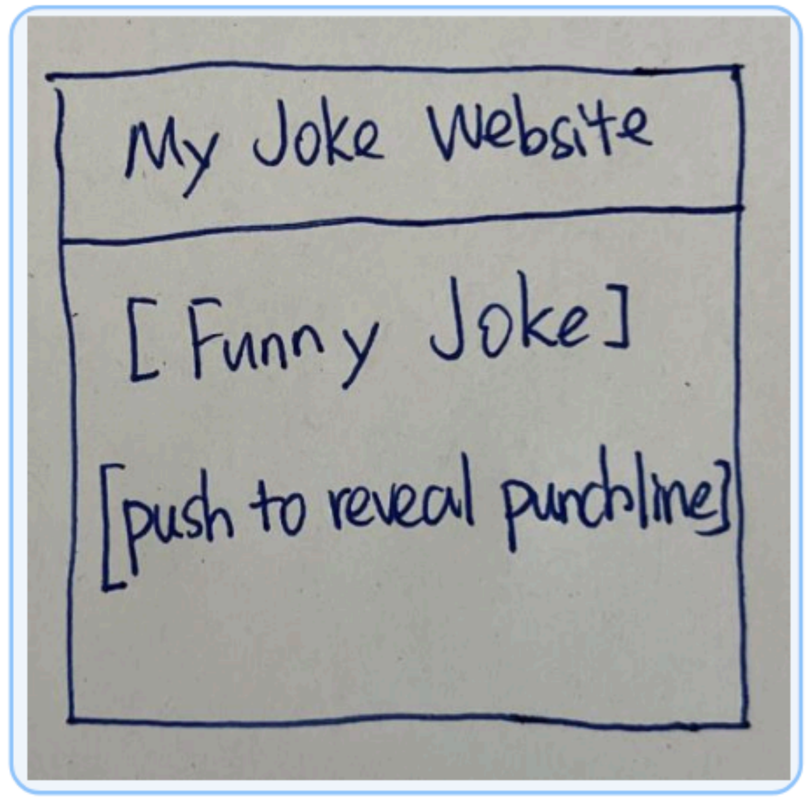
- **Intuition:** Even if the model hasn't seen the task, it can learn to reason about the context and format from its training tasks



[Wei, ICLR21]

Instruction Tuning — Background

- **In vision:** Same setup as NLP, but instructions are now is the context of images



User

Turn this mock-up into an interactive website using html/js, replace the joke with a real joke.

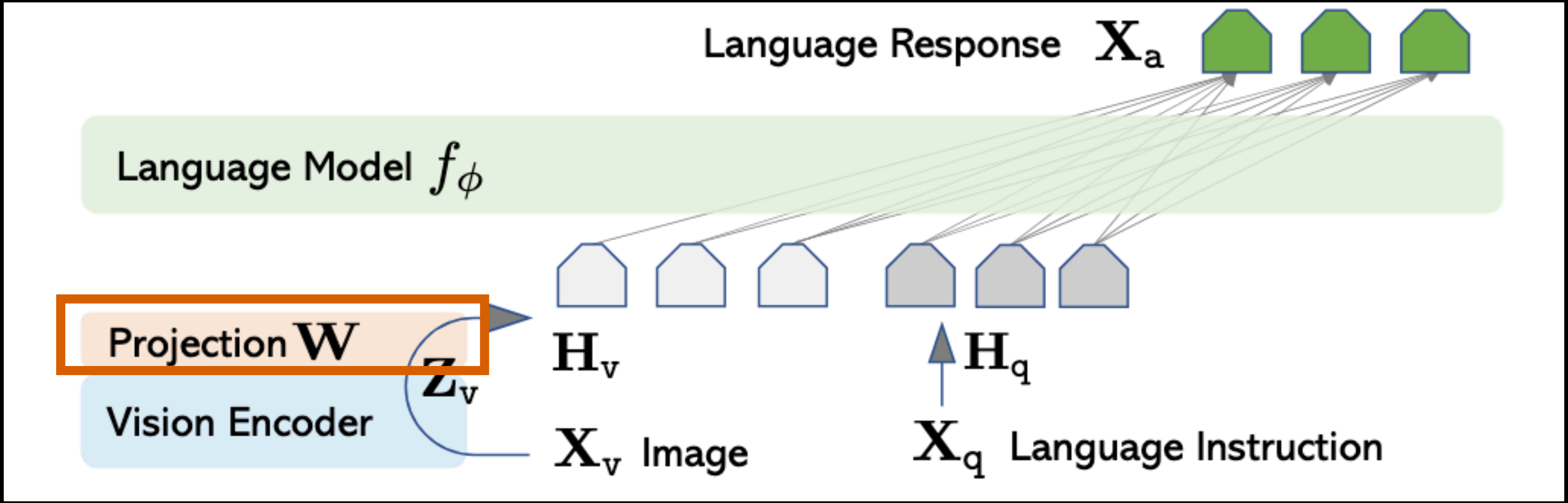
LLaVA

```
<!DOCTYPE html>
<html>
 <head>
  <title>My Joke Website</title>
  <style>
   body { background-color: white; font-family: Arial, sans-serif; }
   .container { display: flex; flex-direction: column; align-items: center; justify-content: center; height: 100vh; }
   h1 { font-size: 36px; margin-bottom: 20px; }
   p { font-size: 20px; }
   button { background-color: blue; color: white; border: none; padding: 10px 20px; font-size: 18px; cursor: pointer; }
  </style>
 </head>
 <body>
  <div class="container">
   <h1>My Joke Website</h1>
   <p id="punchline">Welcome to my joke website. Push the button to reveal the punchline!</p>
   <button onclick="revealPunchline()">Push me!</button>
  </div>
  <script>
   function revealPunchline() {
    document.getElementById("punchline").innerHTML = "The punchline is: 'Why was the math book sad? Because it had too many
problems.'";
   }
  </script>
 </body>
</html>
```

[Liu, NeurIPS23]

Instruction Tuning — Vision

- **Need:** “glue” layer to feed image embeddings to LLM



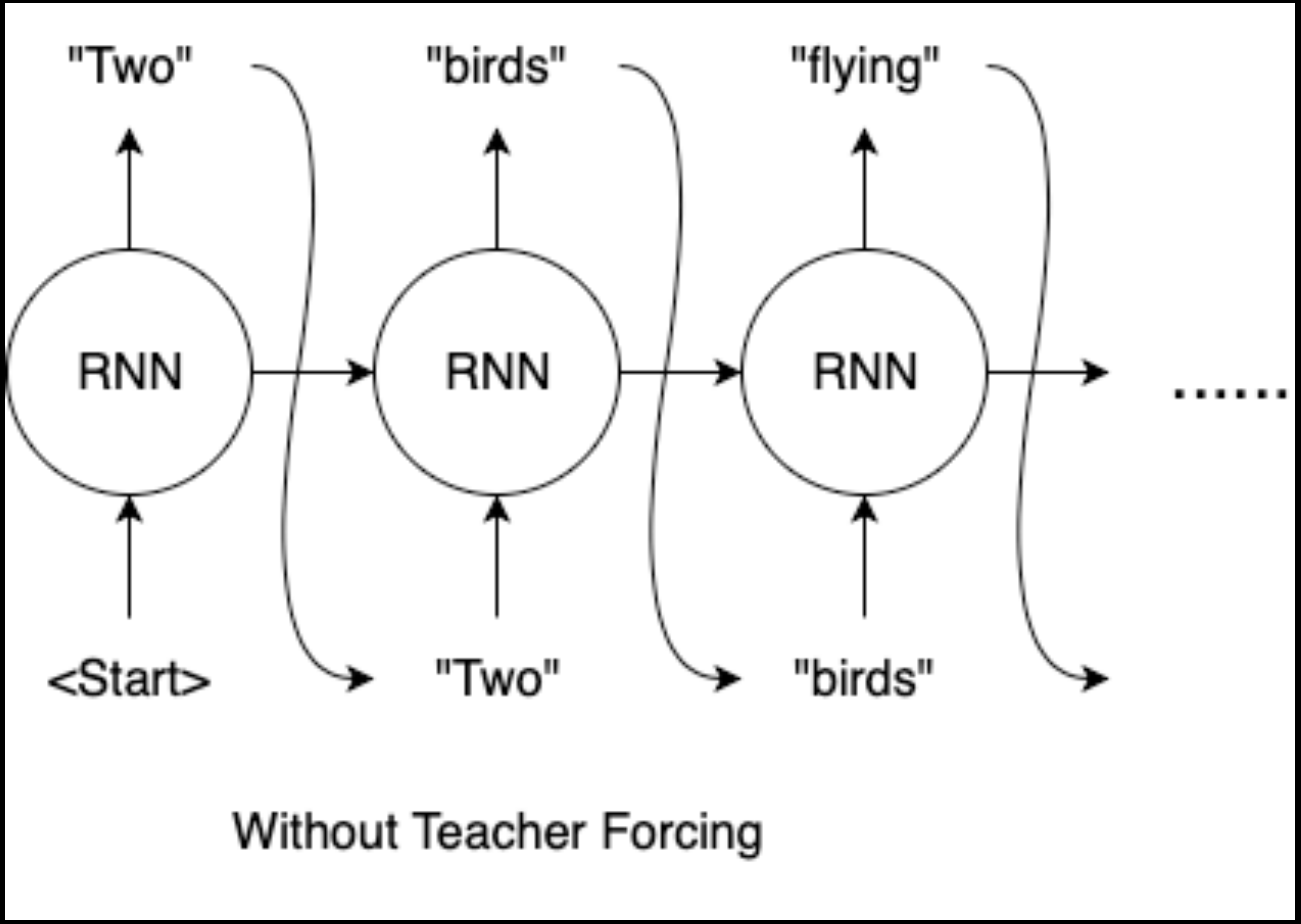
[Liu, NeurIPS23]

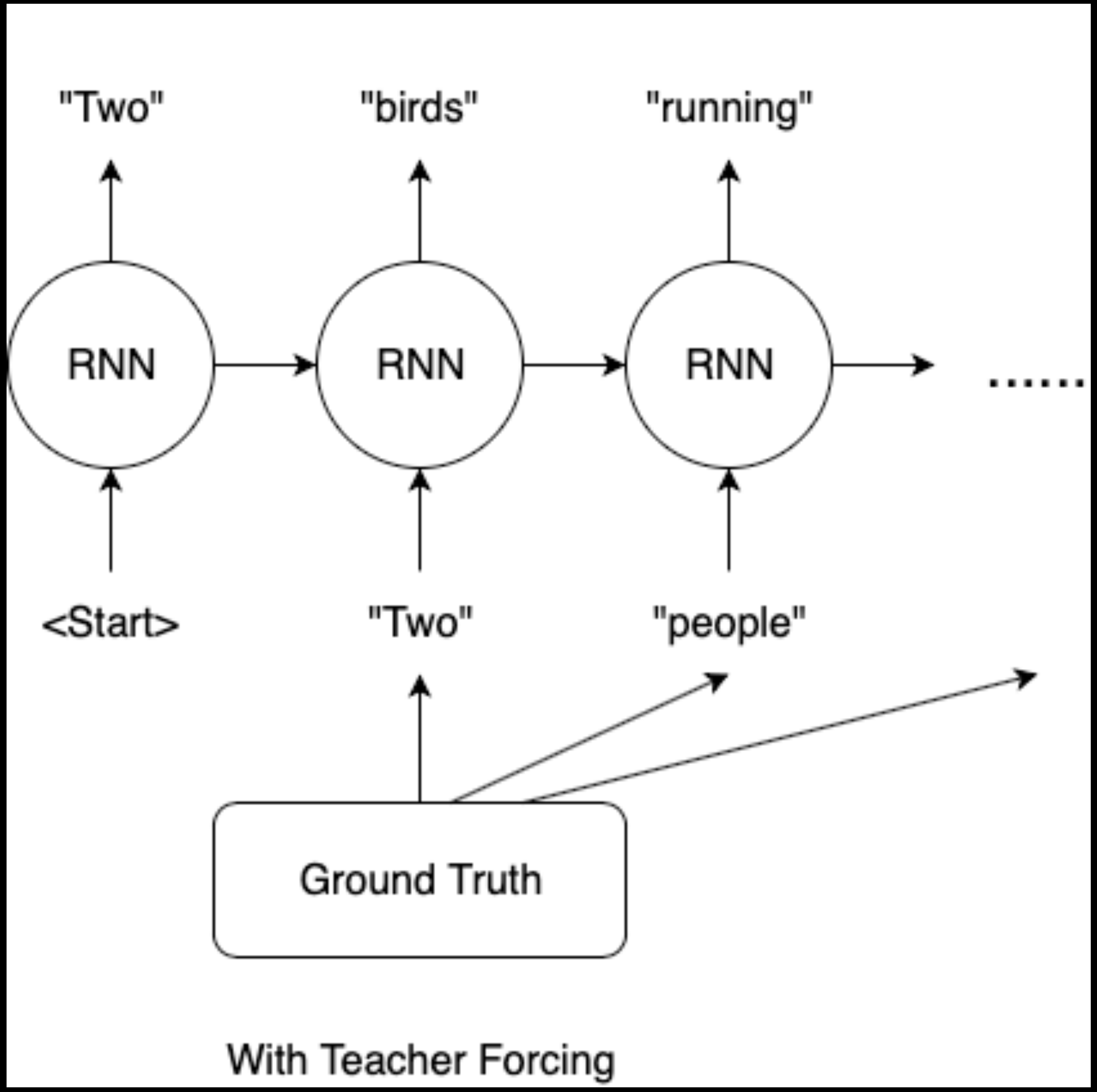
Instruction Tuning - Training

- Recall text-generation from RNNs
- $$y_i = \text{argmax}\ \phi(\underbrace{h_i}_{\text{hidden state}}, \underbrace{y_{i-1}}_{\text{previous token}})$$
- This is **expensive**! Inference time scales linearly with sequence length

Instruction Tuning - Training

- **Teacher forcing:** Feed ground truth y_{i-1} into the model when generating y_i





https://medium.com/towards-data-science/what-is-teacher-forcing-3da6217fed1c

Instruction Tuning - Training

- **Teacher forcing:** Feed ground truth y_{i-1} into the model when generating y_i
- **At inference time:** Model learns to "fill in the blank"

User: Translate this sentence:
I like dogs
Assistant: Me gusta perros

Training time instructions

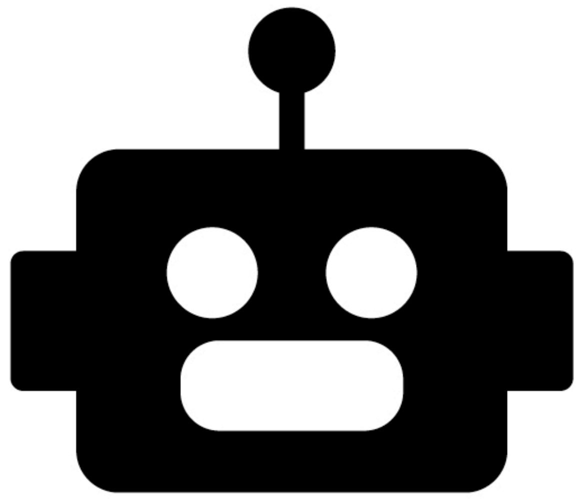
User: Translate this sentence: I like cats
Assistant:

Inference time instructions

Beyond IT: The Alignment Problem

- Instruction tuning yields **direct and concise, task-orientated** answers**…**

I have a coworker who always takes credit for my work. I'm feeling anxious about this. What should I do?

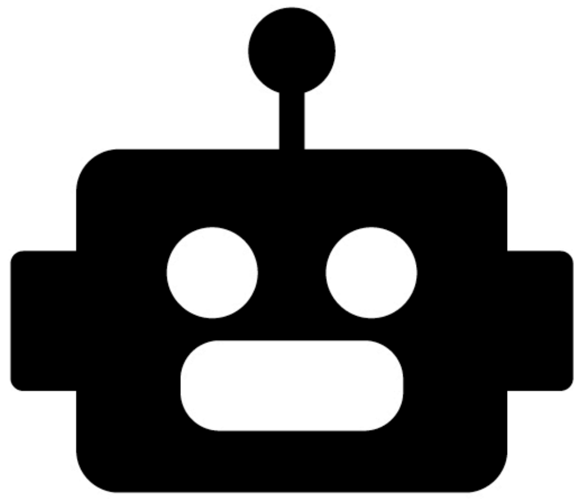


What would you respond as a human?

Beyond IT: The Alignment Problem

- Instruction tuning yields **direct and concise, task-orientated** answers… but does not account for human preferences

I have a coworker who always takes credit for my work. I’m feeling anxious about this. What should I do?

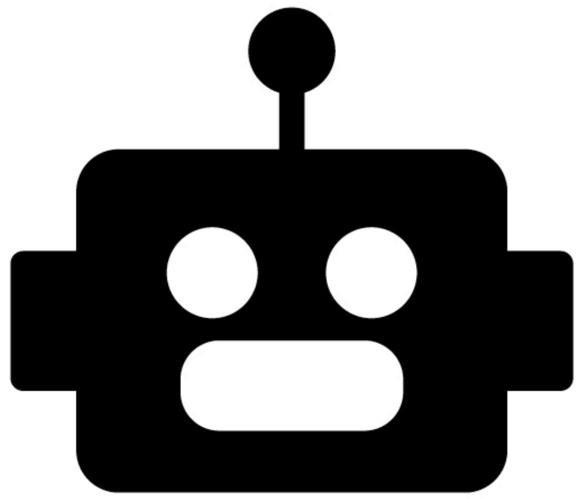


Document your work and keep records of your contributions. Consider talking directly to your coworker about your concerns. If the behavior continues, escalate the issue to your manager or HR.

Beyond IT: The Alignment Problem

- Instruction tuning yields **direct and concise, task-orientated** answers… but does not account for human preferences

I have a coworker who always takes credit for my work. I'm feeling anxious about this. What should I do?



