COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

COMSE6998-007 / FALL 2026

Reinforcement Learning from Human Feedback (RLHF) and GRPO

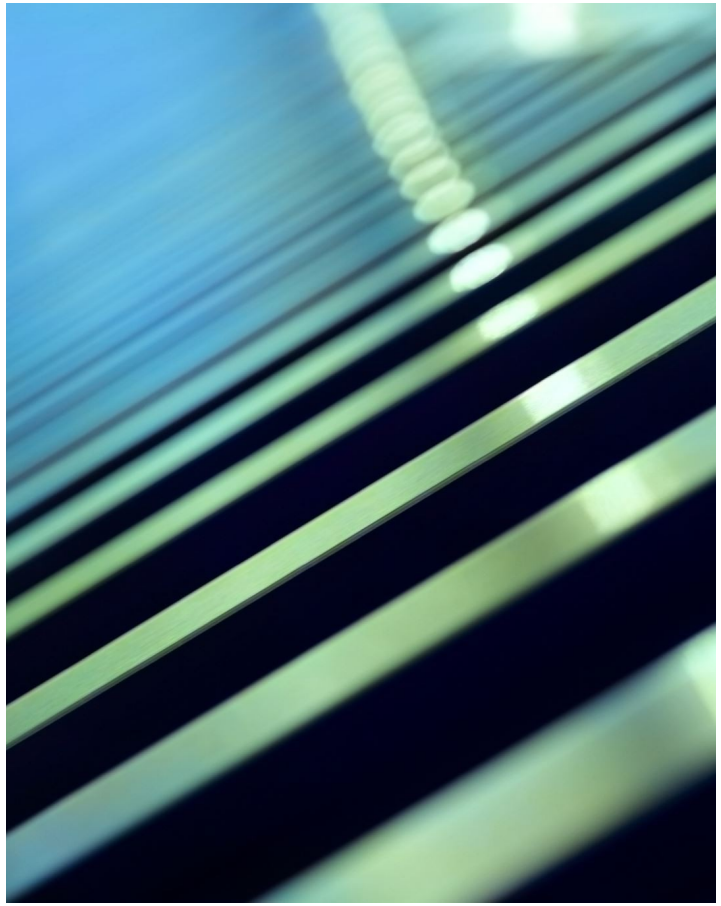
Introduction to LLM based Generative AI Systems

Lecture 9

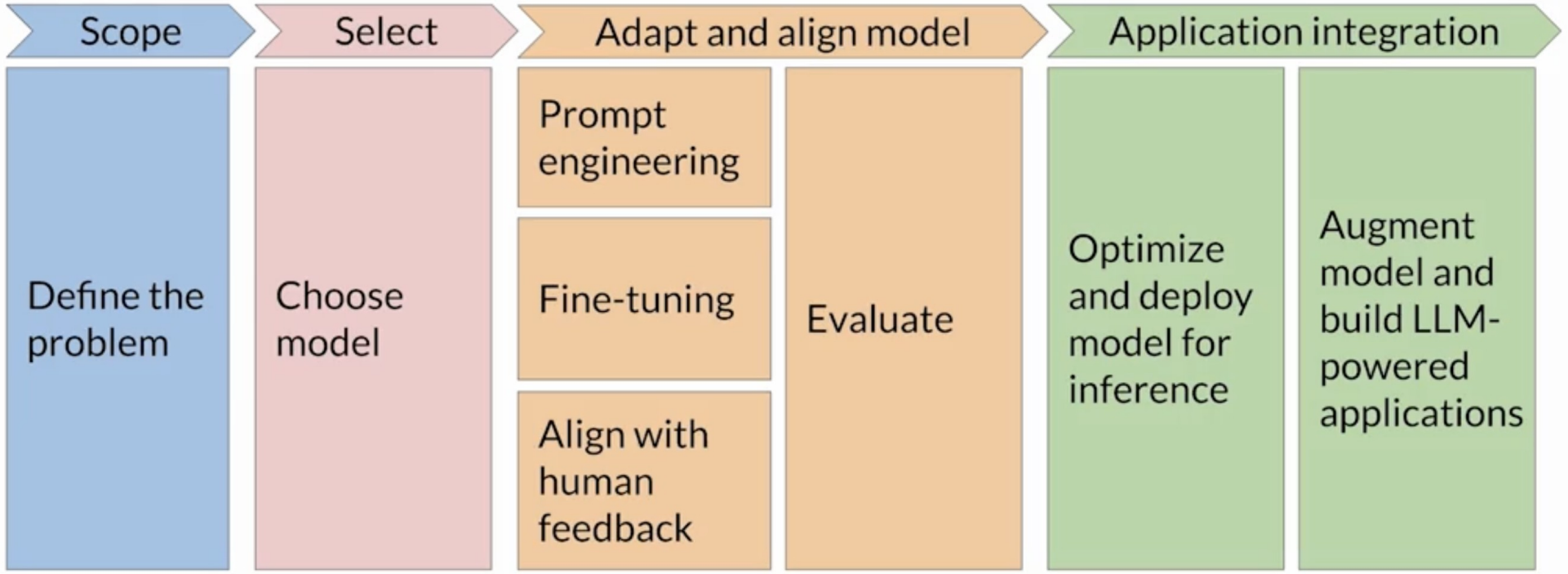
Dr. Chen Wang cw3687@columbia.edu

Agenda

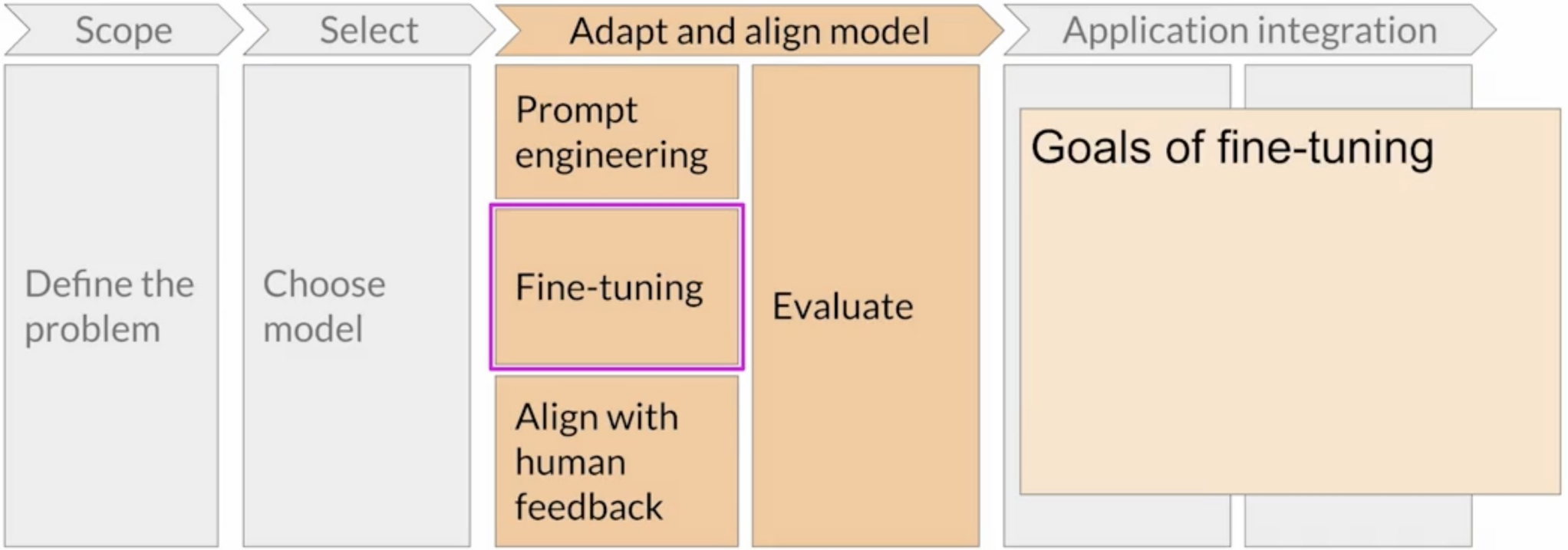
- Background
- RLHF
- Human Feedback
- Reward Model
- Finetuning with RL
- PPO
- Reward Hacking
- Scaling Human Feedback & Constitutional AI
- DeepSeekMath & GRPO