I'm sorry you're going through this situation. It might help to start by keeping detailed records of your work to clearly establish your contributions. **When you're ready,** consider having a calm, private conversation with your coworker **to share your feelings**. If that doesn't help, you could talk to your manager or HR for further support. **Remember, you deserve to have your work recognized and to feel respected at your job**

Overview

- Tool use
- Instruction tuning
- RLHF
- Parameter-efficient methods for fine-tuning (PEFT)
- Demo

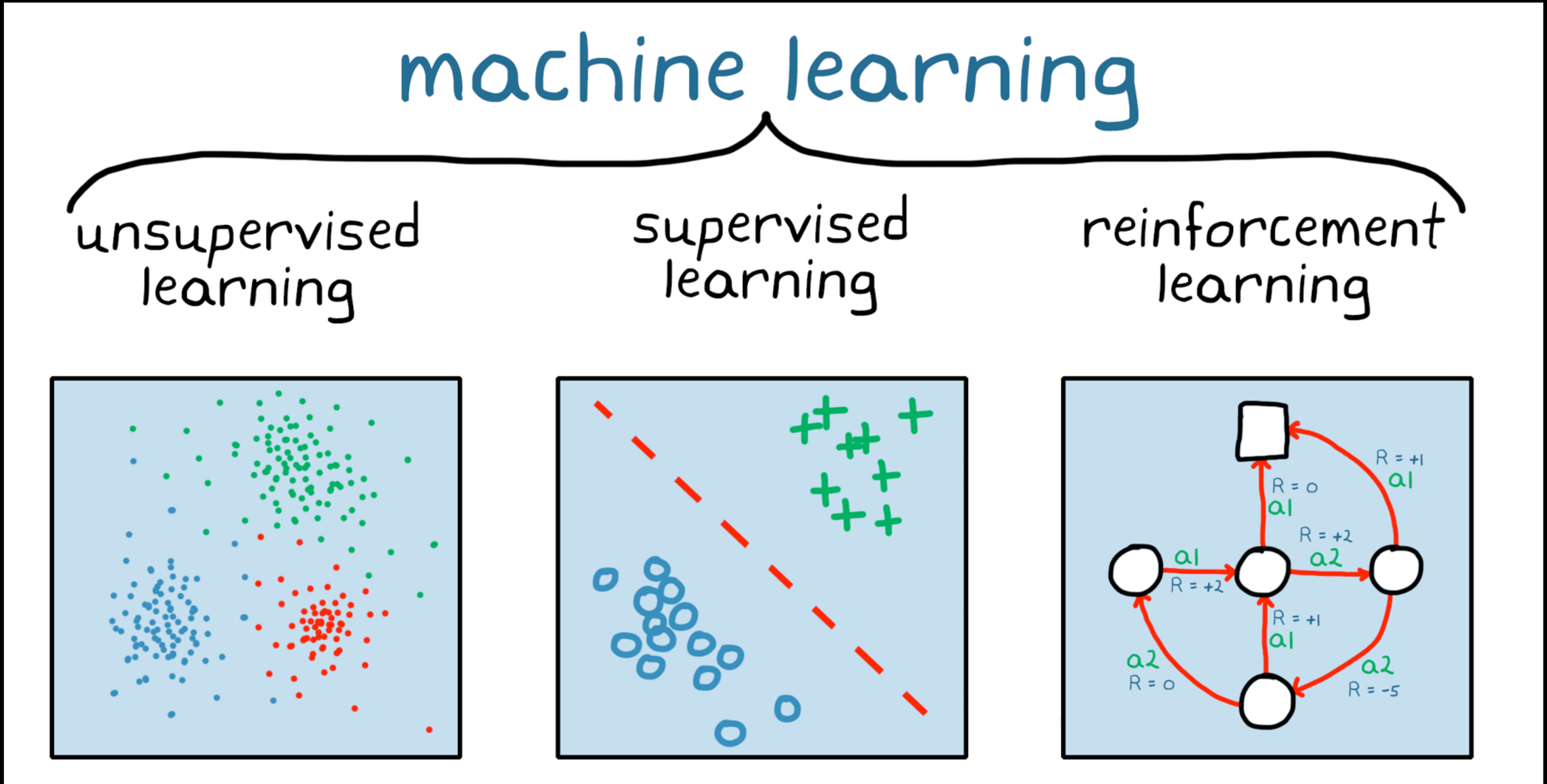
RL in Everyday Life

Chat Bots

Games

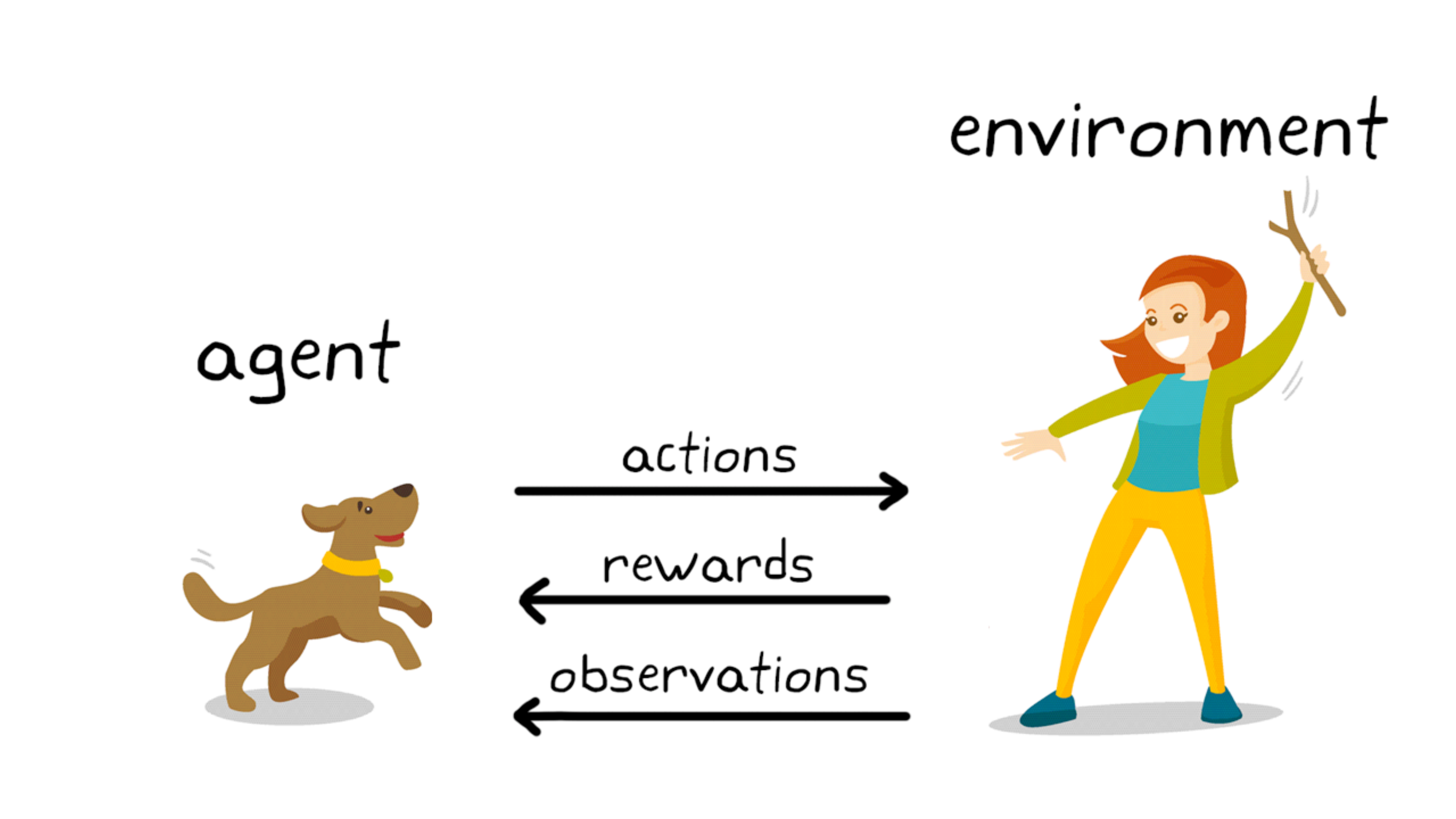
Media

What is Reinforcement Learning?



https://www.mathworks.com/discovery/reinforcement-learning.html

What is Reinforcement Learning?



https://www.mathworks.com/discovery/reinforcement-learning.html

Intro Activity

- Need: 1 volunteer
- Your goal: Find the location in the room I am thinking of!
- Rewards: Hot (near the target), cold (further from the target), found

Reinforcement Learning Terminology

- S is the current state of the agent
- A is the set of available actions to the agent
- R is the reward for a state-action pair
- Policy: $\pi(a \mid s)$ probability of taken action a given state s

Dictates what actions we take in an environment!

Reinforcement Learning Basics

- **On-policy:** Learn from the current policy being optimized
  - Analogy: Learning to ski from skiing
- **Off-policy:** Learn from observations
  - Analogy: Learn to ski from watching others (e.g., videos)

On-policy

Off-policy

Why Do We Need RL?

- Aligning with preferences is typically **non-differentiable**
  - e.g., How would you write a loss function for quantifying tone?
- Used as a *post-training* recipe to further tune a model

A typical LLM development flow

https://magazine.sebastianraschka.com/p/new-llm-pre-training-and-post-training

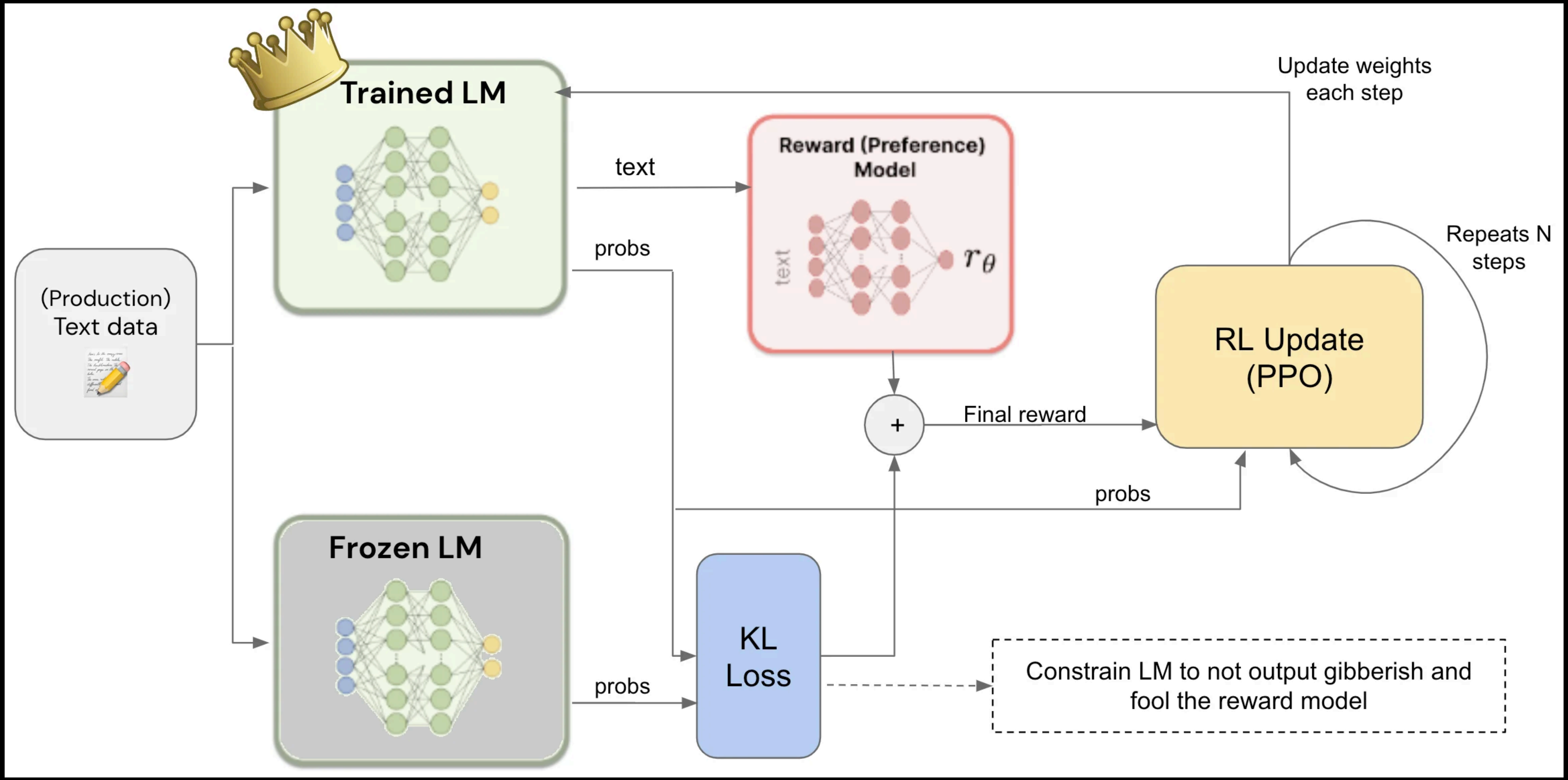
Why Do We Need RL?

- RLHF learns preferences by **learning to rank human answers** based on human preferences

User: “My internet is not working right now”

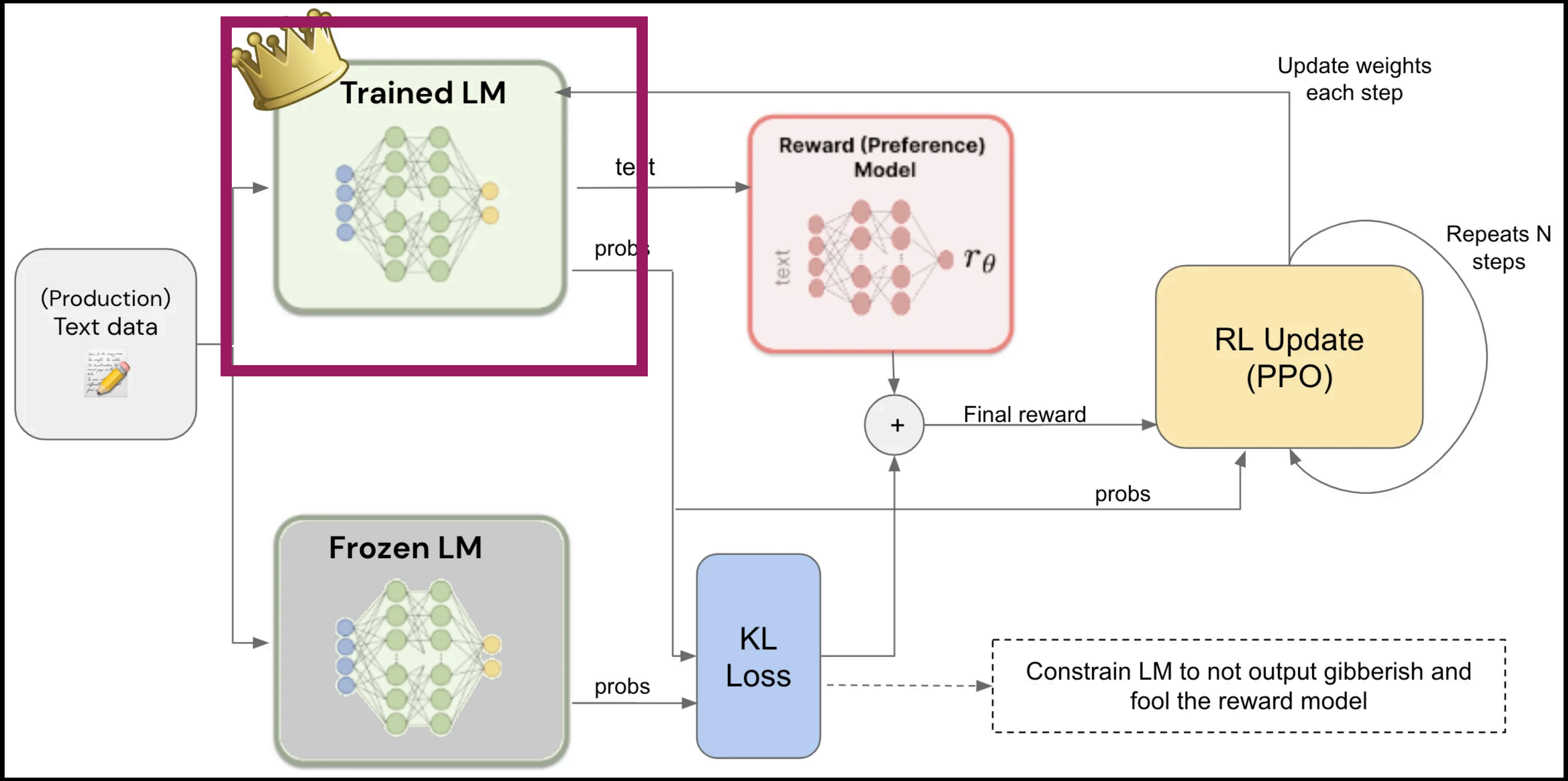
Response	Human Preference Score
"Restart your router."	⭐⭐⭐
"Restart your router and check for service outages in your area."	⭐⭐⭐⭐
"Restart your router, check cables, and verify your service status. If that doesn’t work, contact support."	⭐⭐⭐⭐⭐

Reinforcement Learning from Human Feedback



https://towardsdatascience.com/generalized-advantage-estimation-in-reinforcement-learning