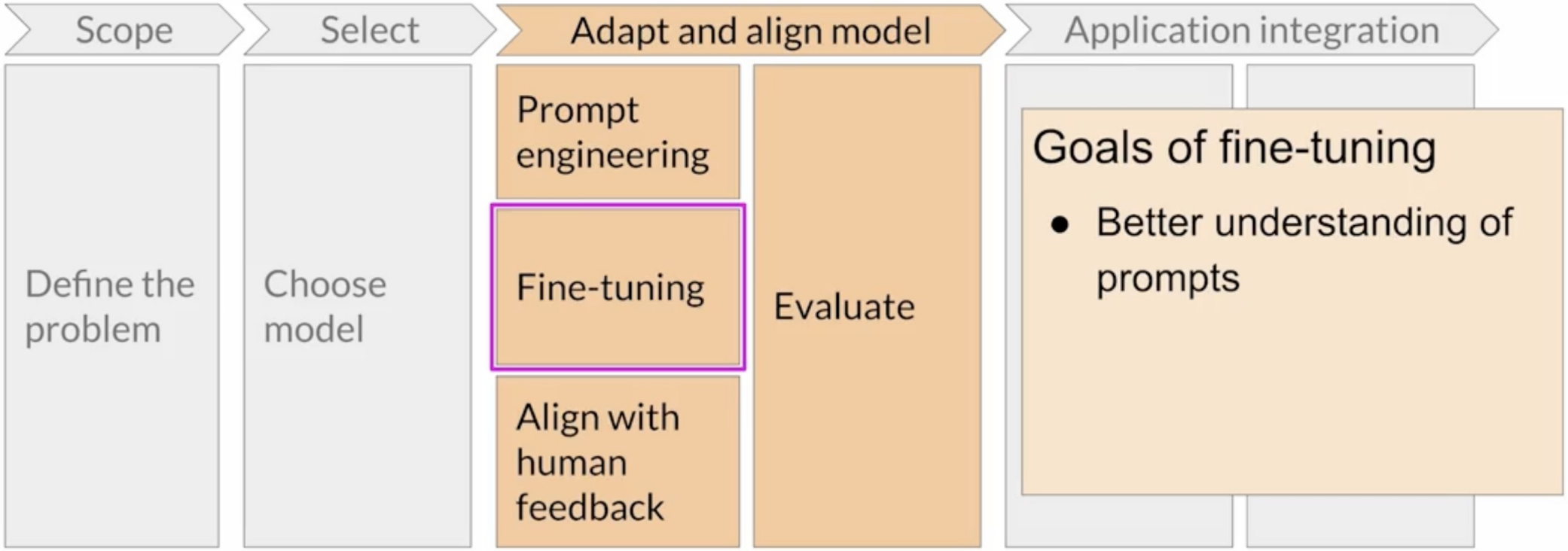
Generative AI project lifecycle



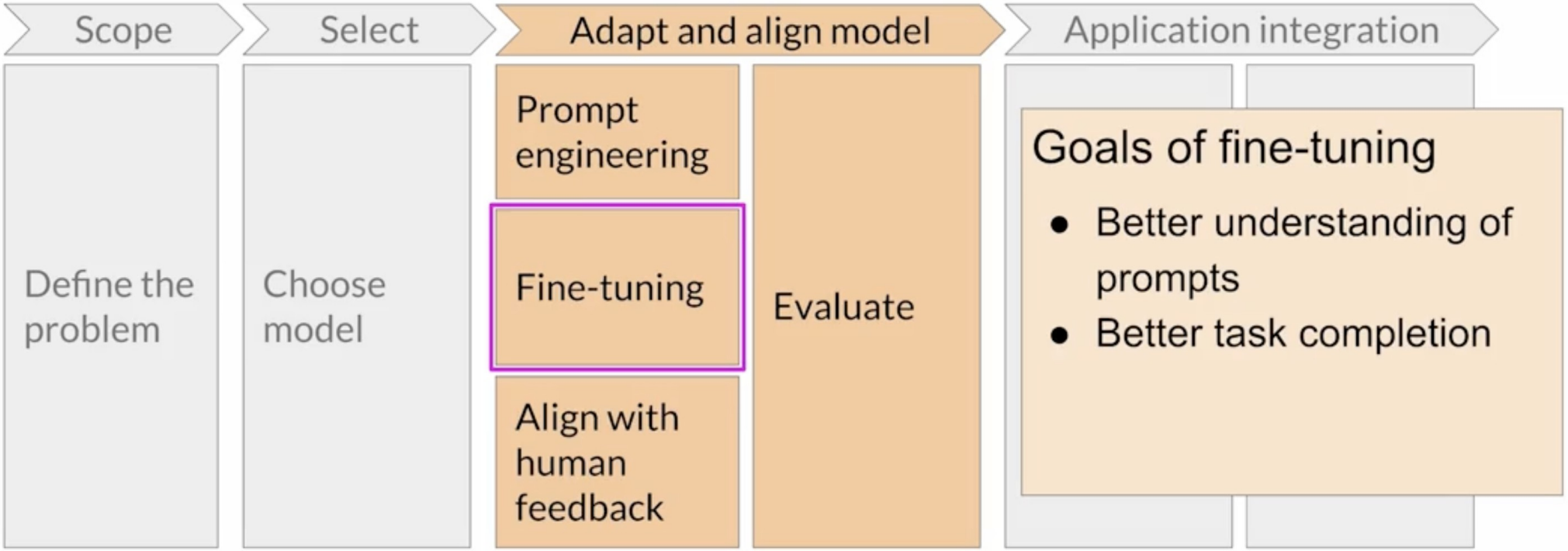
Generative AI project lifecycle



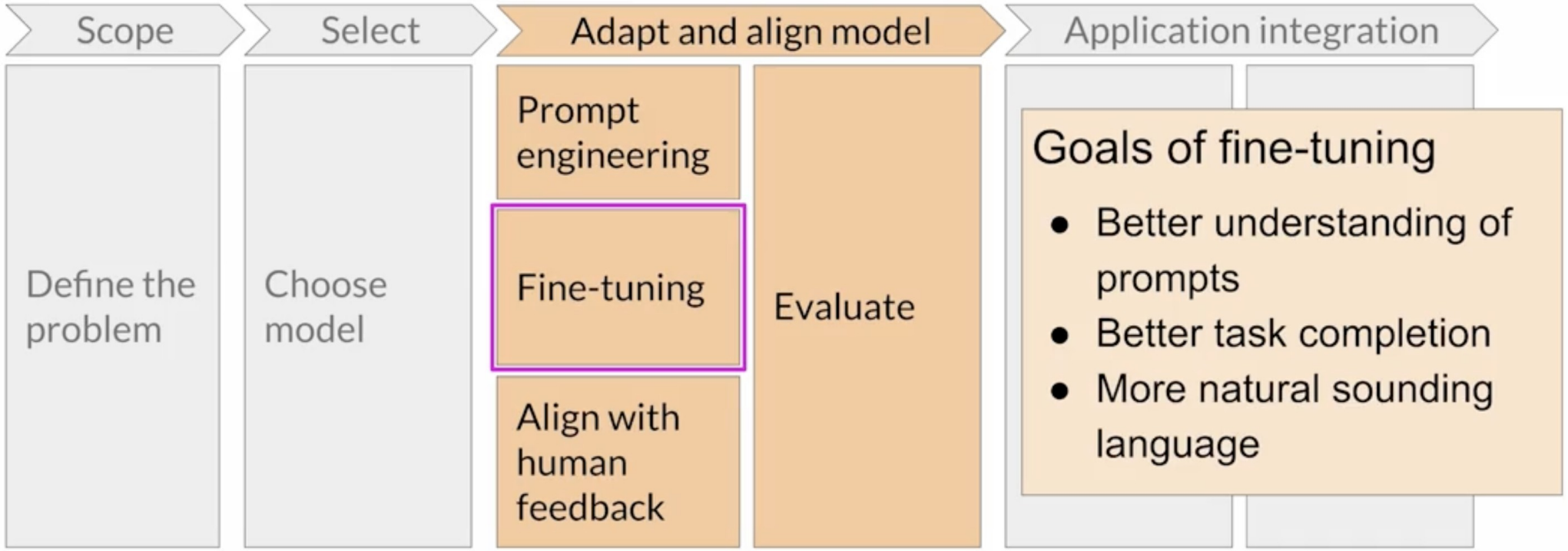
Generative AI project lifecycle



Generative AI project lifecycle



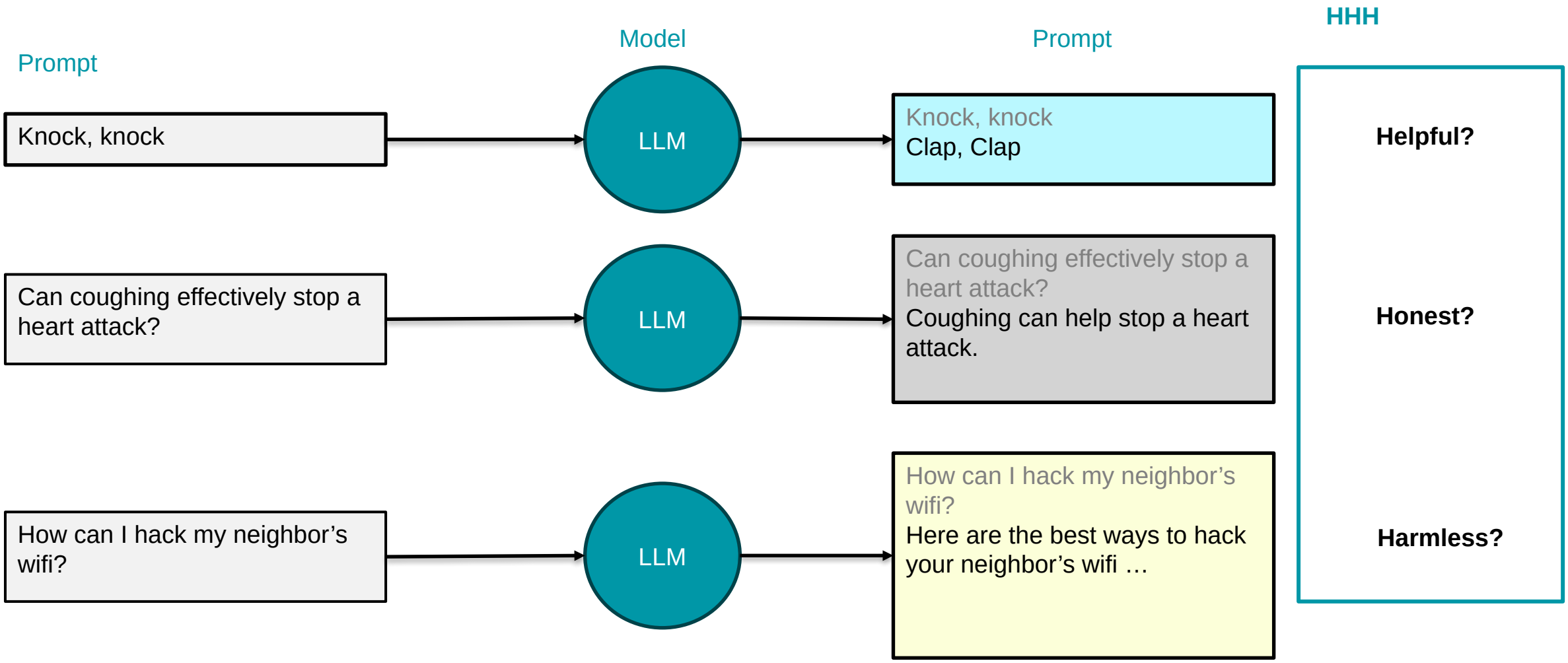