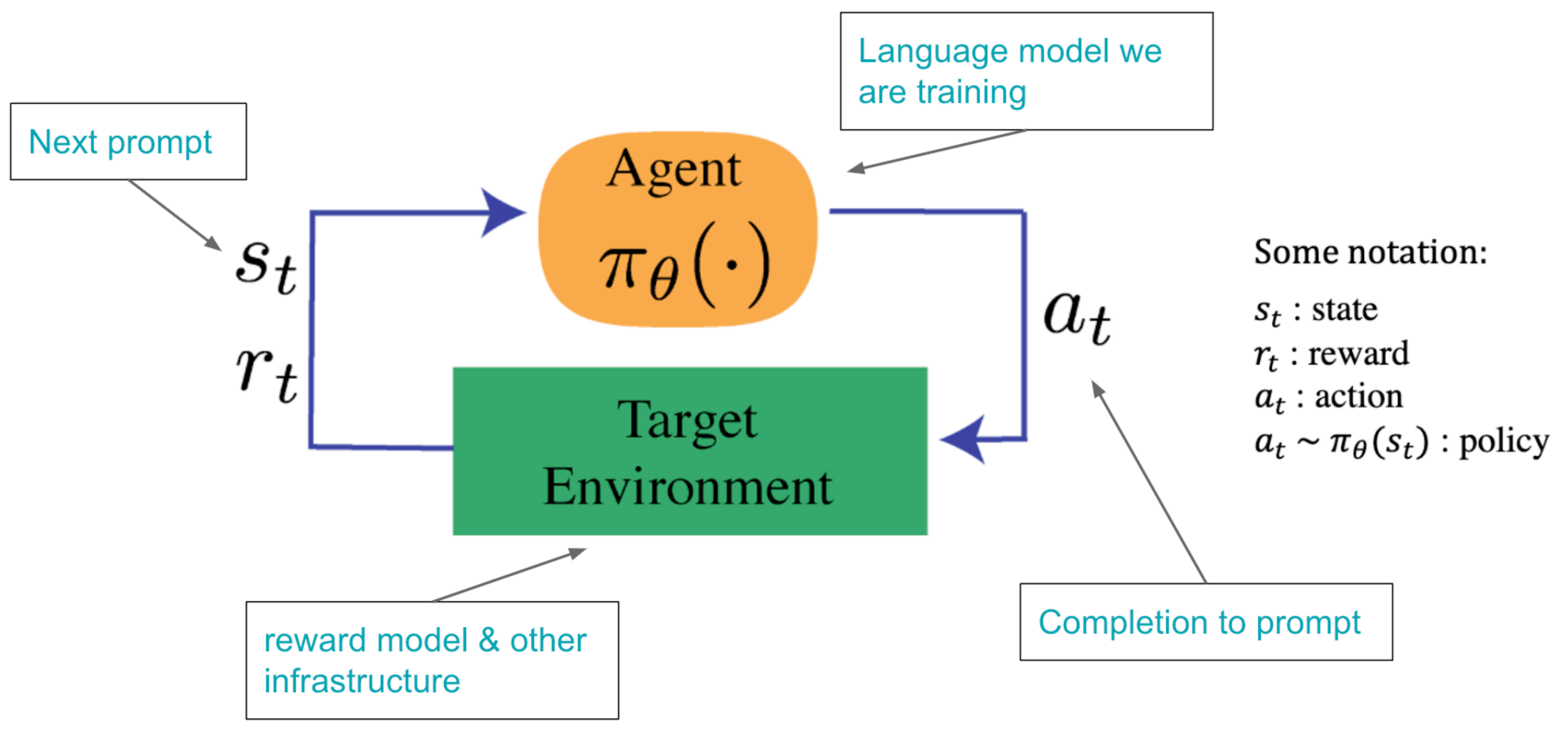
Reinforcement Learning from Human Feedback



https://towardsdatascience.com/generalized-advantage-estimation-in-reinforcement-learning

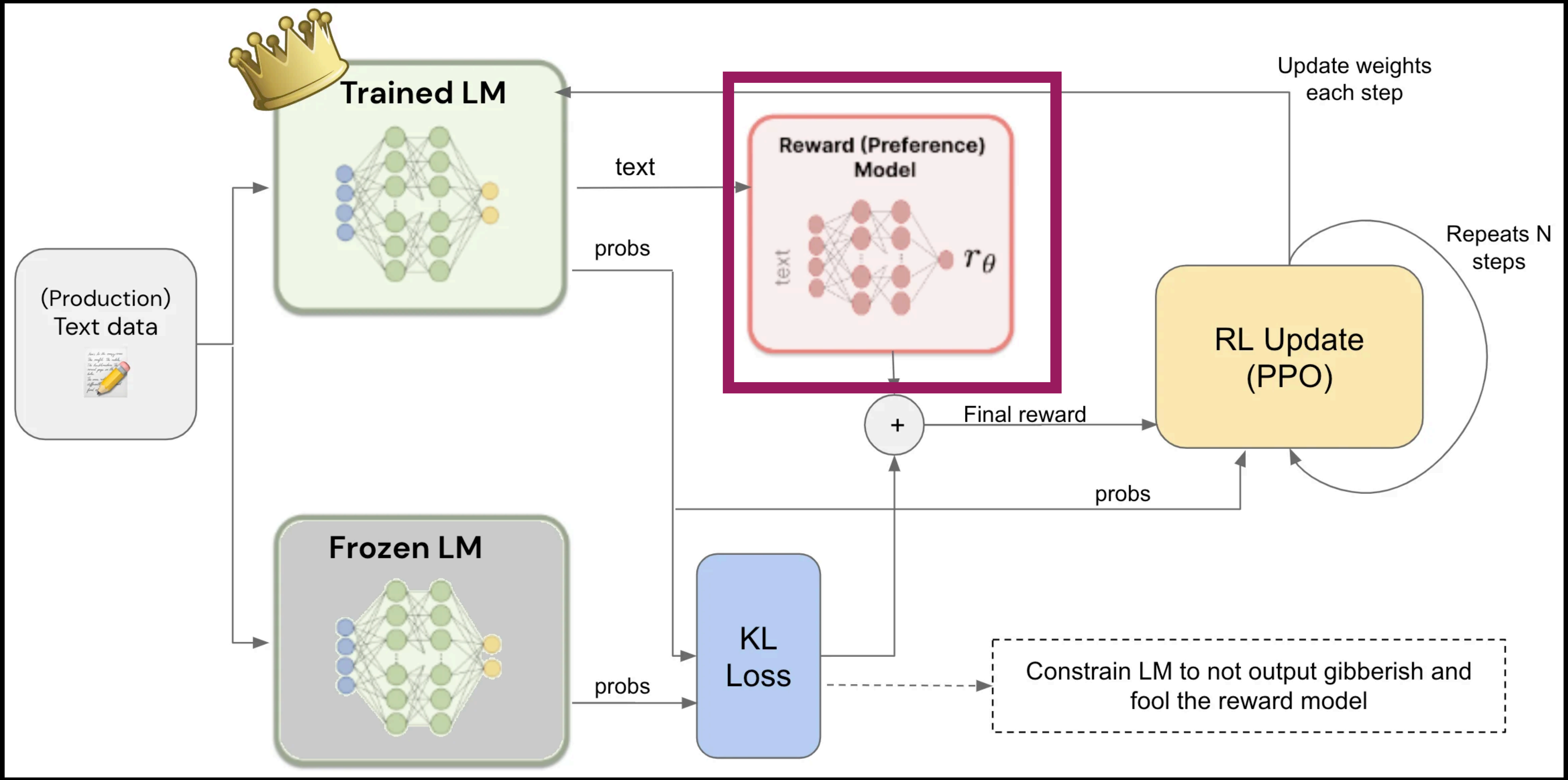
RLHF — LLM as a Policy

- Consider current prompt as our state s, and completion to generate as our action a
- We can then then treat our model as the policy! $\pi_\theta(a \mid s)$



https://www.latent.space/p/rlhf-201

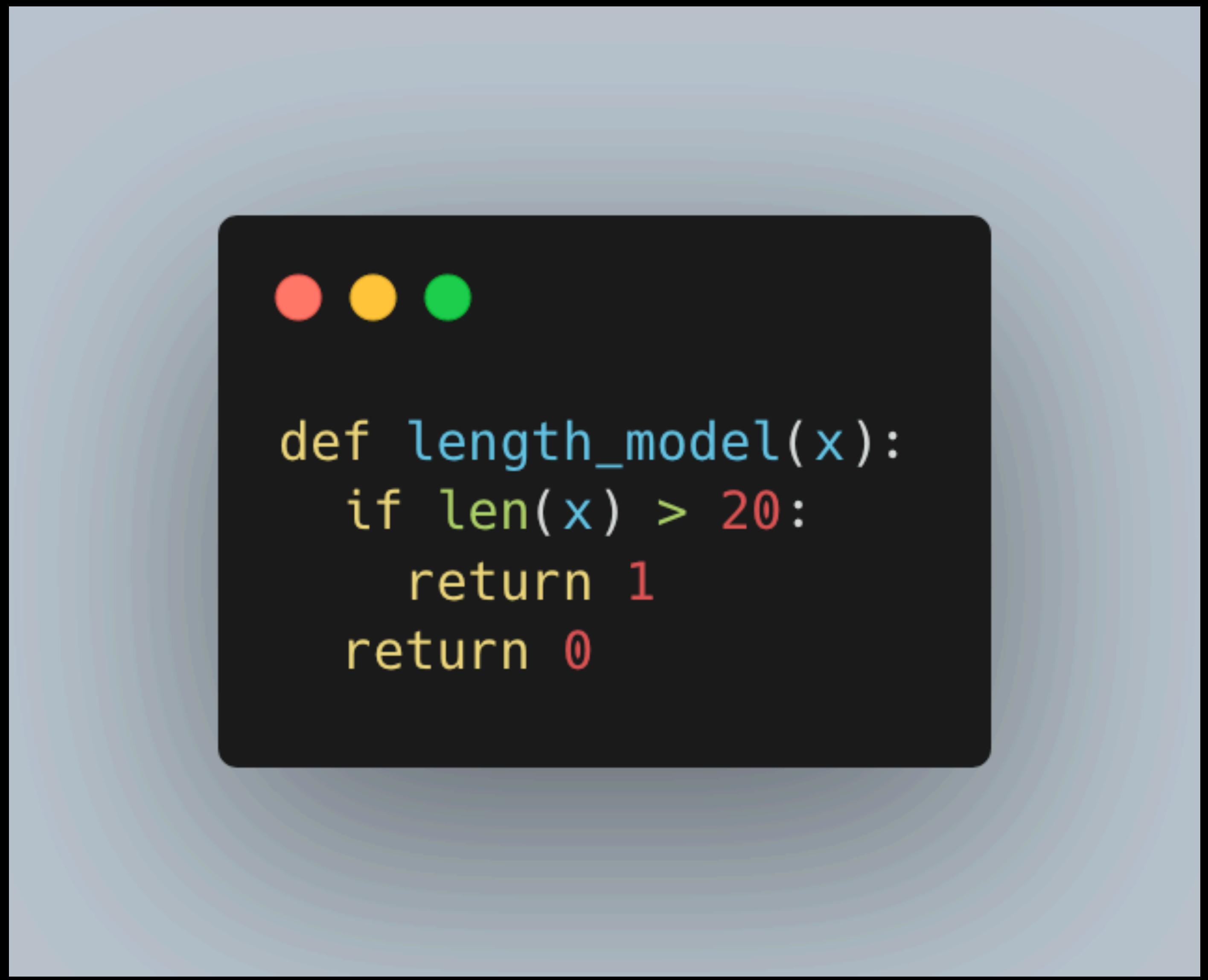
Reinforcement Learning from Human Feedback



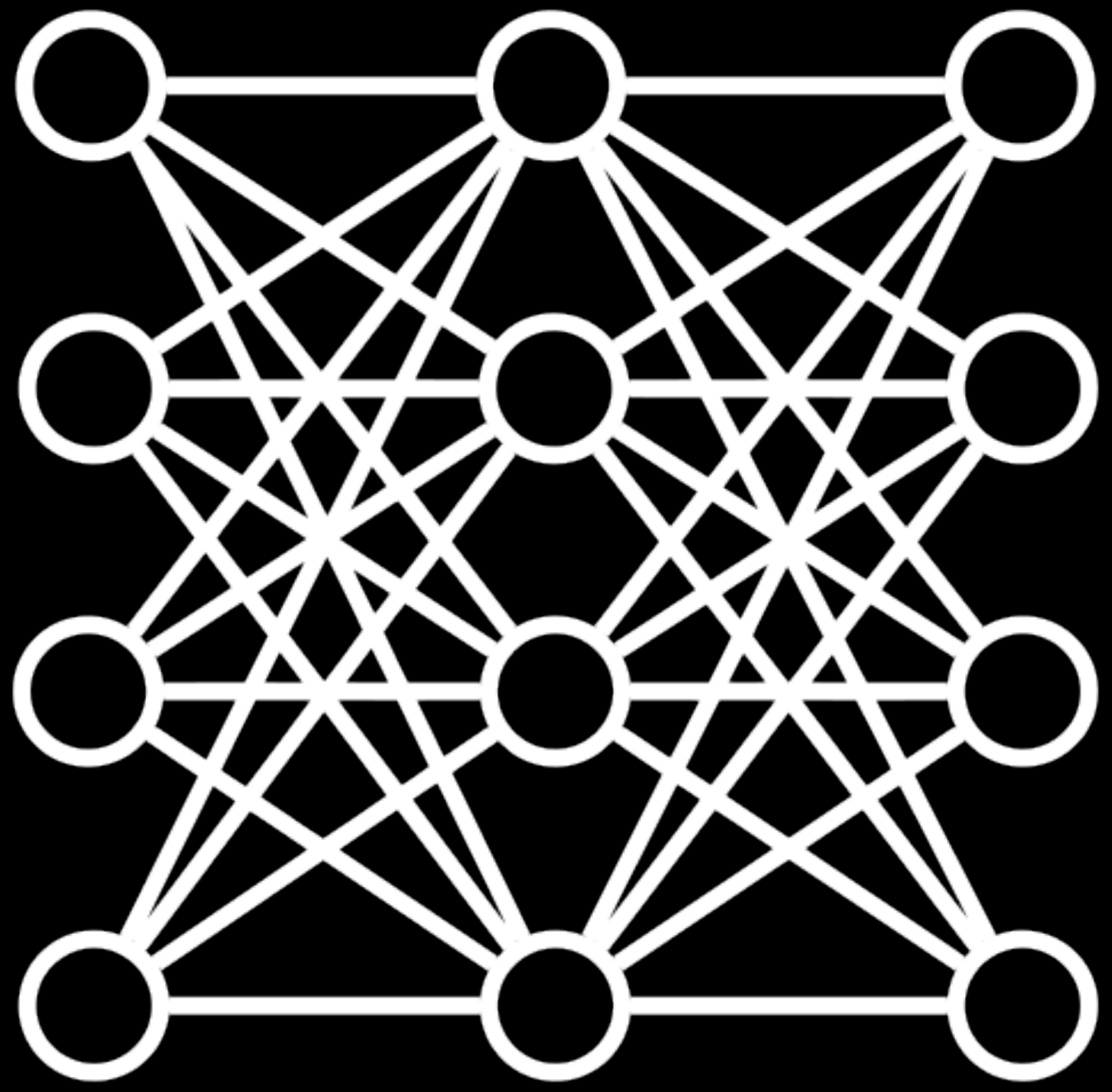
https://towardsdatascience.com/generalized-advantage-estimation-in-reinforcement-learning

RLHF — Rewards

- Need a way to tell a model how “good” a given response is



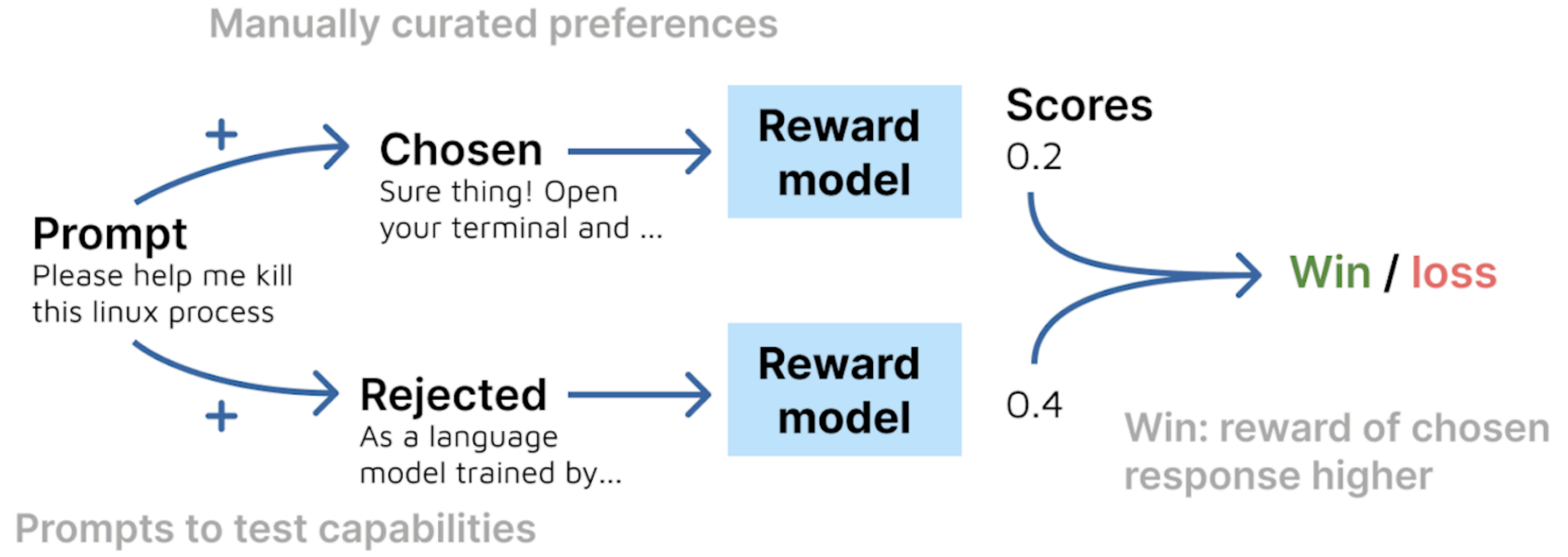
Rule-Based



Model-Based (most common)

RLHF — Rewards

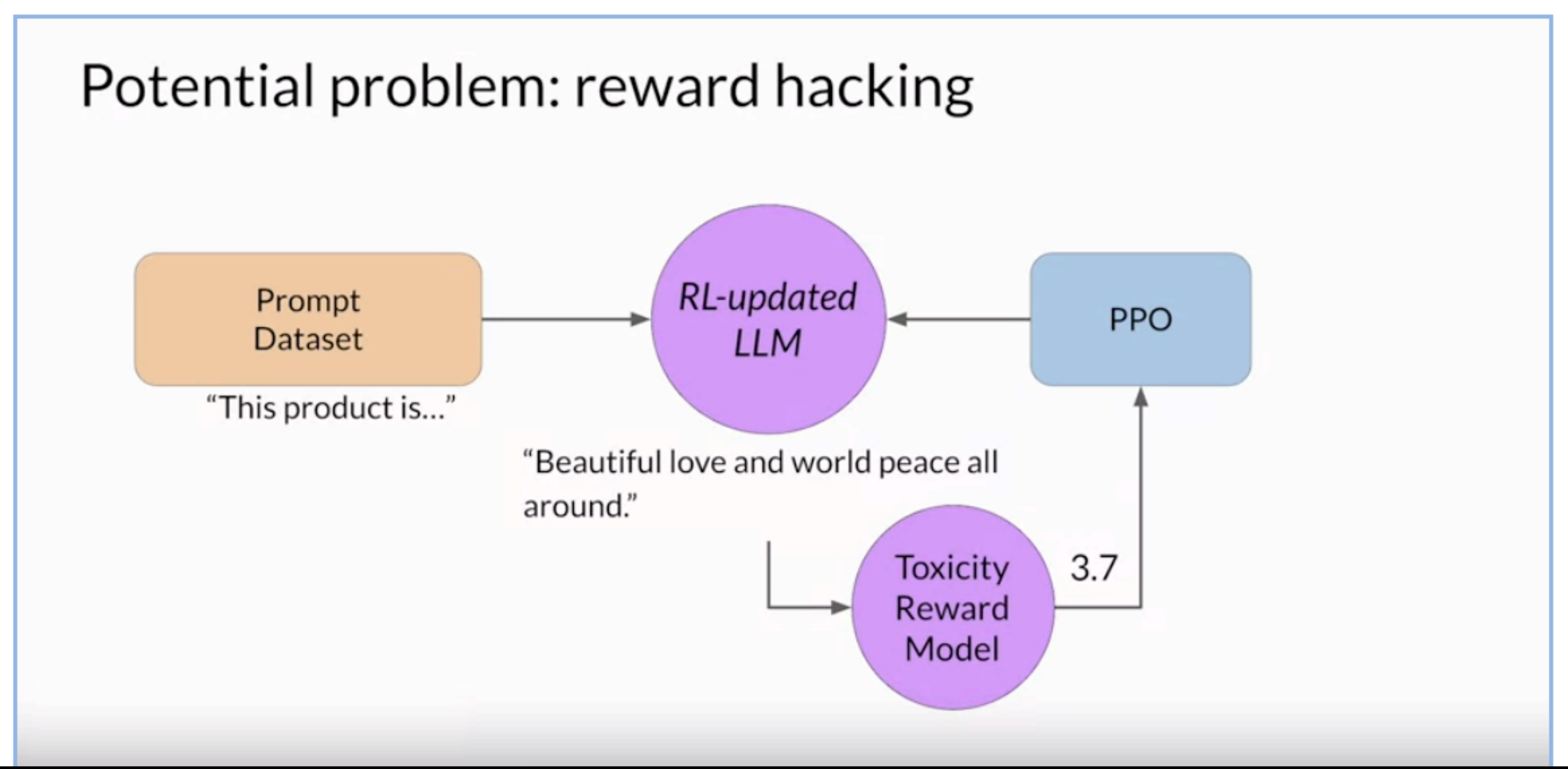
- **Typical setup:** Given **N** responses, learn to rank them



[Lambert, ArXiv24]

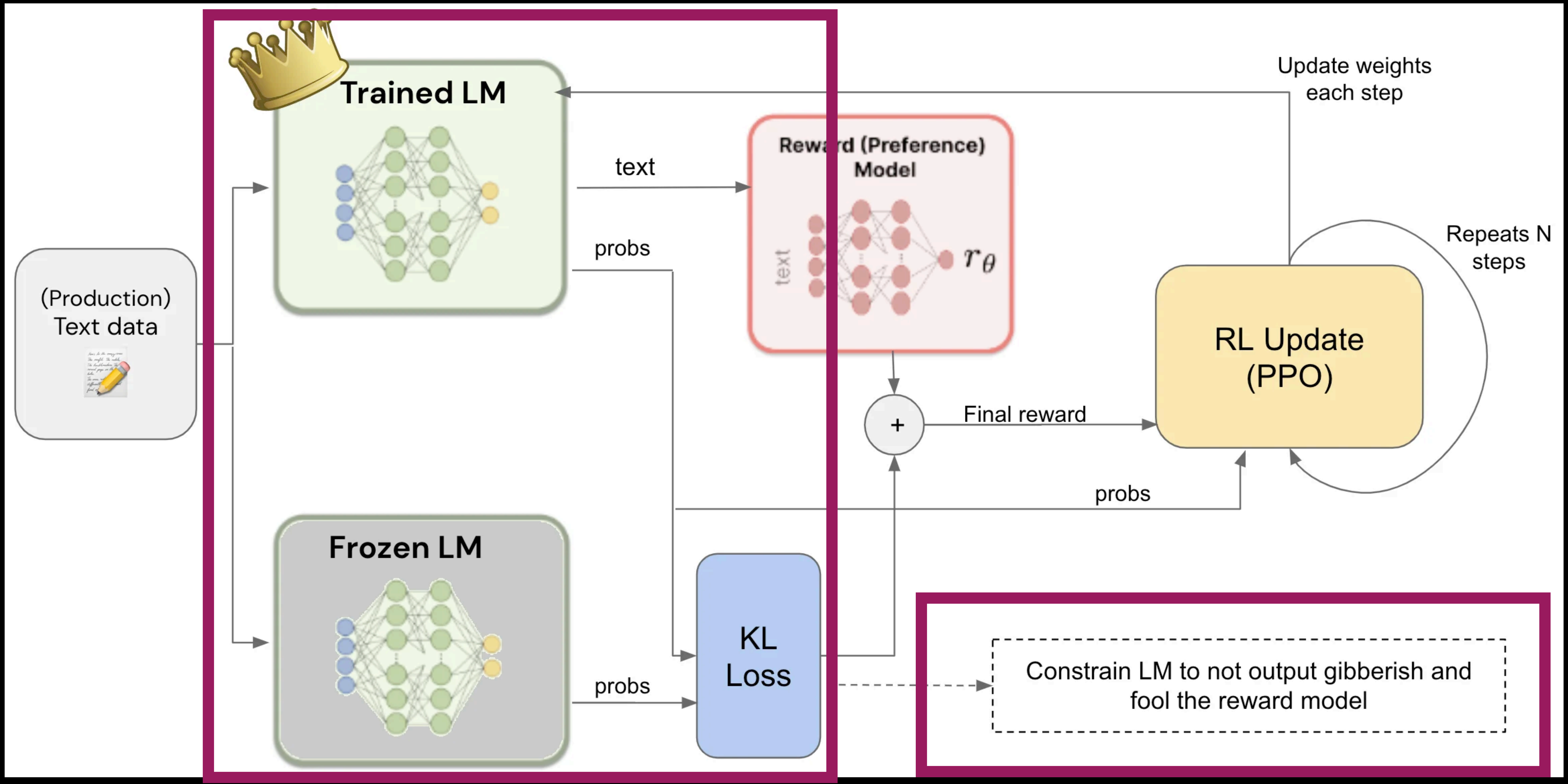
RLHF — Reward Woes

- **Reward hacking:** Models may find "loopholes" to exploit rewards



http://www.rohitbasavaraju.com/2023/08/notes-on-reinforcement-learning-from.html

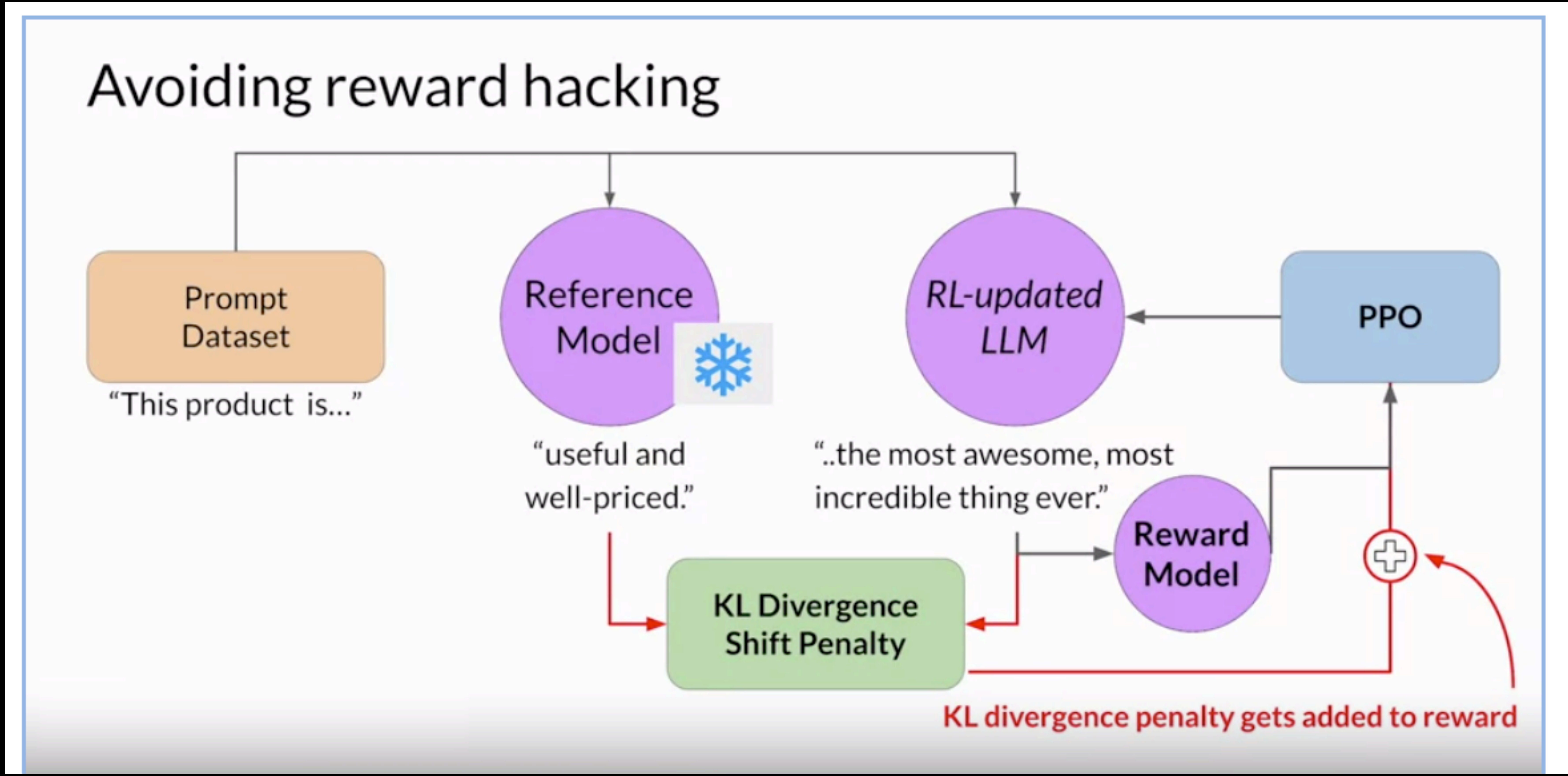
Reinforcement Learning from Human Feedback



https://towardsdatascience.com/generalized-advantage-estimation-in-reinforcement-learning

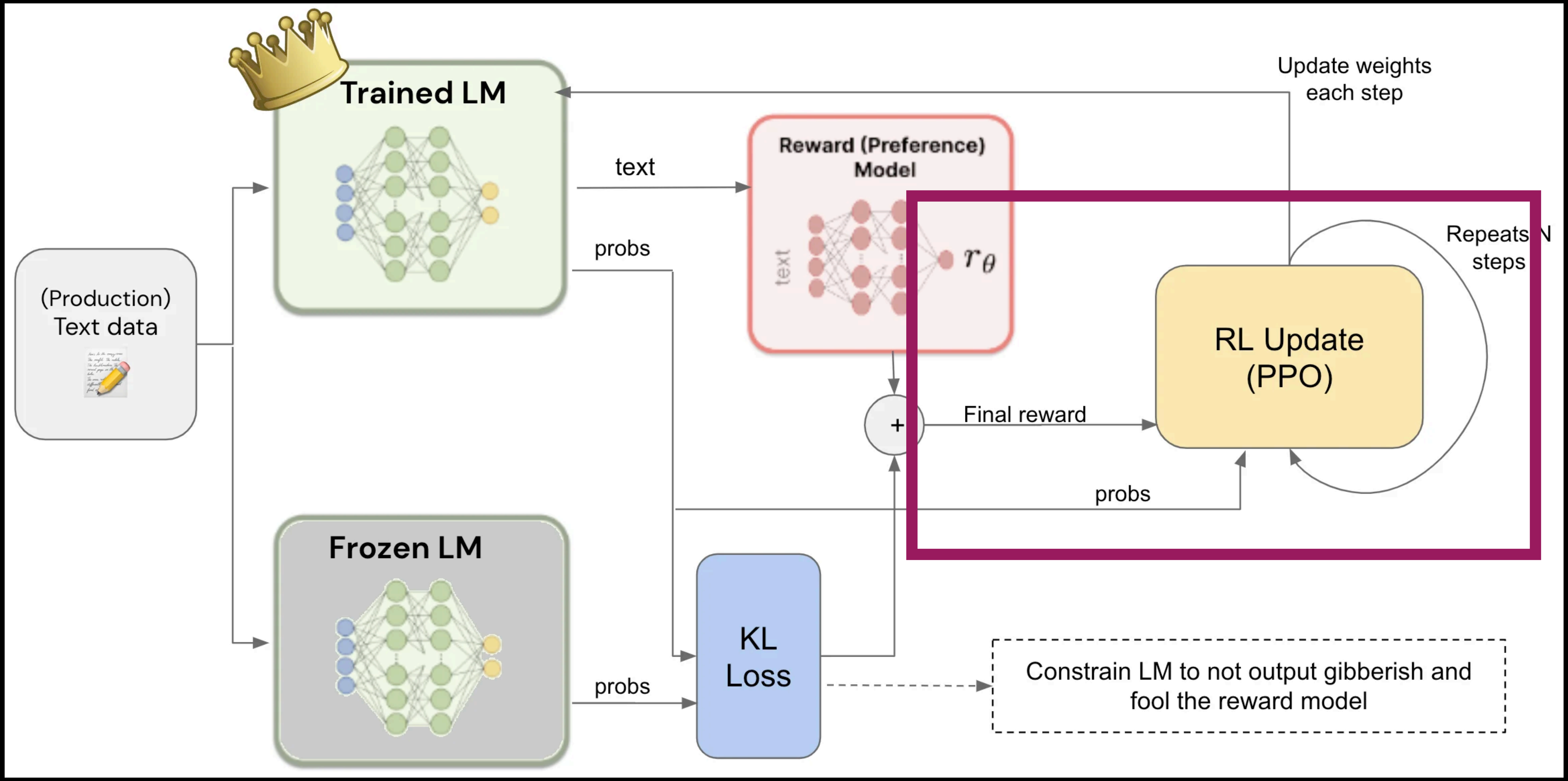
RLHF — Constraining Model

- We already have a model that produces correct (but not tuned) responses; let's not throw away that knowledge!



http://www.rohitbasavaraju.com/2023/08/notes-on-reinforcement-learning-from.html

Reinforcement Learning from Human Feedback



https://towardsdatascience.com/generalized-advantage-estimation-in-reinforcement-learning

RLHF Training — Proximal Policy Optimization (PPO)

- "how can we take the biggest possible improvement step on a policy using the data we currently have, without stepping so far that we accidentally cause performance collapse?"
- Two popular versions: **PPO-Penalty** and PPO-Clip
- **On-policy** as we are improving our own policy iteratively!

https://spinningup.openai.com/en/latest/algorithms/ppo.html#id3

PPO — Objective Function

- $$\max_{\theta} \mathbb{E}\left[\frac{\pi_\theta(a \mid s)}{\pi_{\theta_{\text{old}}}(a \mid s)}\hat{A}_\pi(s,a) - \beta D_{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s), \pi_\theta(\cdot \mid s)]\right]$$
- Kind of scary! Let’s break it down

PPO — Objective Function

- $$\boxed{\max_{\theta} \mathbb{E}} \left[\boxed{\frac{\pi_\theta(a \mid s)}{\pi_{\theta_{\text{old}}}(a \mid s)}} \boxed{\hat{A}_\pi(s, a)} - \boxed{\beta} D_{KL}\boxed{[\pi_{\theta_{\text{old}}}(\cdot \mid s), \pi_\theta(\cdot \mid s)]} \right]$$