Generative AI project lifecycle



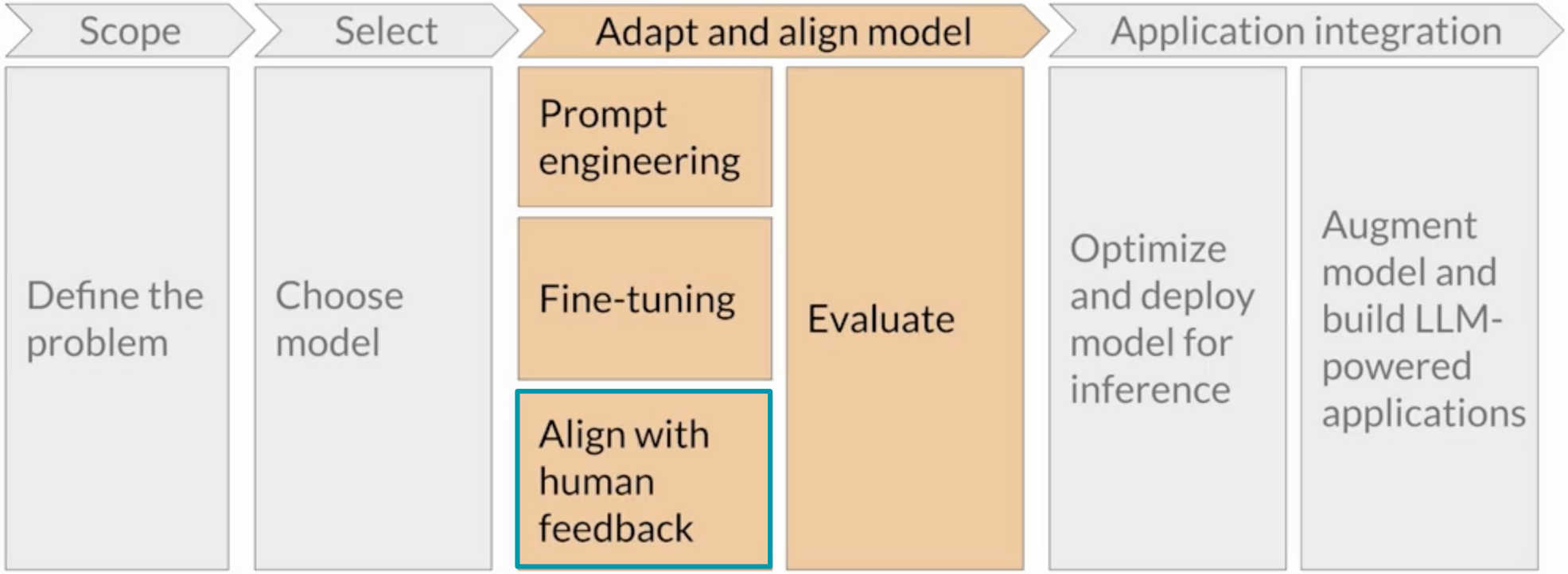
Models behaving badly

- Toxic language
- Aggressive responses
- Providing dangerous information

Models behaving badly

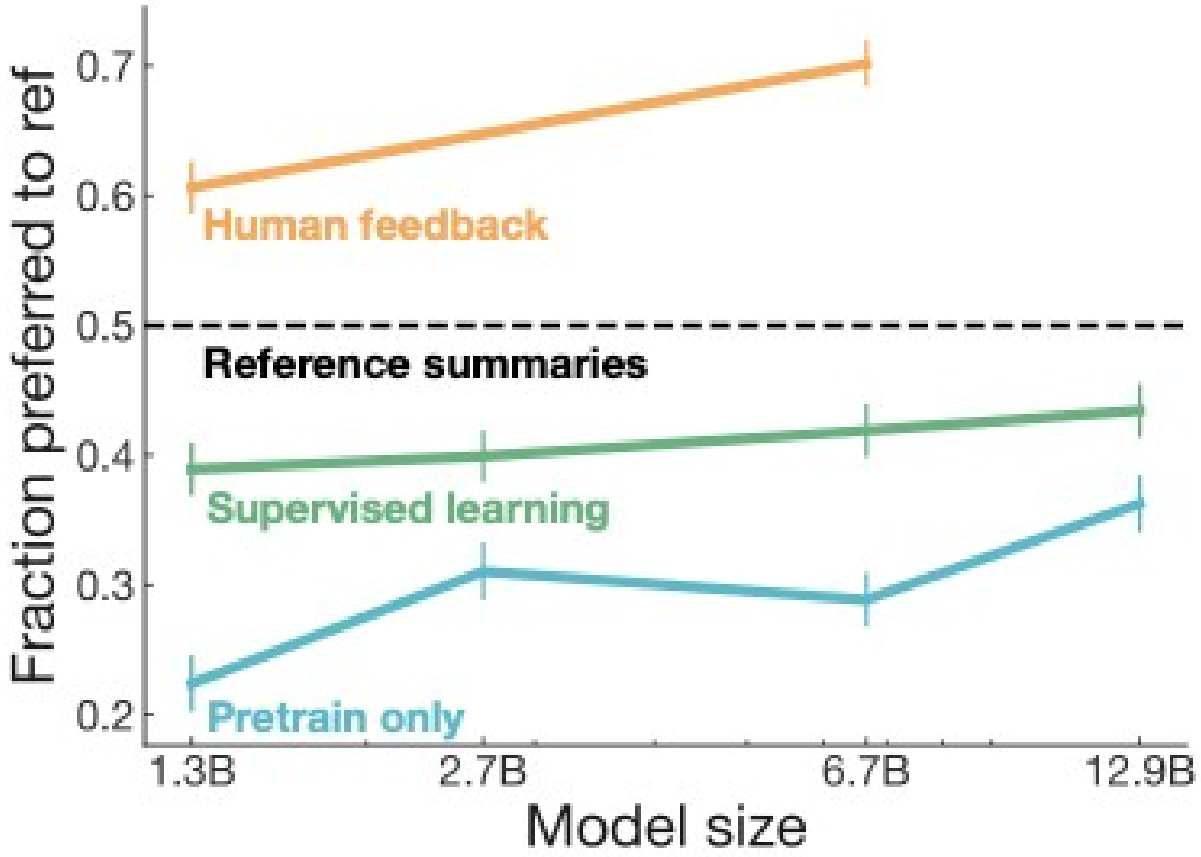


Generative AI project lifecycle



Fine-tuning with human feedback

Fraction of model generated results preferred over human responses.



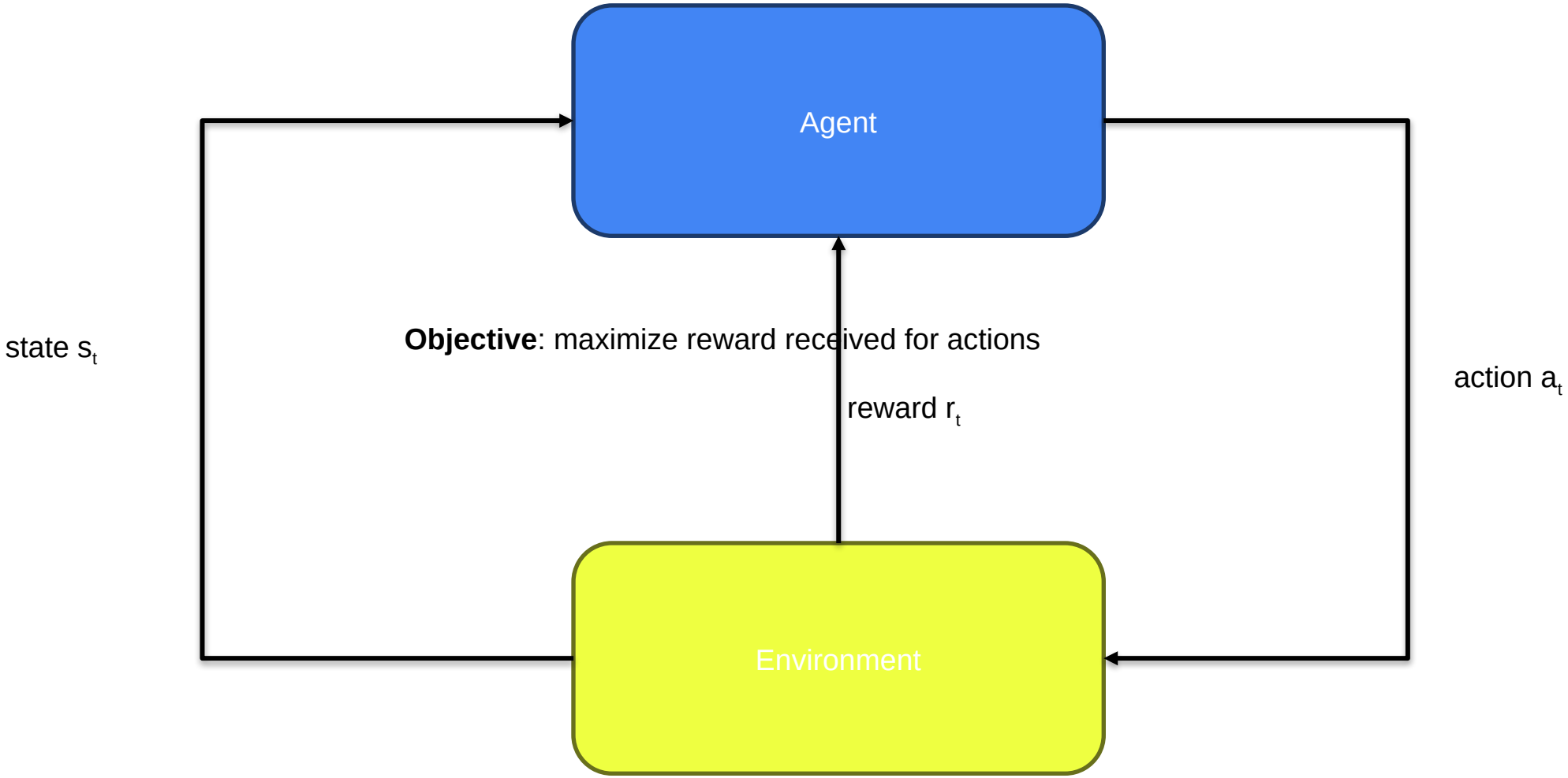
Stiennon, Nisan, et al. "Learning to summarize with human feedback." *Advances in neural information processing systems* 33 (2020): 3008-3021.

Reinforcement learning from human feedback (RLHF)