Find the parameters that maximize the expected rewards

Probability ratio: Controls how much we update the policy

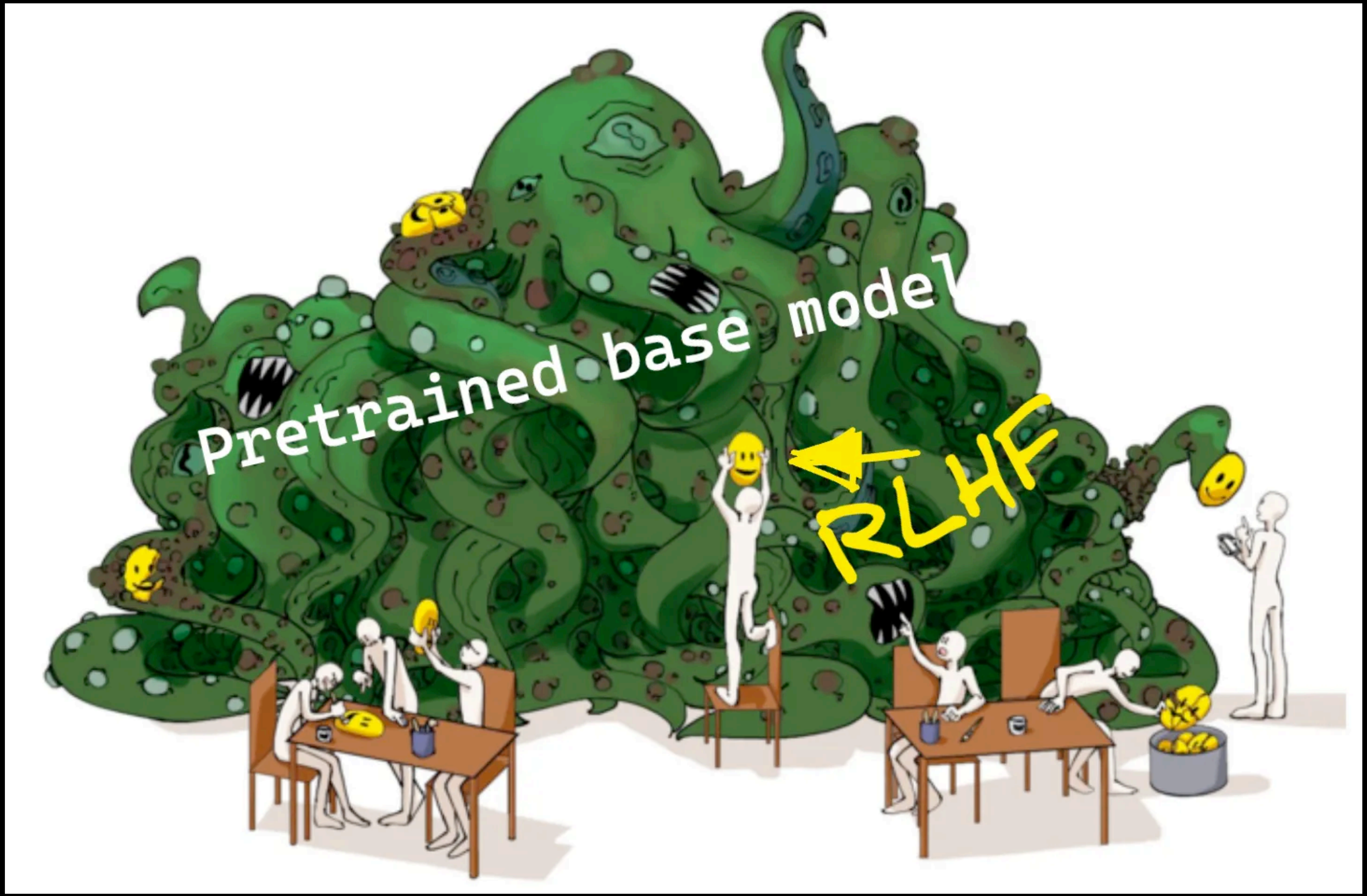
Advantage function: tells us if the actions were better than expected. If positive, increase $\pi_\theta(a \mid s)$, else decrease

Knob controlling how much we penalize deviations from original policy

How much we deviate from original policy

Challenges in RLHF

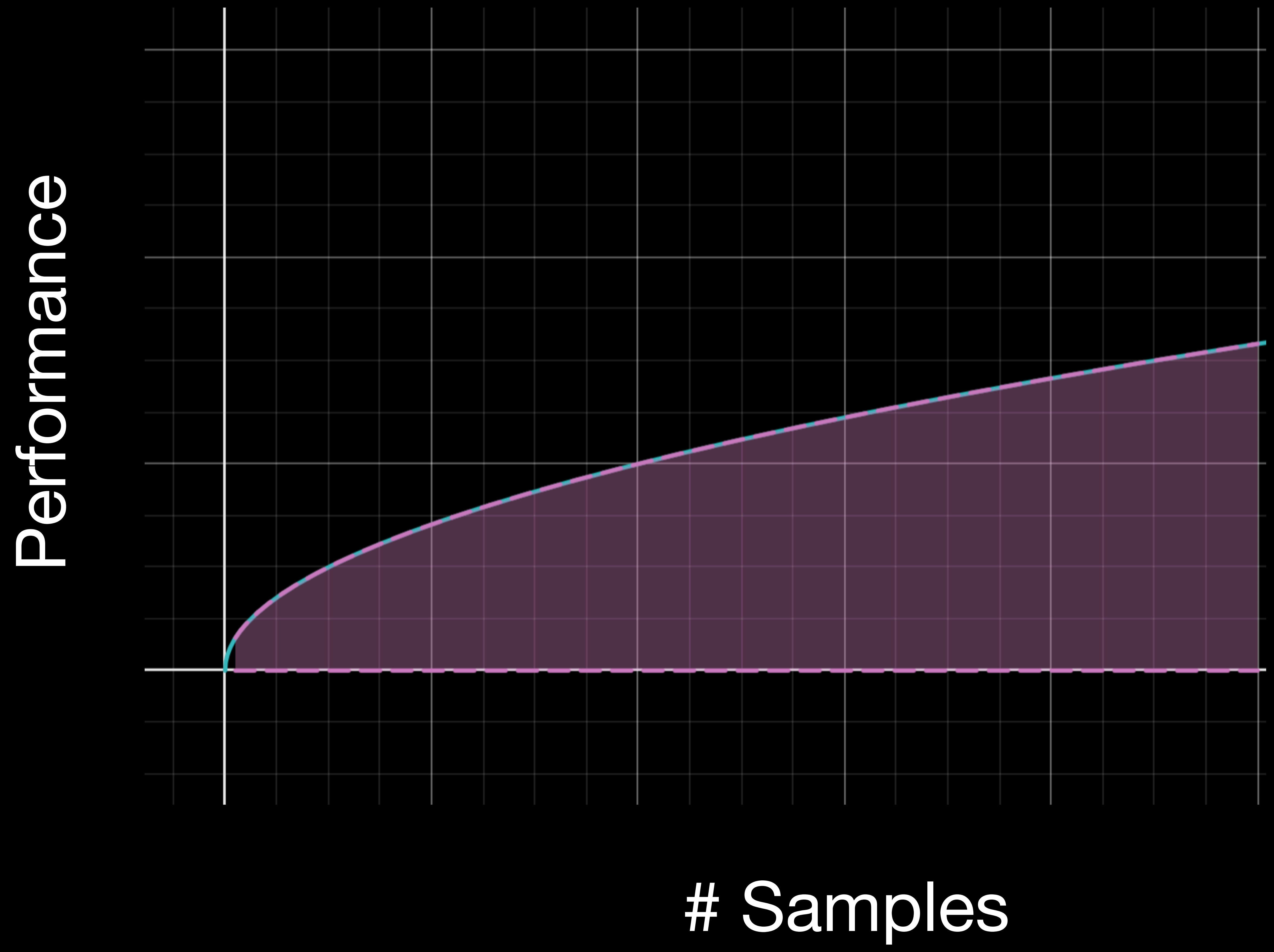
- Sometimes RLHF is just putting a mask on a nasty mess



https://www.latent.space/p/rlhf-201

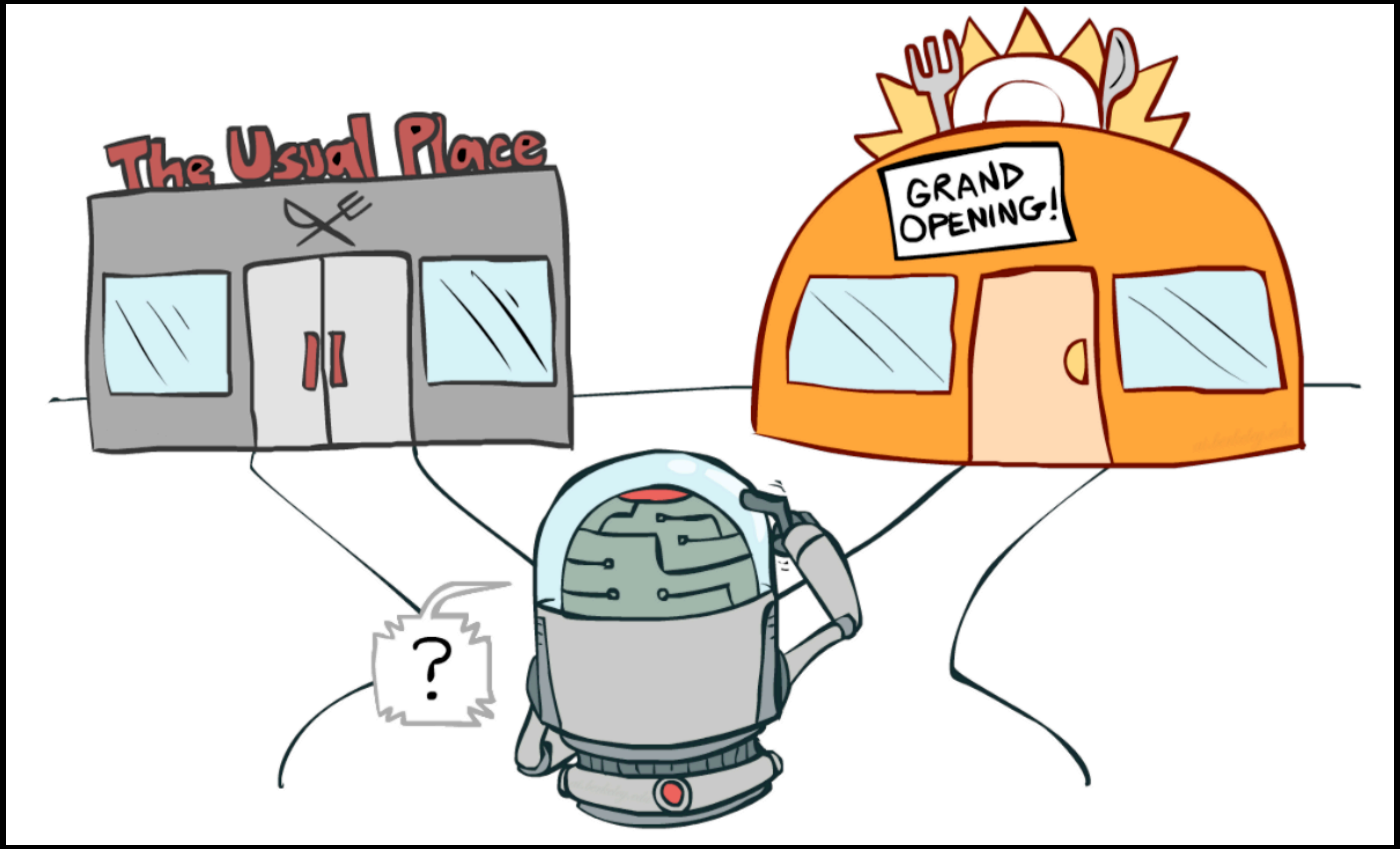
Challenges in RLHF

- **Sample efficiency:** RL algorithms typically require more observations than supervised learning



Challenges in RLHF

- **Exploration vs. exploitation:** How do we balance trying new actions (exploration) while exploiting what we know works?



https://huggingface.co/learn/deep-rl-course/unit1/exp-exp-tradeoff

Want to Learn More About RL at CU?

- Decision Making Under Uncertainty — ASEN 5264
  - Zachary Sunberg — https://github.com/CU-ADCL/CU-DMU-Materials
  - Typically offered Spring
- Deep Reinforcement Learning — CSCI 7000
  - Alessandro Roncone
  - Typically offered Fall

Overview

- Tool Use
- Instruction Tuning
- RLHF
- Parameter-Efficient Methods for Fine-Tuning (PEFT)
- Demo

How Can Academia Keep Up?

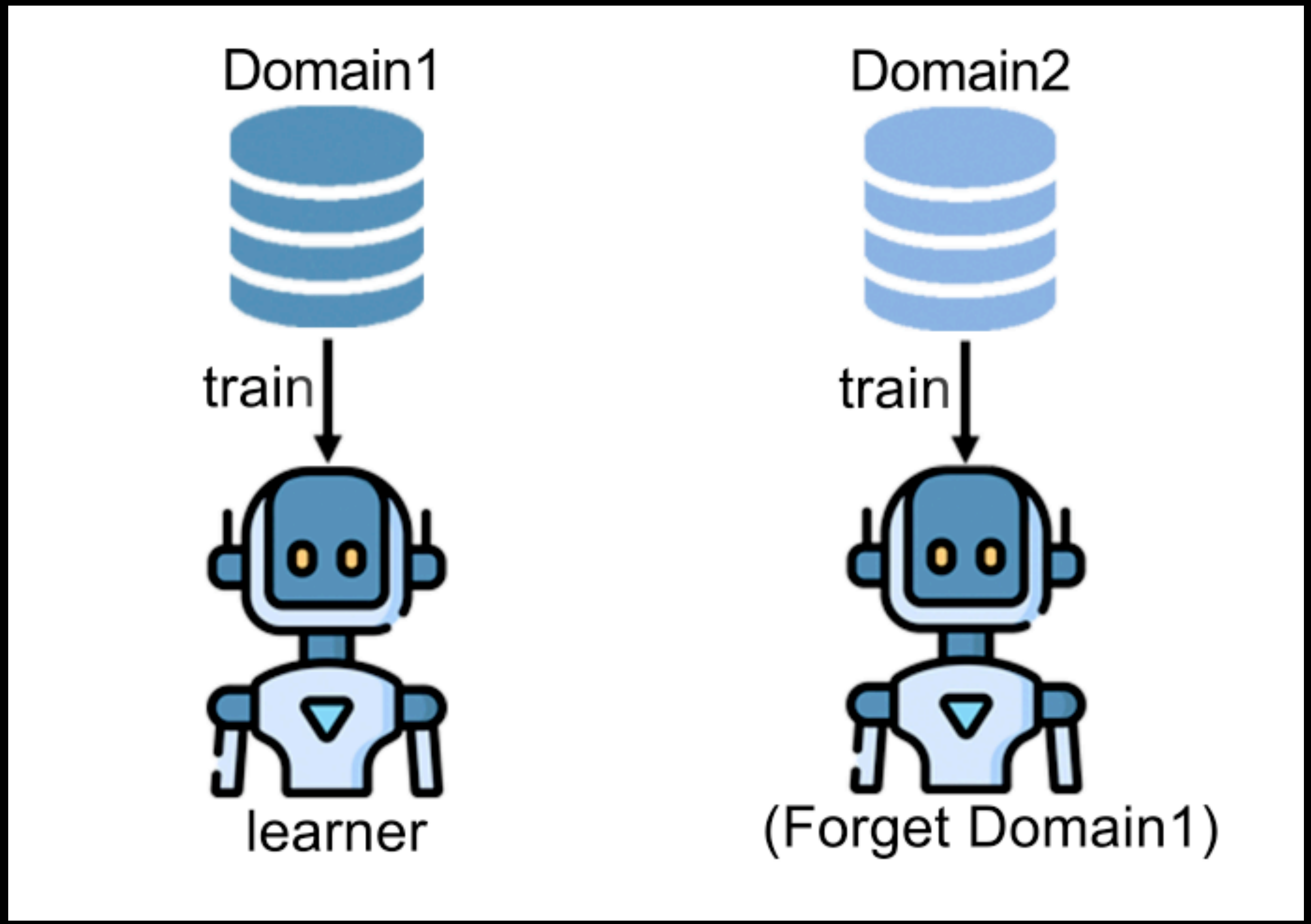
- Most large models come from industry labs with large amounts of resources
- How can we take advantage of these models?

Can’t We Just Use Transfer Learning?

- Yes… If we don’t care about retaining previous knowledge
- No… If the models are large or you are GPU poor (e.g., grad students)

Can’t We Just Use Transfer Learning?

- Yes… If we don’t care about retaining previous knowledge
- **Catastrophic forgetting:** A model “forgets” old knowledge as new knowledge is learned



https://cobusgreyling.medium.com/catastrophic-forgetting-in-llms-bf345760e6e2

Can’t We Just Use Transfer Learning?

- No… If the models are large or you are GPU poor (e.g., grad students)
- What gets stored in memory when training models?

Can't We Just Use Transfer Learning?

- Need to store model weights, gradients, optimizer states, etc.
- Consider model with 1B parameters, AdamW optimizer, 32-bit precision
- How much GPU VRAM do you think we need?

(a) 12GB (b) 48GB (c) 200GB (d) 100GB

Can't We Just Use Transfer Learning?

- $\left(\underbrace{4 \text{ bytes}}_{\text{Gradients}} + \underbrace{4 \text{ bytes}}_{\text{Model}} + \underbrace{12 \text{ bytes}}_{\text{Optimizer}} \right) \cdot 1\text{e}^9 \approx \text{2B bytes} \approx 200GB$
- Also need memory for images/text/audio!

Can’t We Just Use Transfer Learning?

GPU	Memory	Cost (2/2024)	(Cloud) Machines
T40 / K80	24GB	$150	Google Colab, AWS p2.*
V100	32GB	$2,500	Google Colab
A100	40GB or 80GB	$8,000/$16,000	Google Colab, AWS p3.*
H100	80GB	$44,000	AWS p4.*
6000 Ada, L40	48GB	$8000	N/A
Mac M*	Same as CPU	$2000	N/A

https://phontron.com/class/anlp2024/assets/slides/anlp-08-instructiontuning.pdf

PEFT — Motivation

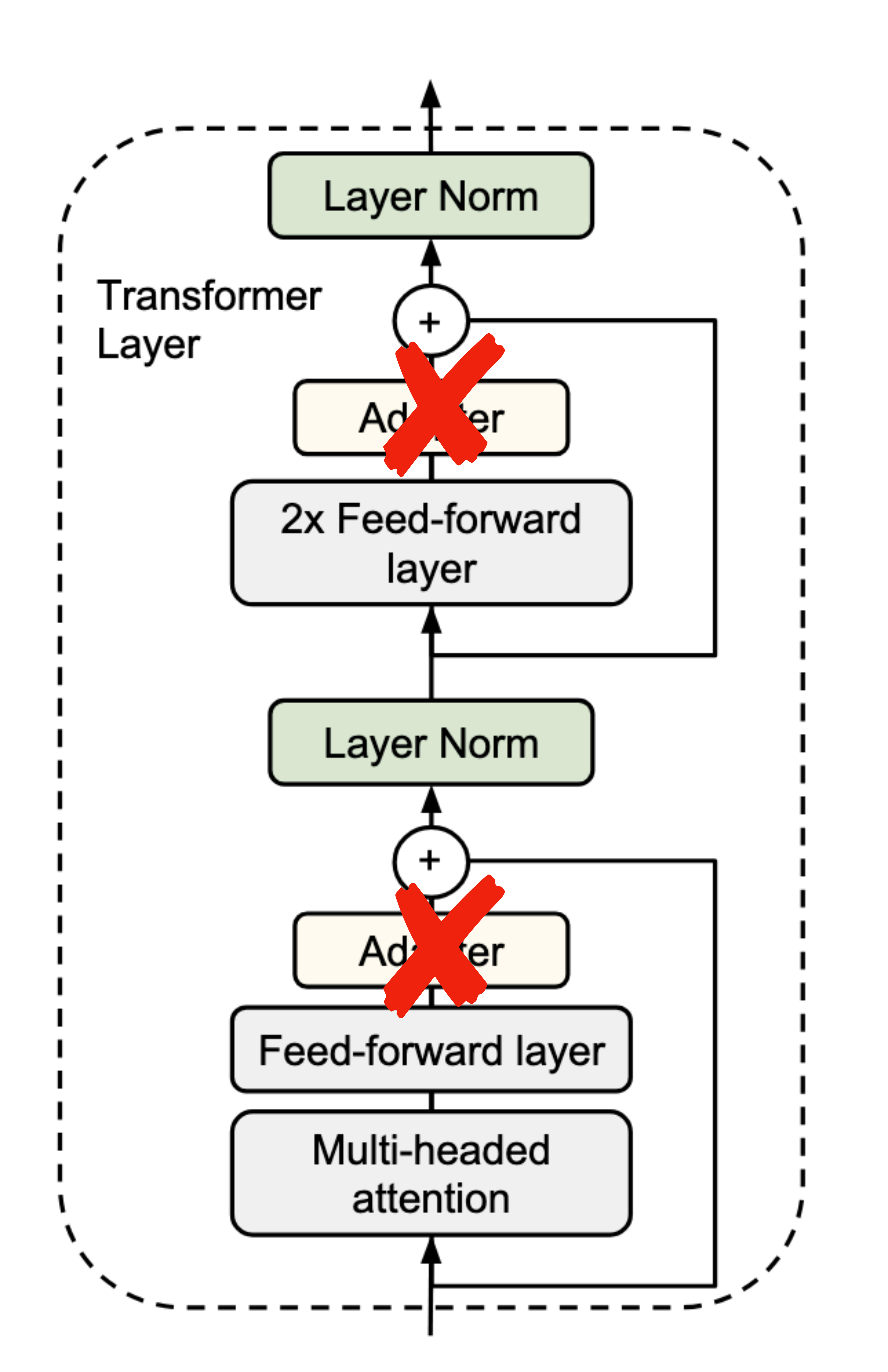
- For **N** specific tasks, need to fine-tune **N** different models
- What if we just train a few specialized parameters?

PEFT — Motivation

- PEFT — Parameter Efficient Fine-Tuning
- Adapt pre-trained models to new tasks by updating only a small subset of parameters to reduce computational and storage costs while maintaining performance
- Less parameters == less memory == less GPUS == happy grad students!

PEFT — Benefits

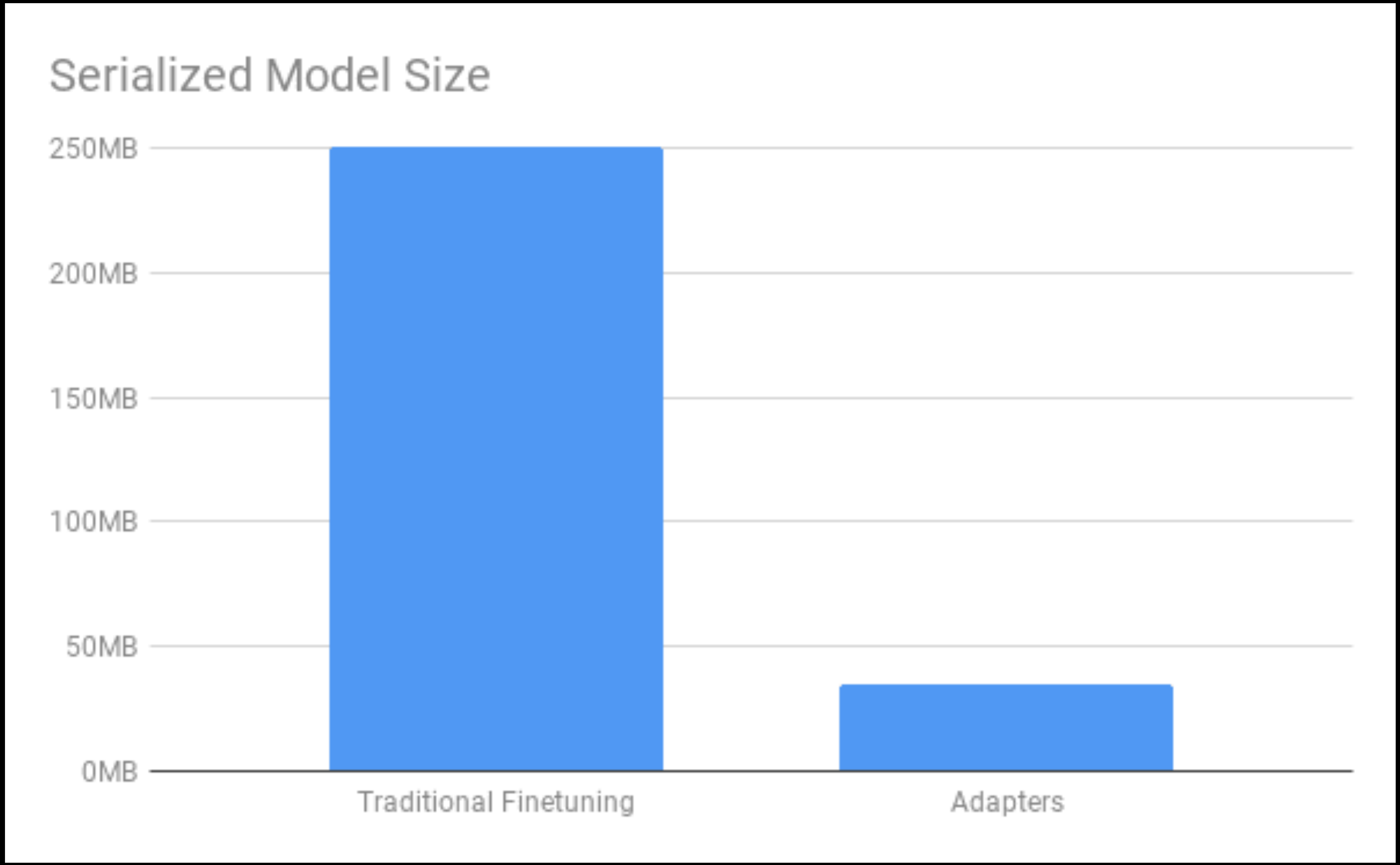
- Prevents catastrophic forgetting! Want to perform your original task? Drop the modified parameters for the original ones!



[Houlsby, MLR19]

PEFT — Motivation

- Cheaper for on device applications as we do not need to store **N** copies of the model!

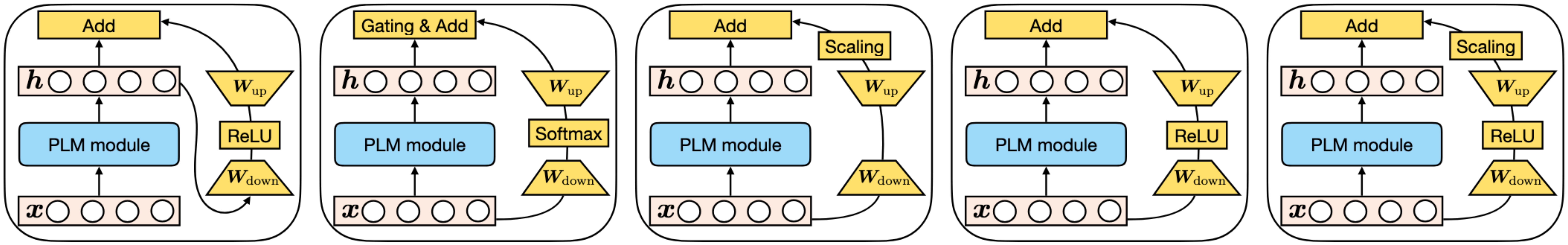


https://indicodata.ai/blog/adapters-faster-prediction-with-fewer-parameters/

PEFT — Motivation

- **Analogy:** Hands serve as our “base” model and tools as adaptations for new tasks

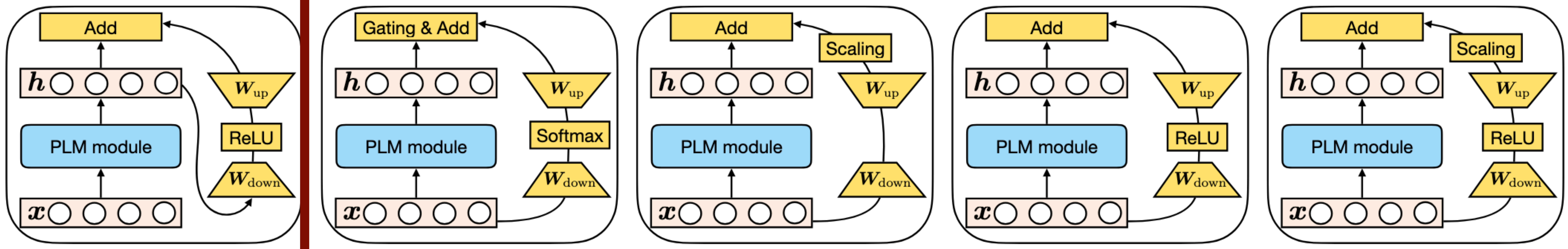
PEFT - Overview



(a) Adapter (b) Prefix Tuning (c) LoRA (d) Parallel Adapter (e) Scaled PA

[He, ICLR22]

PEFT - Overview

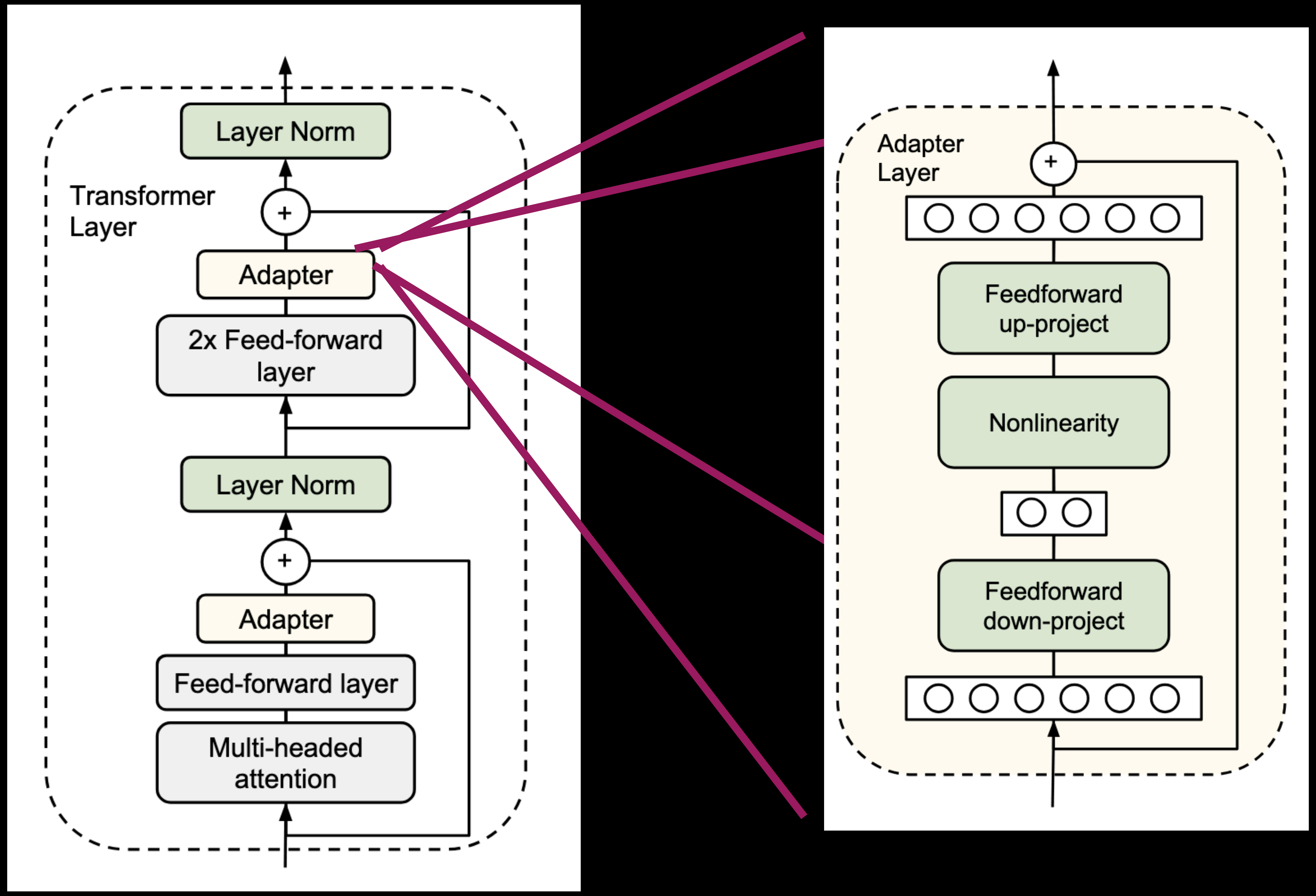


(a) Adapter (b) Prefix Tuning (c) LoRA (d) Parallel Adapter (e) Scaled PA

[He, ICLR22]

PEFT — Adapters

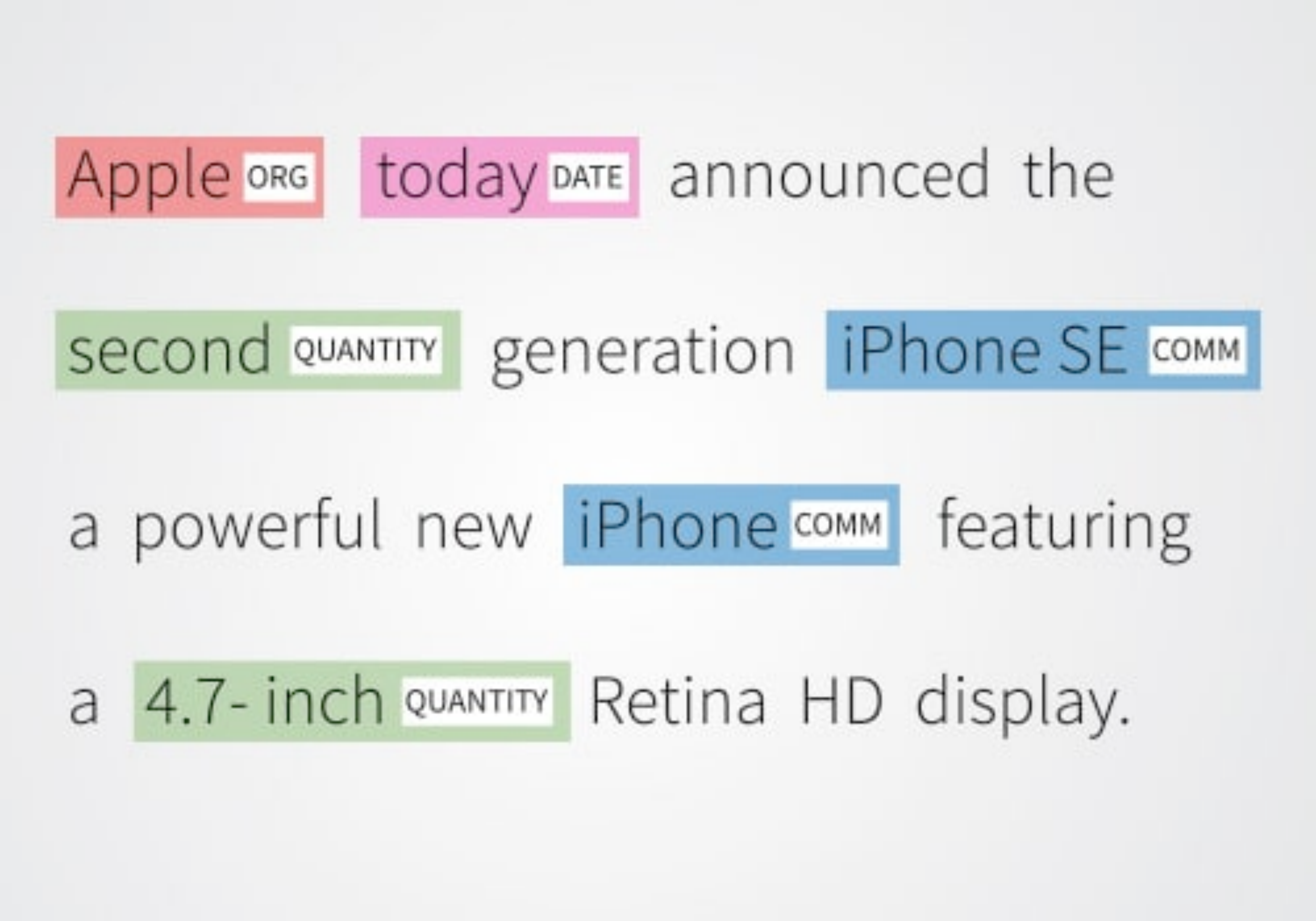
- Adapters introduce **task-specific** trainable parameters (3-5% additional) into transformer blocks
- Freeze gradient updates to the original model parameters



[Houlsby, MLR19]