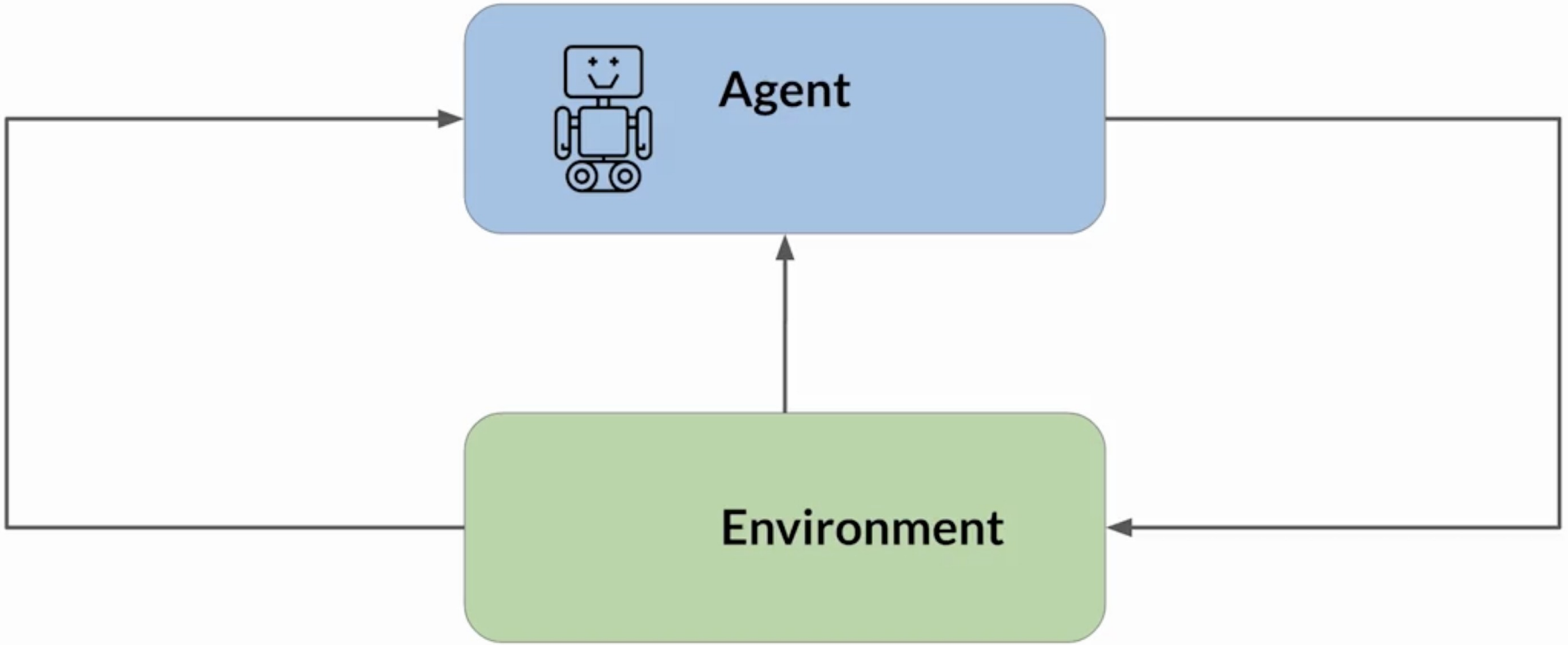


- Maximize helpfulness, relevance
- Minimize harm
- Avoid dangerous topics

Reinforcement Learning

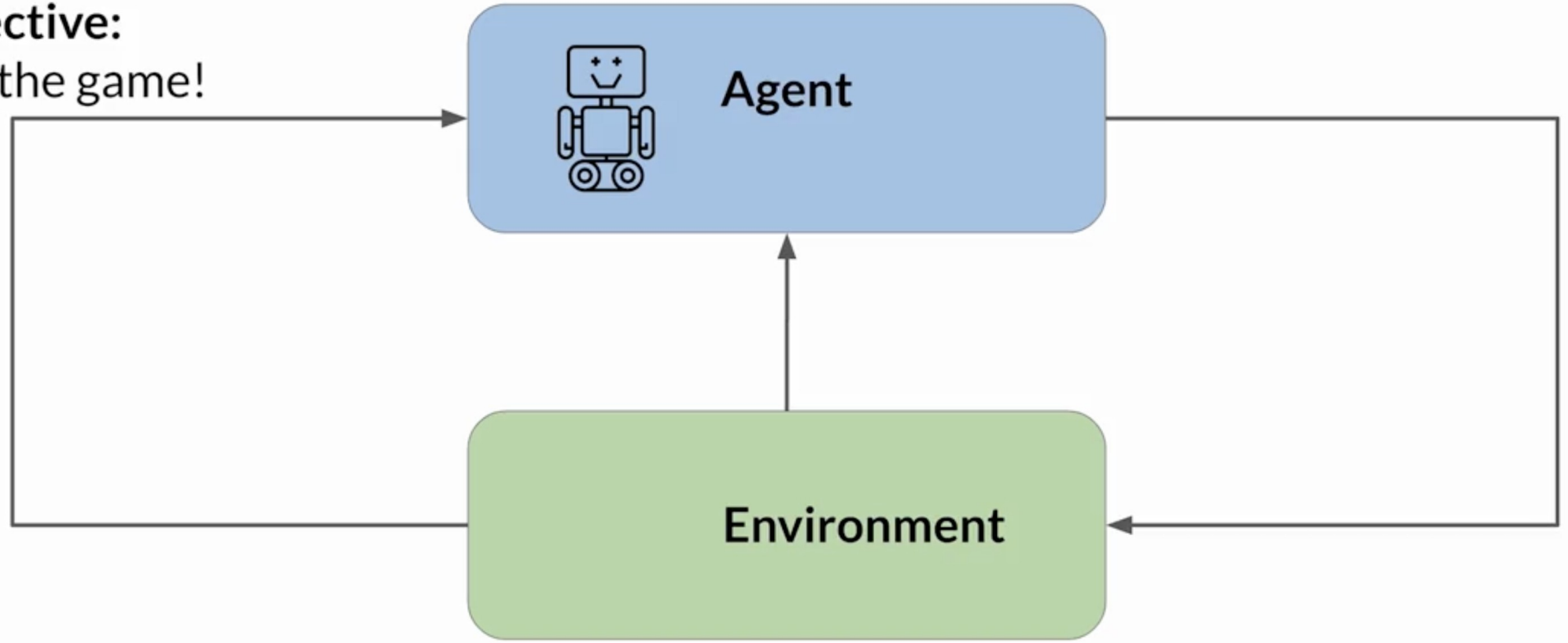


Reinforcement learning: Tic-Tac-Toe

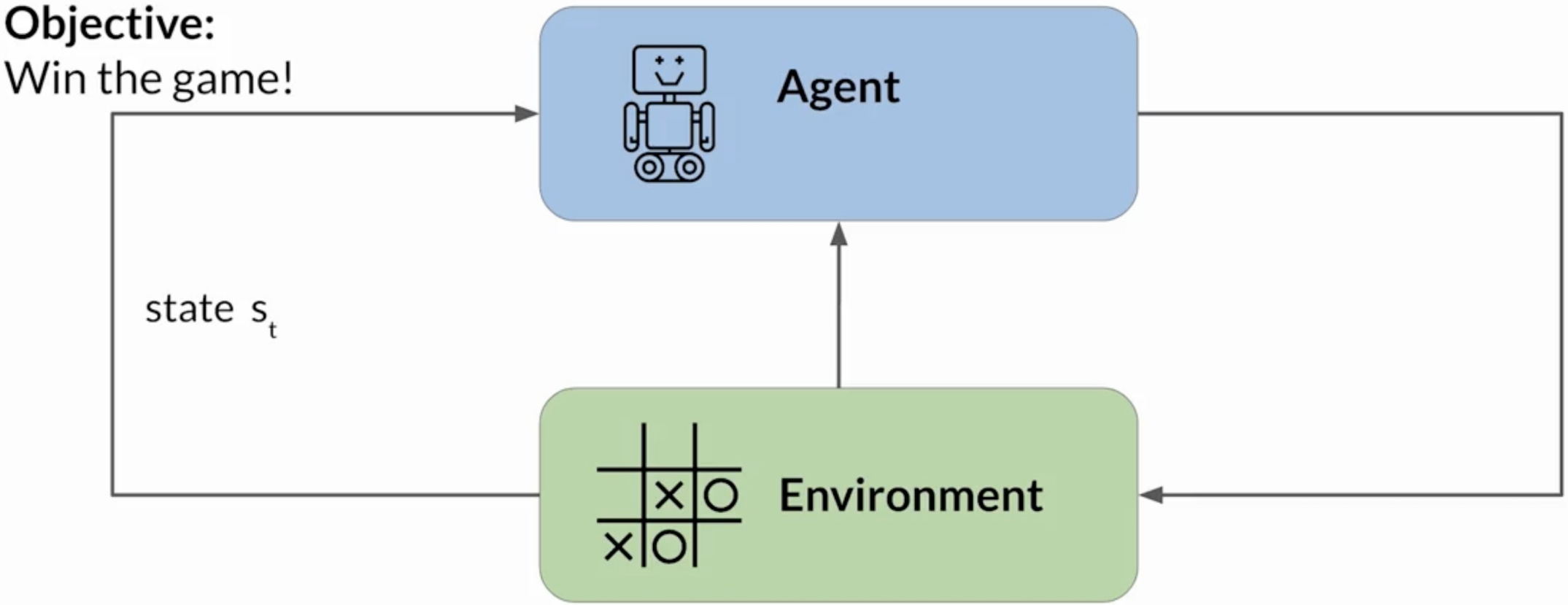


Reinforcement learning: Tic-Tac-Toe

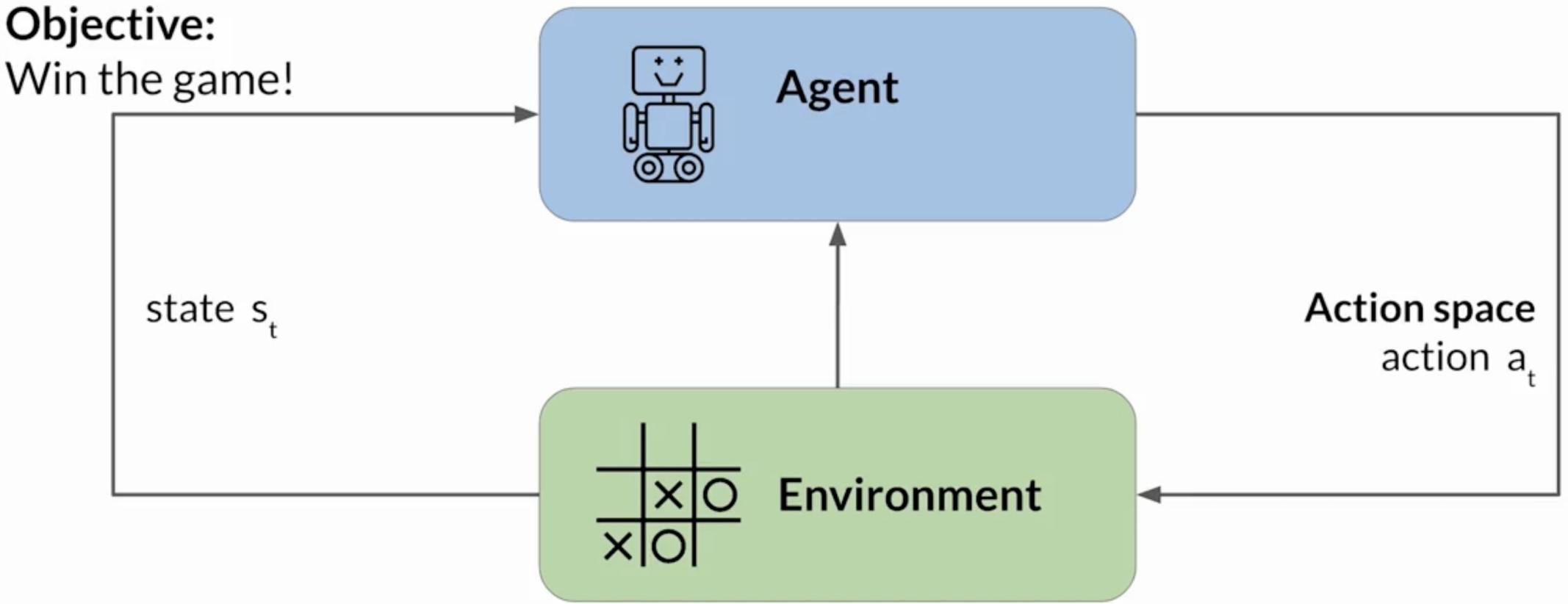
Objective:
Win the game!



Reinforcement learning: Tic-Tac-Toe

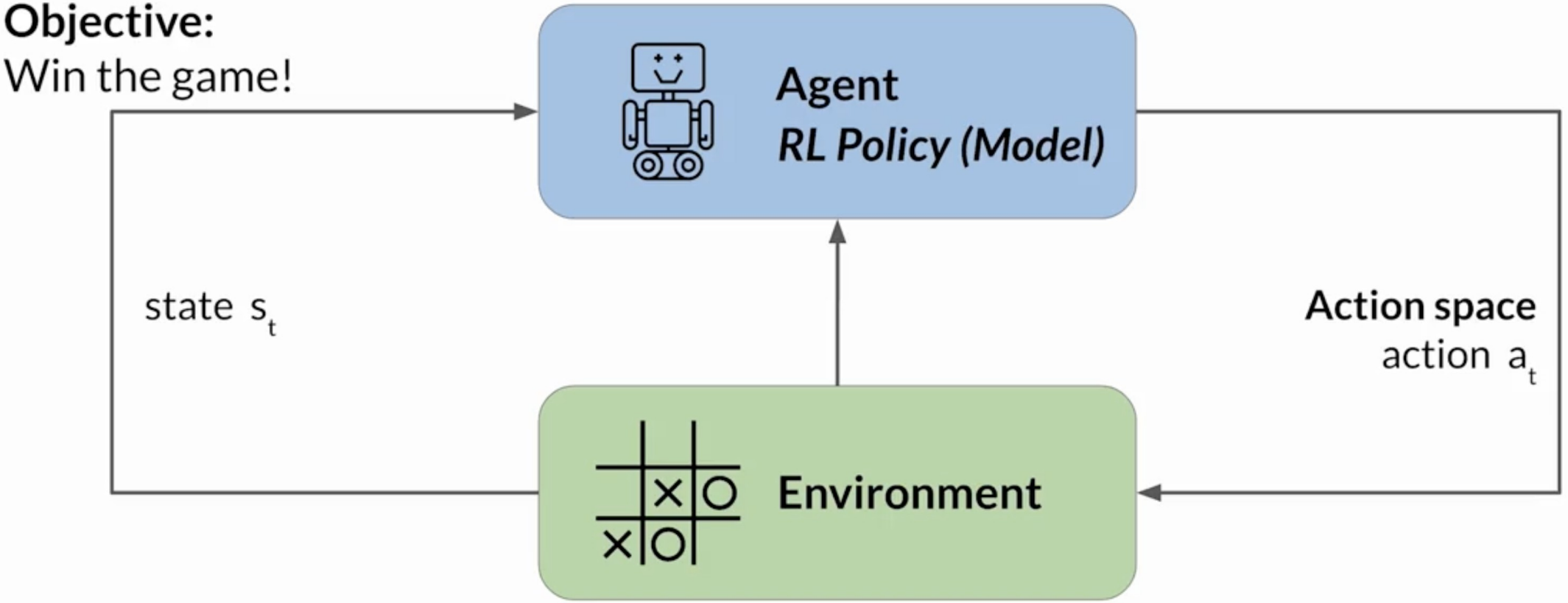


Reinforcement learning: Tic-Tac-Toe

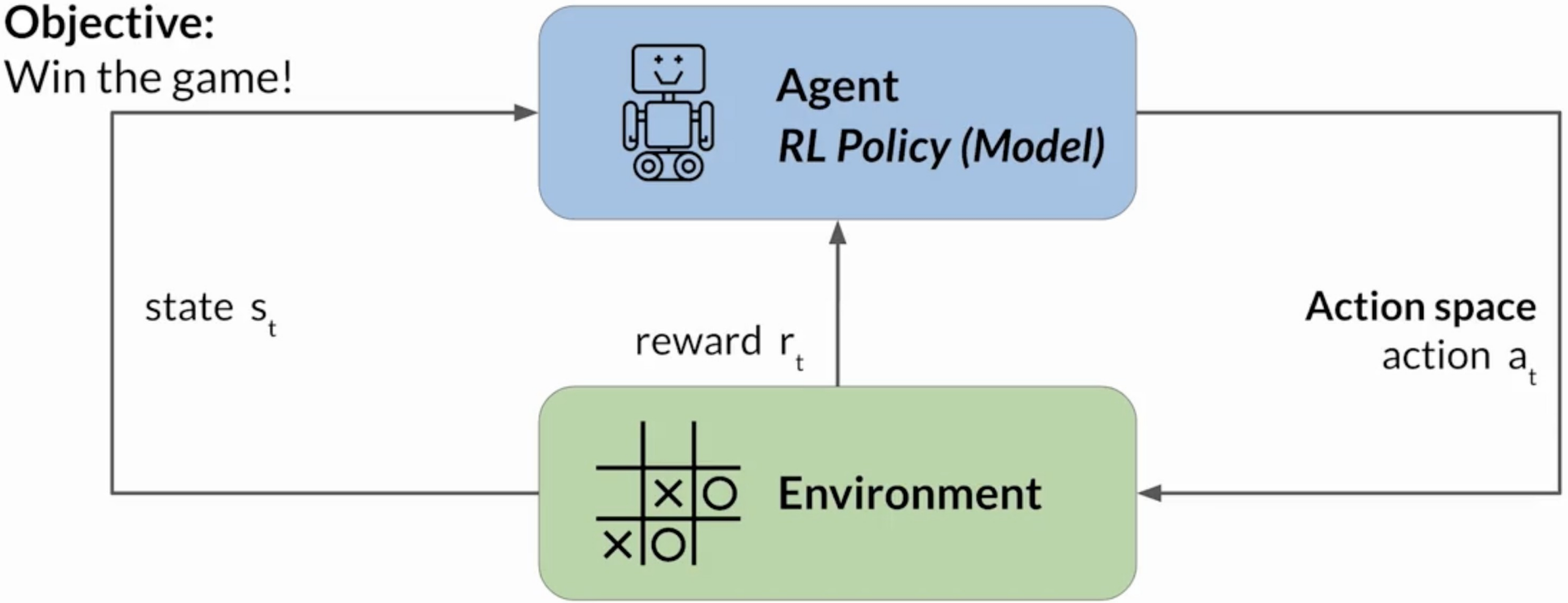


Reinforcement learning: Tic-Tac-Toe

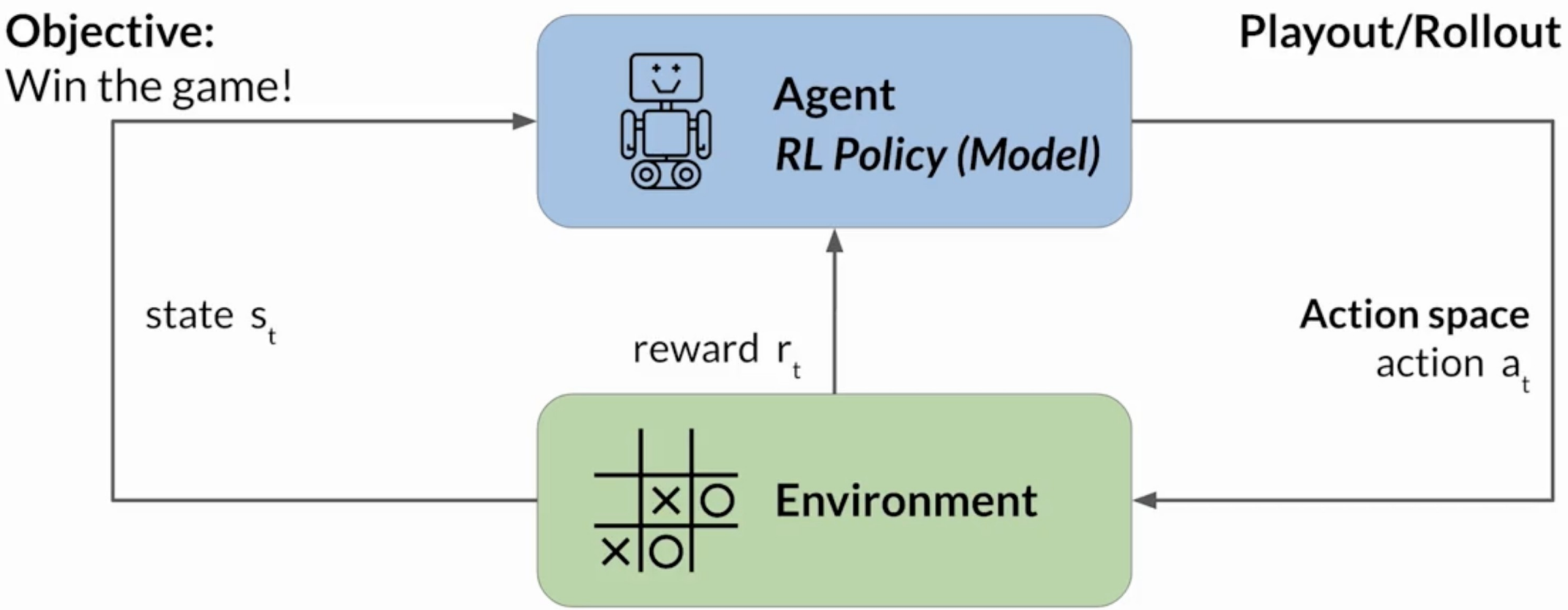
Objective:
Win the game!