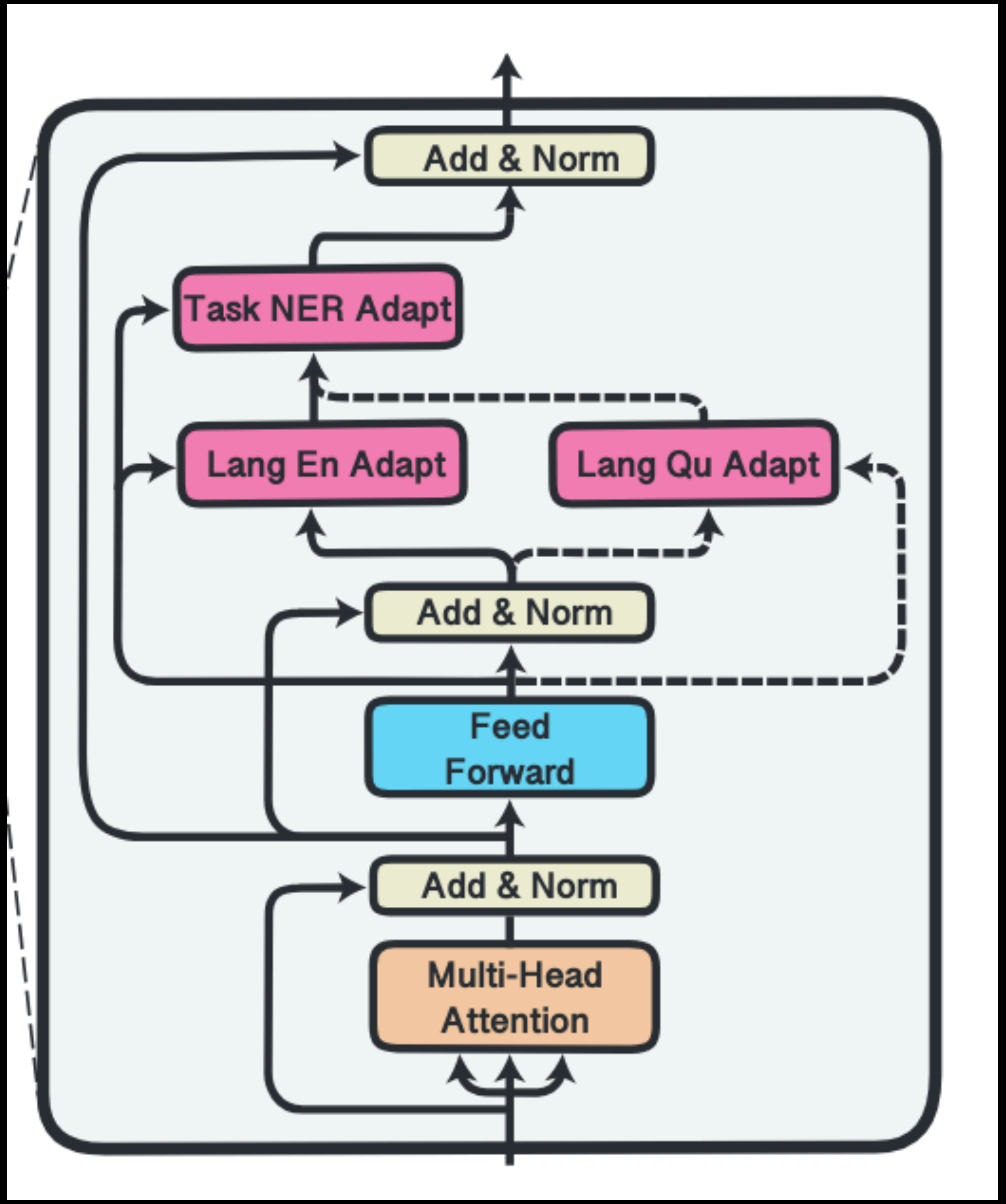
PEFT — Adapters

- Consider a generalist model (e.g., BERT) trained on English
- What if I want to perform NER on a low-resource language (e.g., Quechua)?



PEFT — Adapters

- Each adapter corresponds to some task (e.g., language translation, task translation)
- Let's combine them!



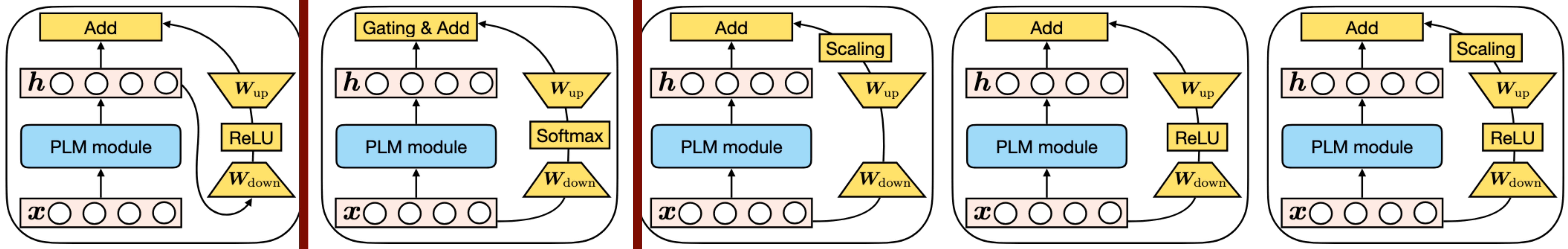
Source adapters (solid)
get replaced by target adapters
(Dashed) at inference time

[Pfieffer, ACL20]

PEFT — Adapters

- Downsides?
- Requires extra parameters (3-5%) — extra inference time
- Need separate modules for each task — no multi-task learning
- Need additional storage for each module

PEFT - Overview



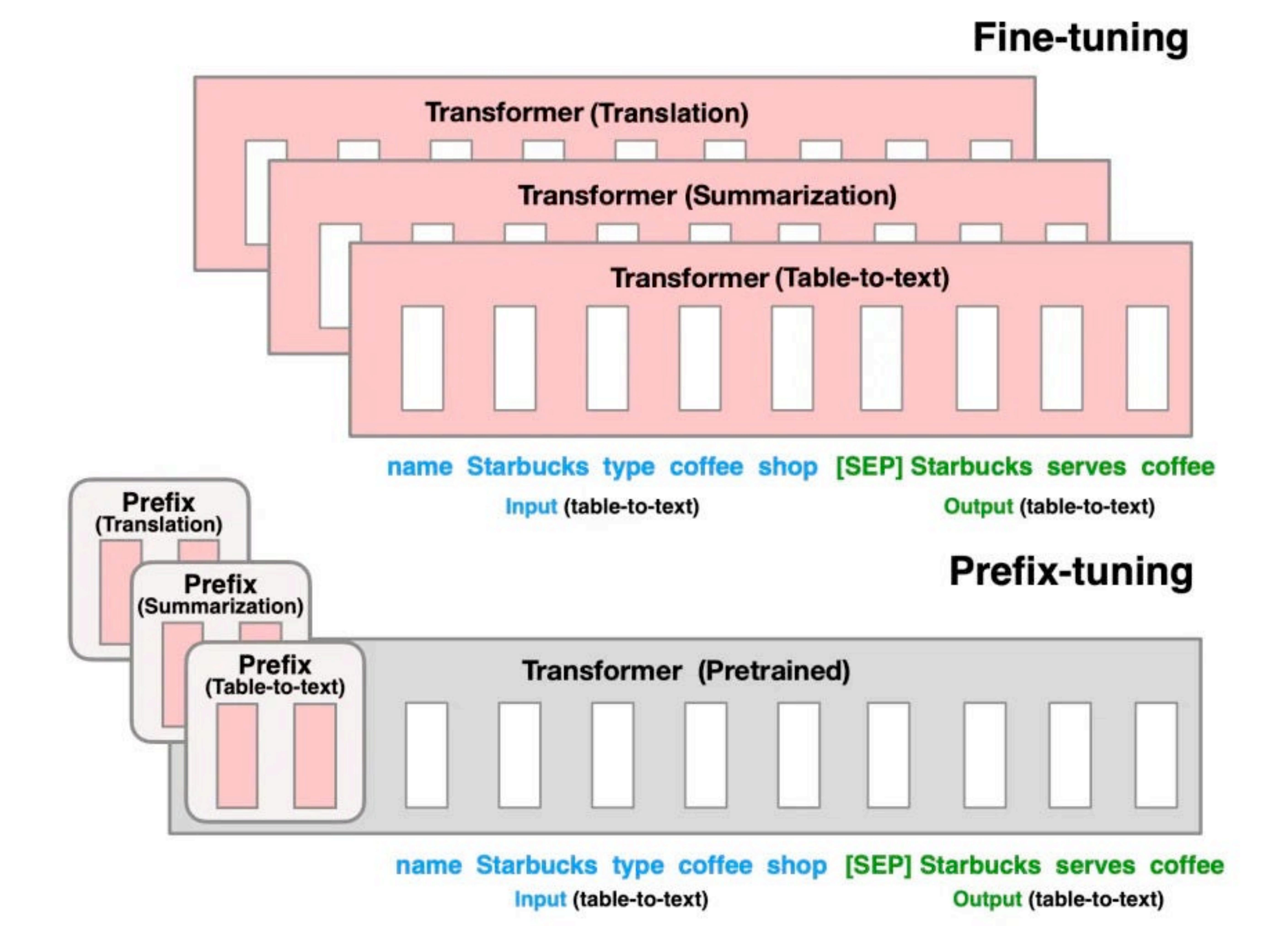
(a) Adapter (b) Prefix Tuning (c) LoRA (d) Parallel Adapter (e) Scaled PA

[He, ICLR22]

PEFT — Prefix Tuning

- Introducing **Soft Prompts**: learnable, task specific vectors

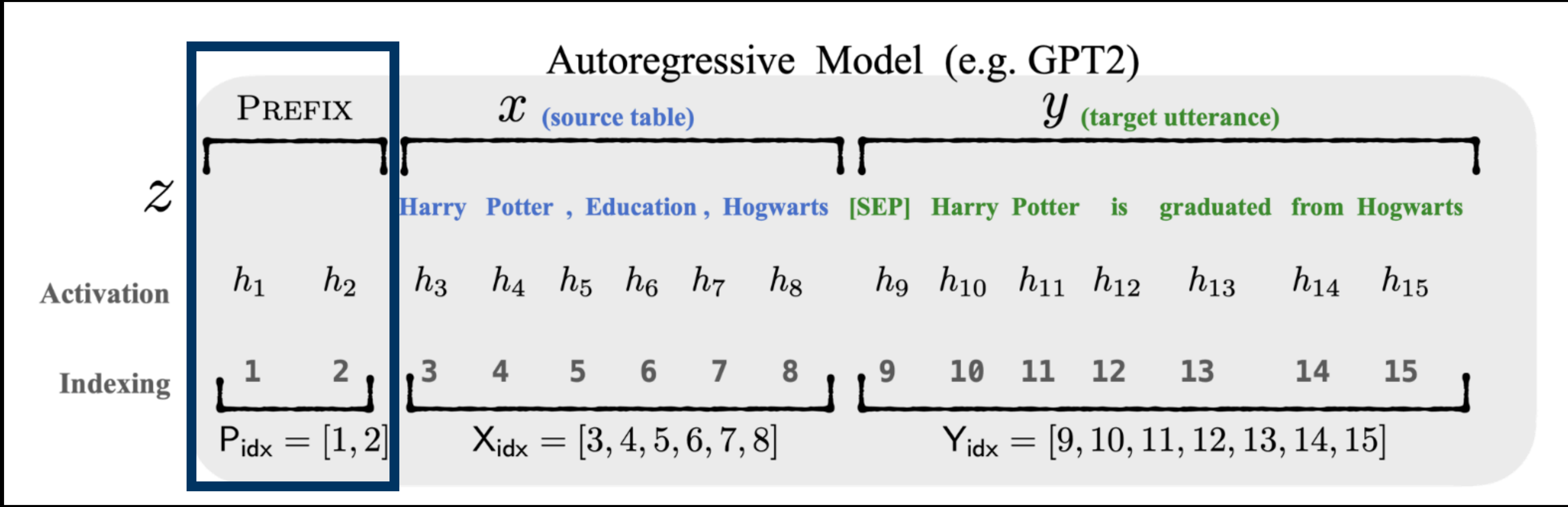
- Instead of giving the model a text prompt, prepend a **task specific vector** to the model inputs and freeze the rest of the model!



[Li, ACL21]

PEFT — Prefix Tuning

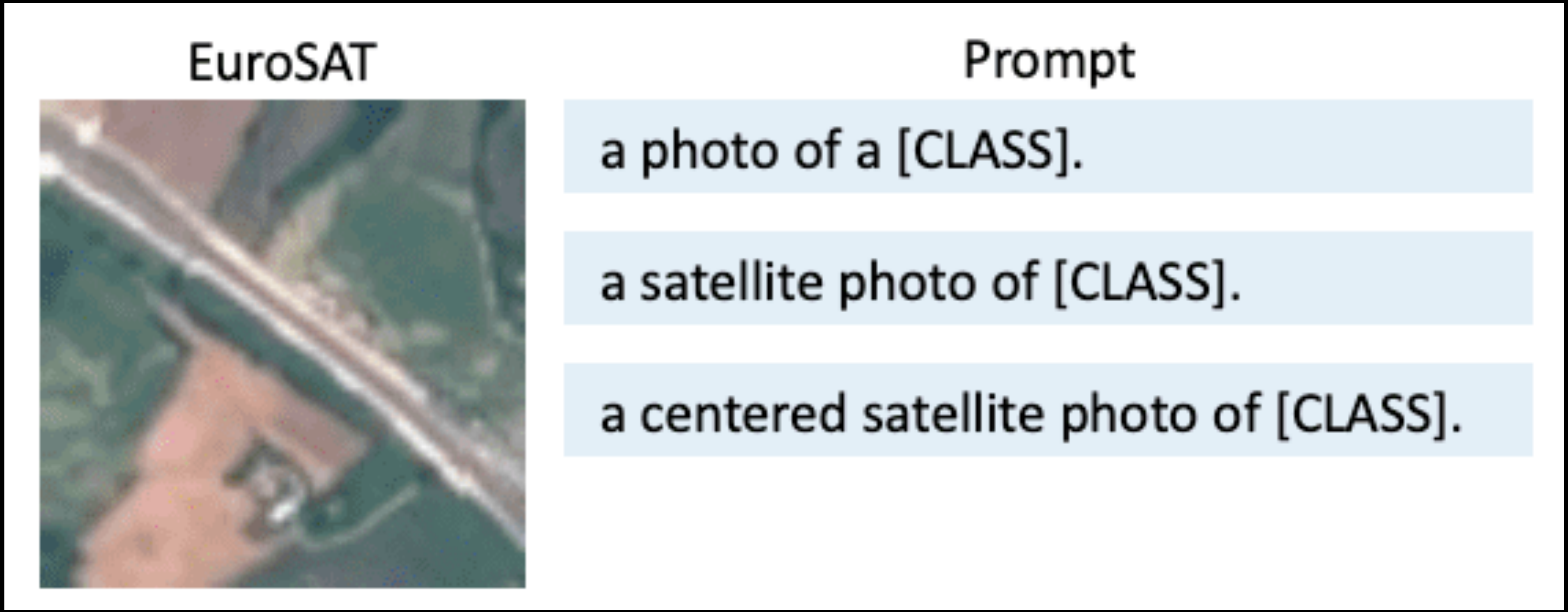
- **Key Idea:** First 2 hidden states (h_i) are the learnable vectors (i.e., rows from a matrix P_θ)
- Only need to update P_θ during training!



[Li, ACL21]

PEFT — Prefix Tuning: CoOp

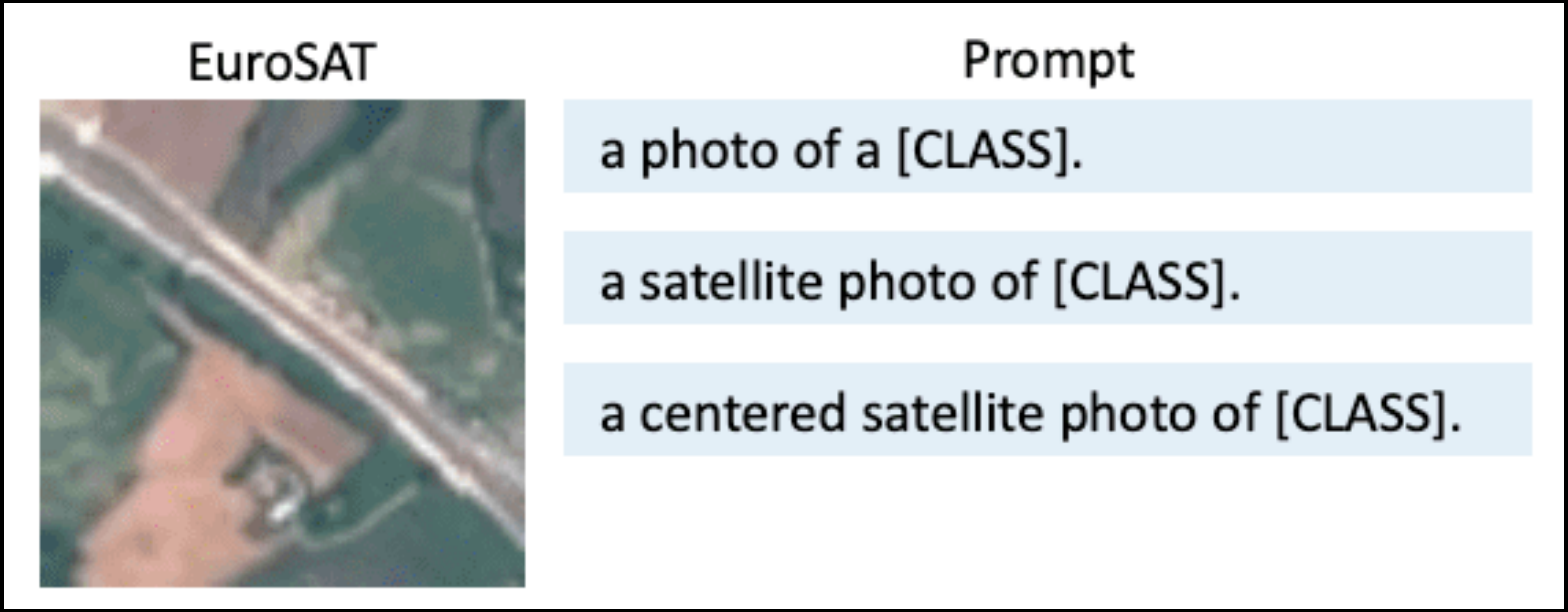
- Prefix tuning works for vision too!
- Consider CLIP; How well would this do on satellite imagery? Why?