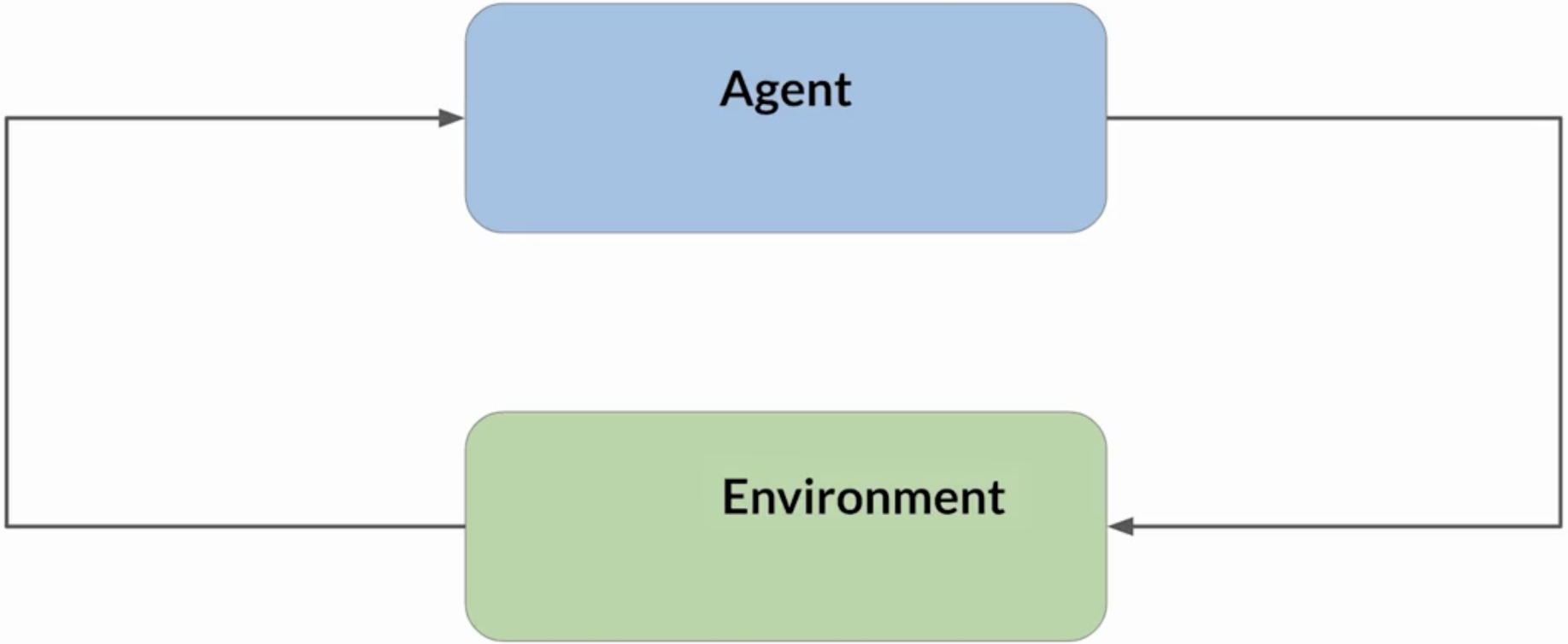
Reinforcement learning: Tic-Tac-Toe



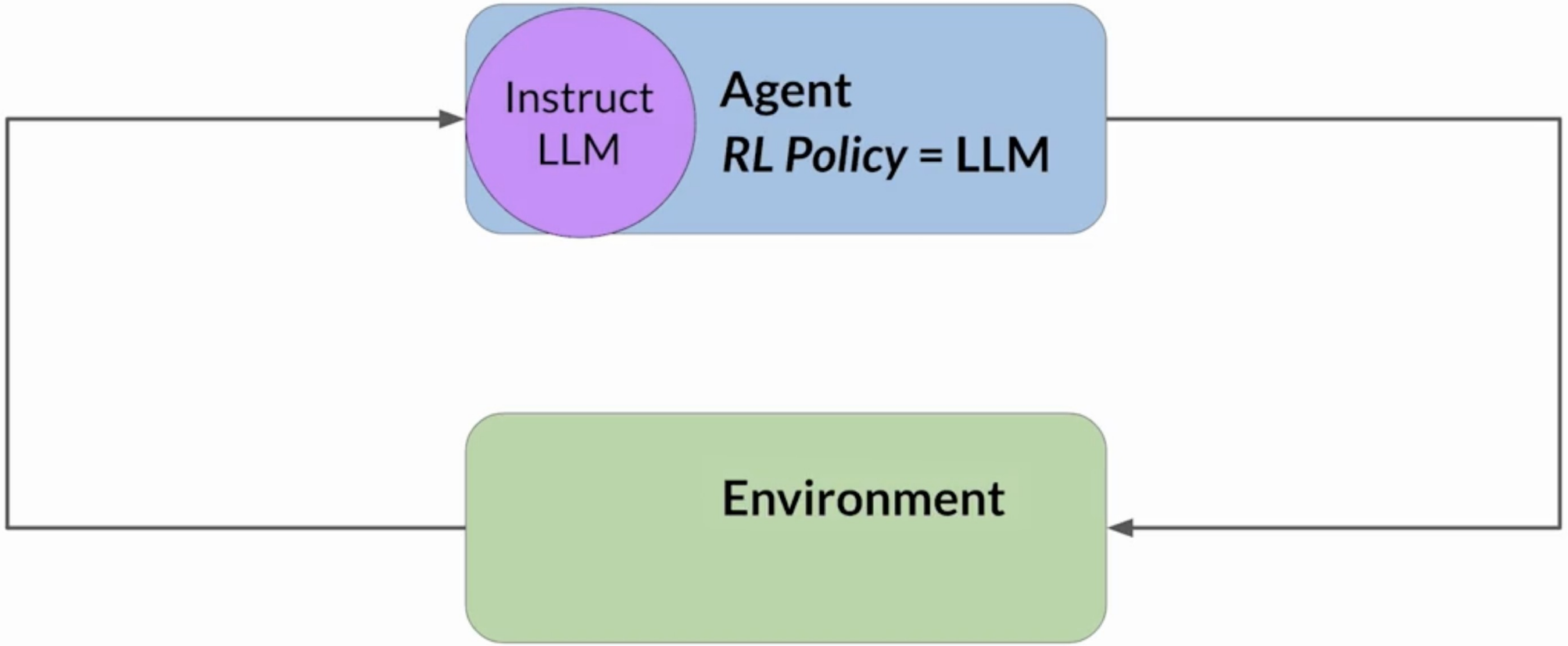
Reinforcement learning: Tic-Tac-Toe