[Zhou, CVPR22]

PEFT — Prefix Tuning: CoOp

- Domain shift from original training data! Prompt engineering only gets so far!



[Zhou, CVPR22]

PEFT — Prefix Tuning: CoOp

- **Co**ntext **Op**timization — optimize for the context the image appears in
- **Key Idea:** Learn the **context** of the image with continuous vectors
- Replace prompts with soft prompts (“a photo of…” —> [V])

EuroSAT

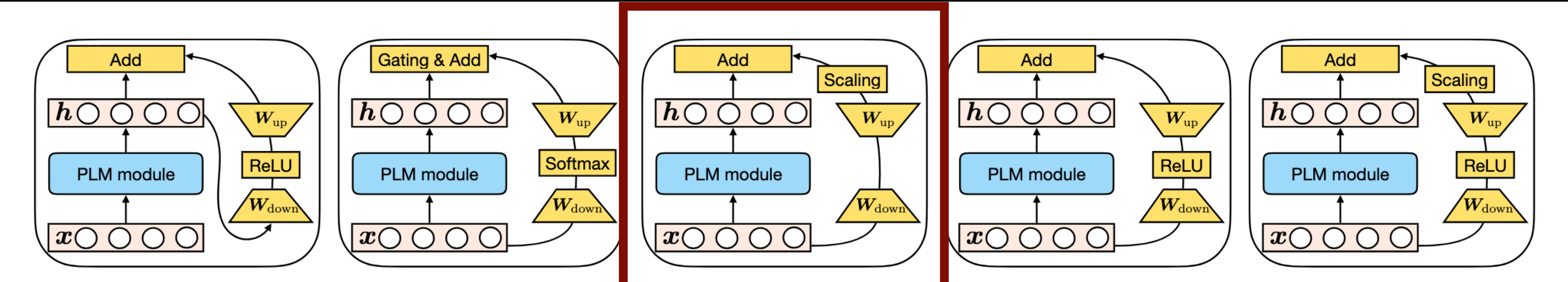
Prompt	Accuracy
a photo of a [CLASS].	22.30
a satellite photo of [CLASS].	31.12
a centered satellite photo of [CLASS].	31.53
$[V]_1 [V]_2 \dots [V]_M$ [CLASS].	**81.60**

[Zhou, CVPR22]

PEFT — Prefix Tuning

- Downsides of prefix tuning
  - Vectors are **task specific** — need to train a new one for each task
  - Prompts are only affected, what if we need to modify the vision features?

PEFT - Overview



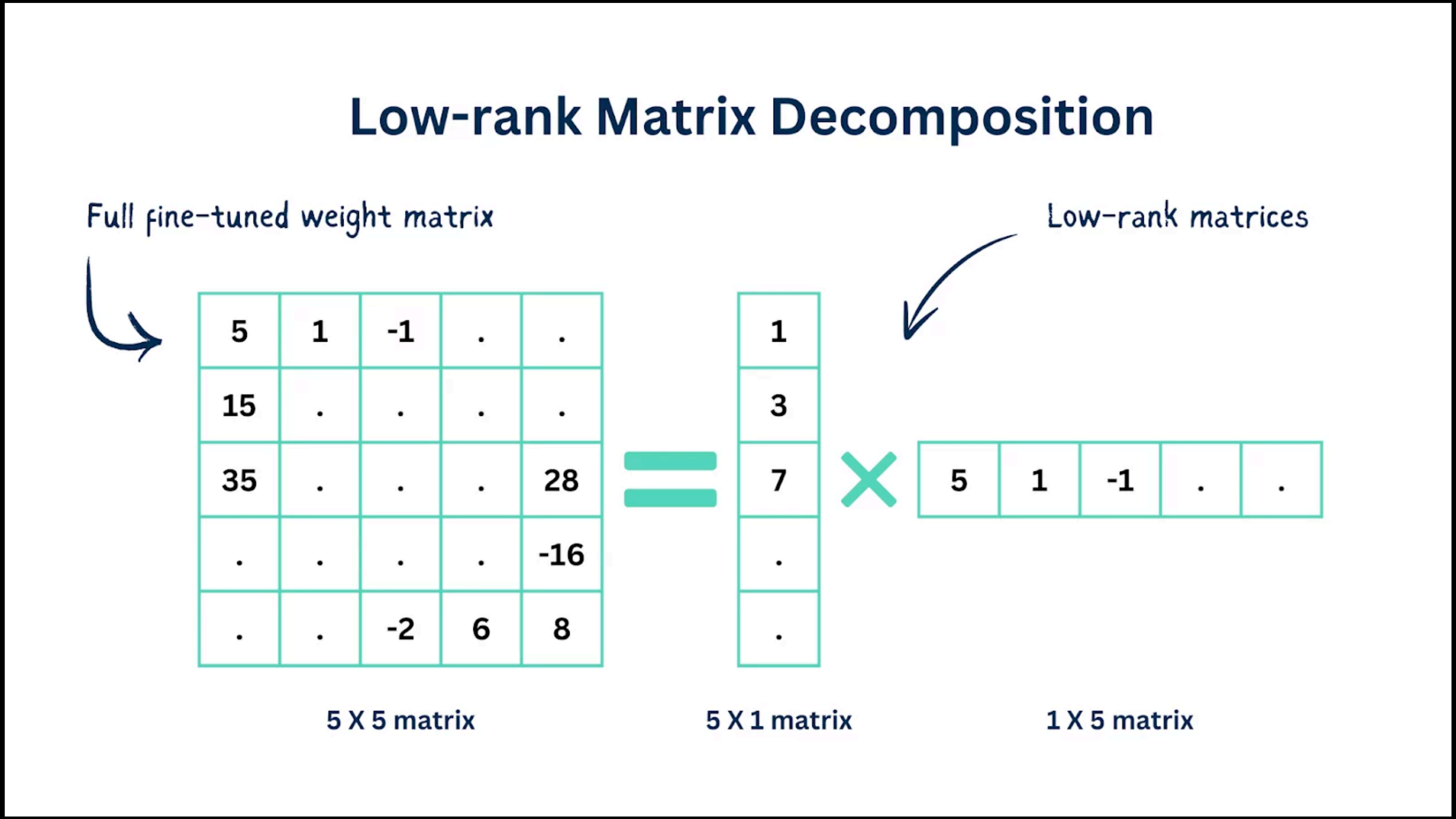
(a) Adapter (b) Prefix Tuning (c) LoRA (d) Parallel Adapter (e) Scaled PA

[He, ICLR22]

PEFT — LoRA

- Let's learn **L**ow **R**ank **A**d**a**ptations of our model weights!

- **Key Idea:** For specific linear layers (e.g., attention weights), learn 2 matrices of rank r during training; freeze all other parameters

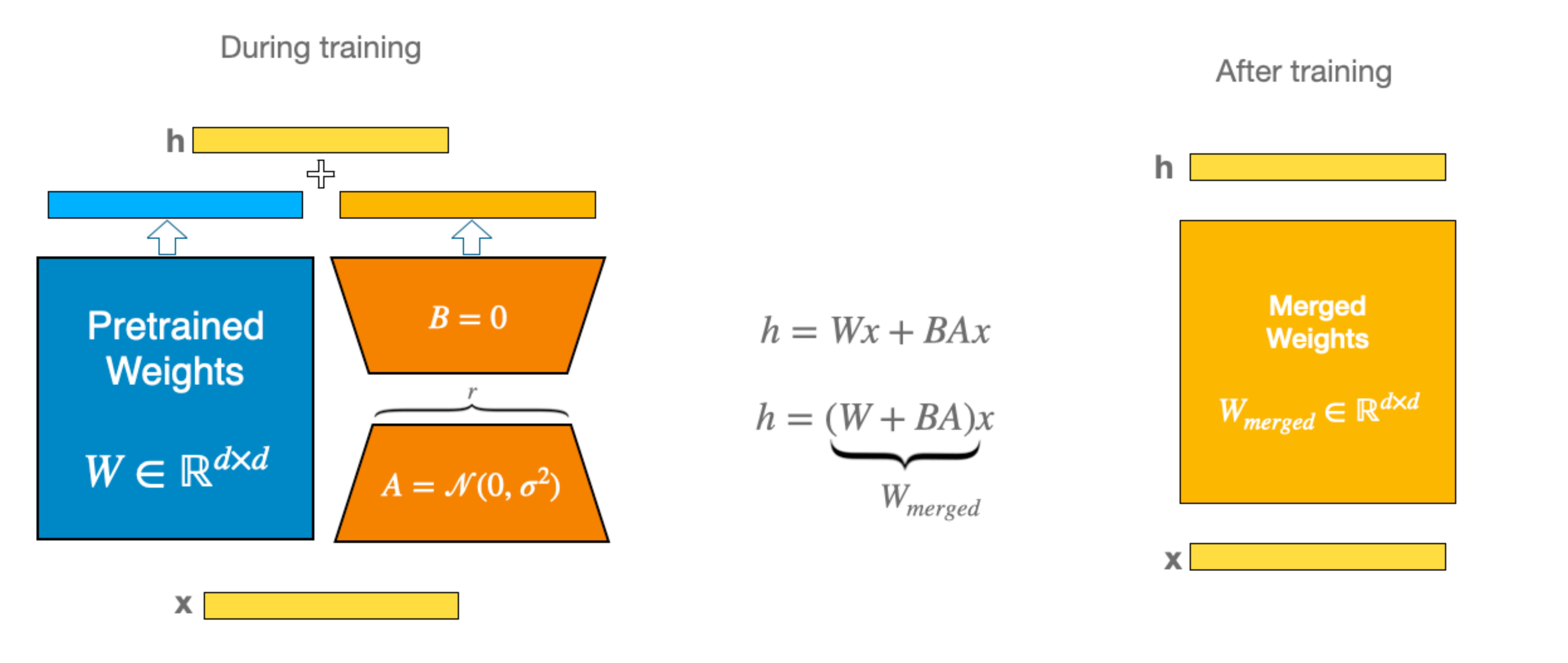


[Hu, ICLR22]

https://www.aporia.com/learn/low-rank-adaptation-lora/

PEFT — LoRA

- At inference: merge newly learned weights with OG weights!



[Hu, ICLR22]

https://huggingface.co/docs/peft/main/en/conceptual_guides/lora

PEFT — LoRA

- Faster at inference time since no new parameters

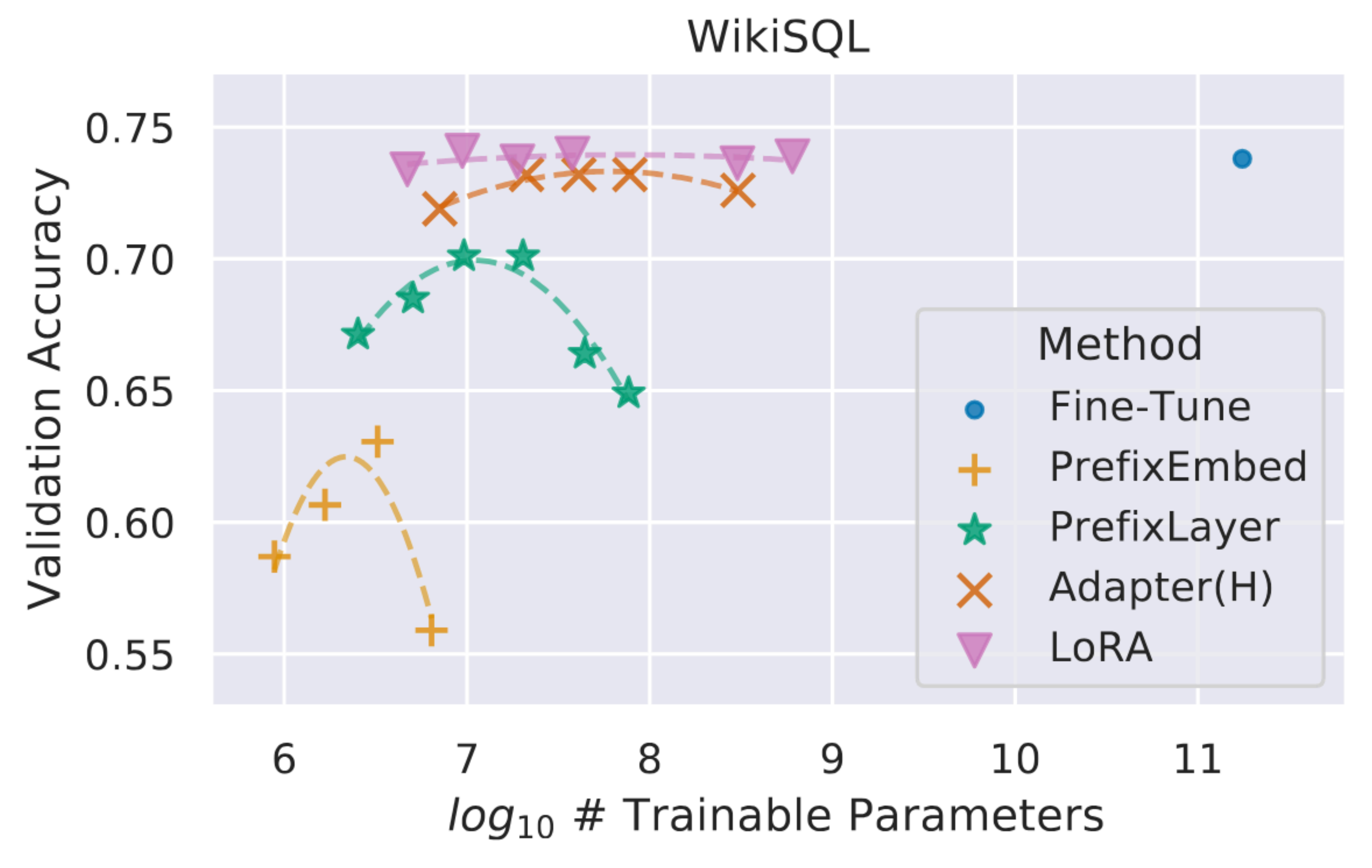
Batch Size Sequence Length $\|\Theta\|$	32 512 0.5M	16 256 11M	1 128 11M
Fine-Tune/LoRA	1449.4±0.8	338.0±0.6	19.8±2.7
Adapter^L	1482.0±1.0 (+2.2%)	354.8±0.5 (+5.0%)	23.9±2.1 (+20.7%)
Adapter^H	1492.2±1.0 (+3.0%)	366.3±0.5 (+8.4%)	25.8±2.2 (+30.3%)

Inference latency

[Hu, ICLR22]

PEFT — LoRA

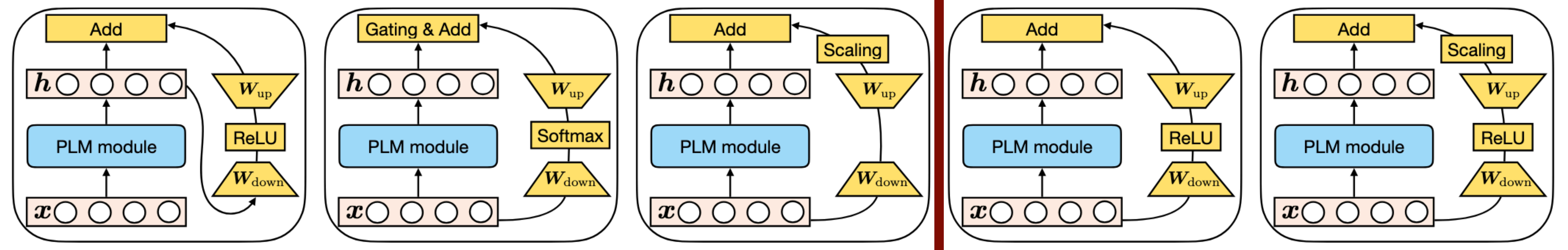
- Better performance on target tasks!



[Hu, ICLR22]

PEFT — Wrap Up

- Most PEFT methods have only slightly different designs!
- How do we know which one to use?



(a) Adapter (b) Prefix Tuning (c) LoRA (d) Parallel Adapter (e) Scaled PA

[He, ICLR22]

PEFT — Wrap Up

- No additional latency?
- Classification task?
- Complex task + low computational budget?
- Complex task + high computational budget?

- LoRA
- Prefix Tuning
- Adapters

https://phontron.com/class/anlp2024/assets/slides/anlp-08-instructiontuning.pdf

PEFT — Wrap Up

- No additional latency?
  - LoRA
- Classification task?
- Complex task + low computational budget?
- Complex task + high computational budget?

https://phontron.com/class/anlp2024/assets/slides/anlp-08-instructiontuning.pdf

PEFT — Wrap Up

- No additional latency?
  - LoRA
- Classification task?
  - Likely doesn't matter
- Complex task + low computational budget?
- Complex task + high computational budget?

https://phontron.com/class/anlp2024/assets/slides/anlp-08-instructiontuning.pdf

PEFT — Wrap Up

- No additional latency?
  - LoRA
- Classification task?
  - Likely doesn't matter
- Complex task + low computational budget?
  - Prefix tuning
- Complex task + high computational budget?

https://phontron.com/class/anlp2024/assets/slides/anlp-08-instructiontuning.pdf

PEFT — Wrap Up

- No additional latency?
  - LoRA
- Classification task?
  - Likely doesn't matter
- Complex task + low computational budget?
  - Prefix tuning
- Complex task + high computational budget?
  - Adapters

https://phontron.com/class/anlp2024/assets/slides/anlp-08-instructiontuning.pdf

PEFT — Wrap Up

- Anecdote: LoRA is the de facto standard for tuning large models
- Adapters are also popular since extra parameters give more expressive power

Overview

- Tool use
- Instruction tuning
- RLHF
- Parameter-efficient methods for fine-tuning (PEFT)
- Demo