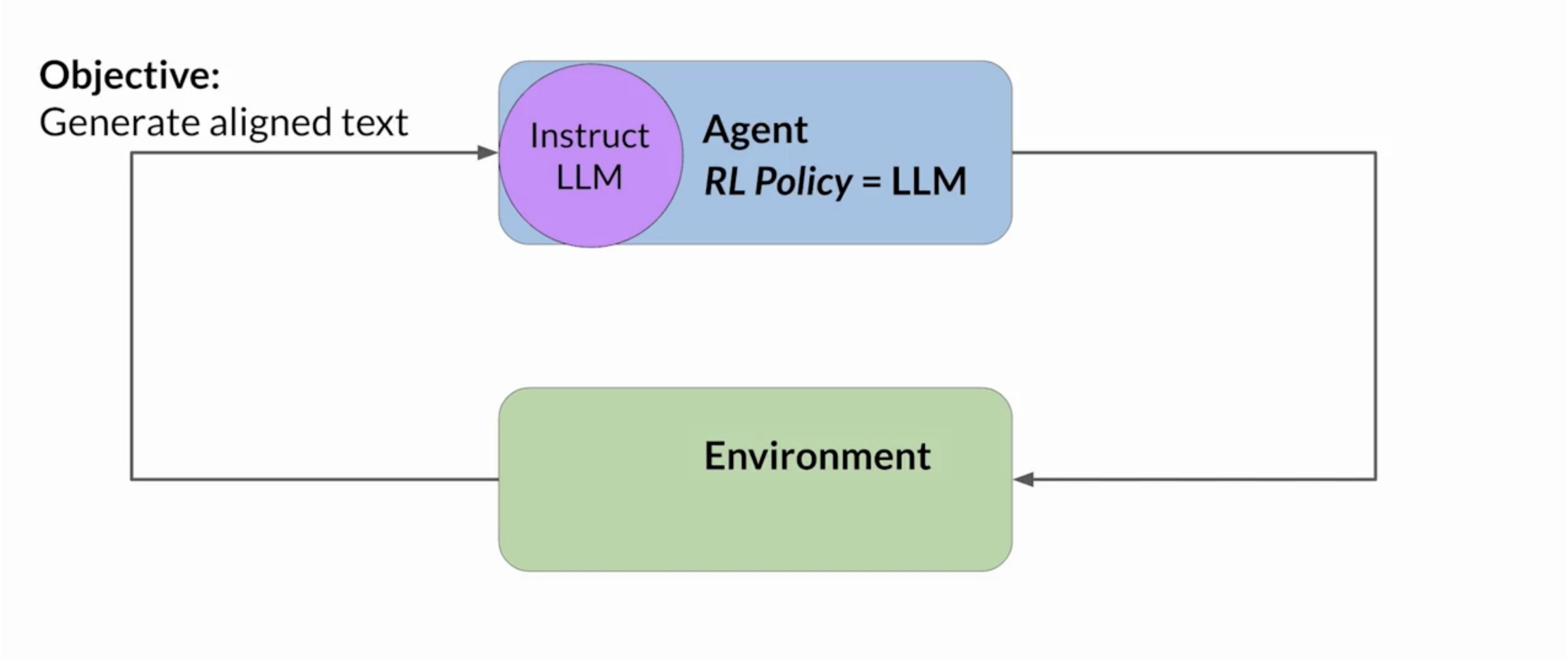
Reinforcement learning: fine-tune LLMs



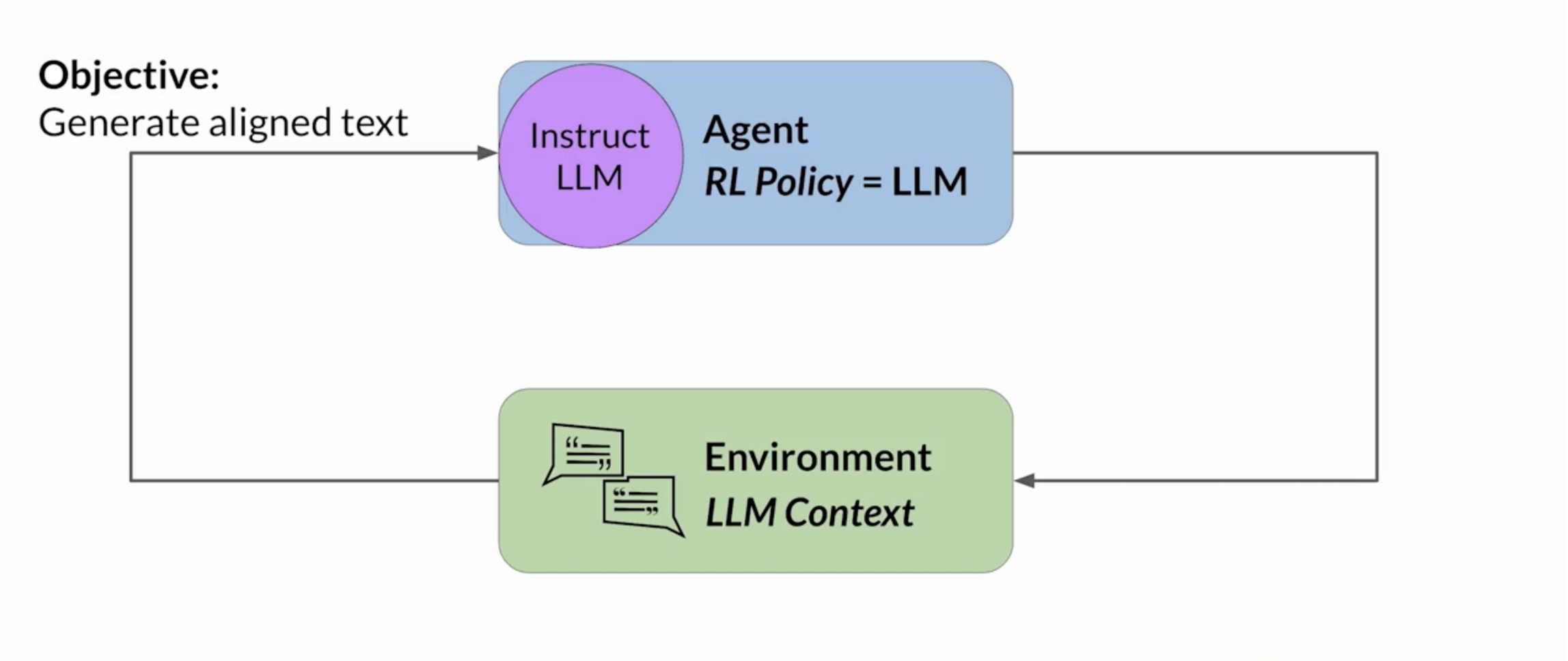
Reinforcement learning: fine-tune LLMs



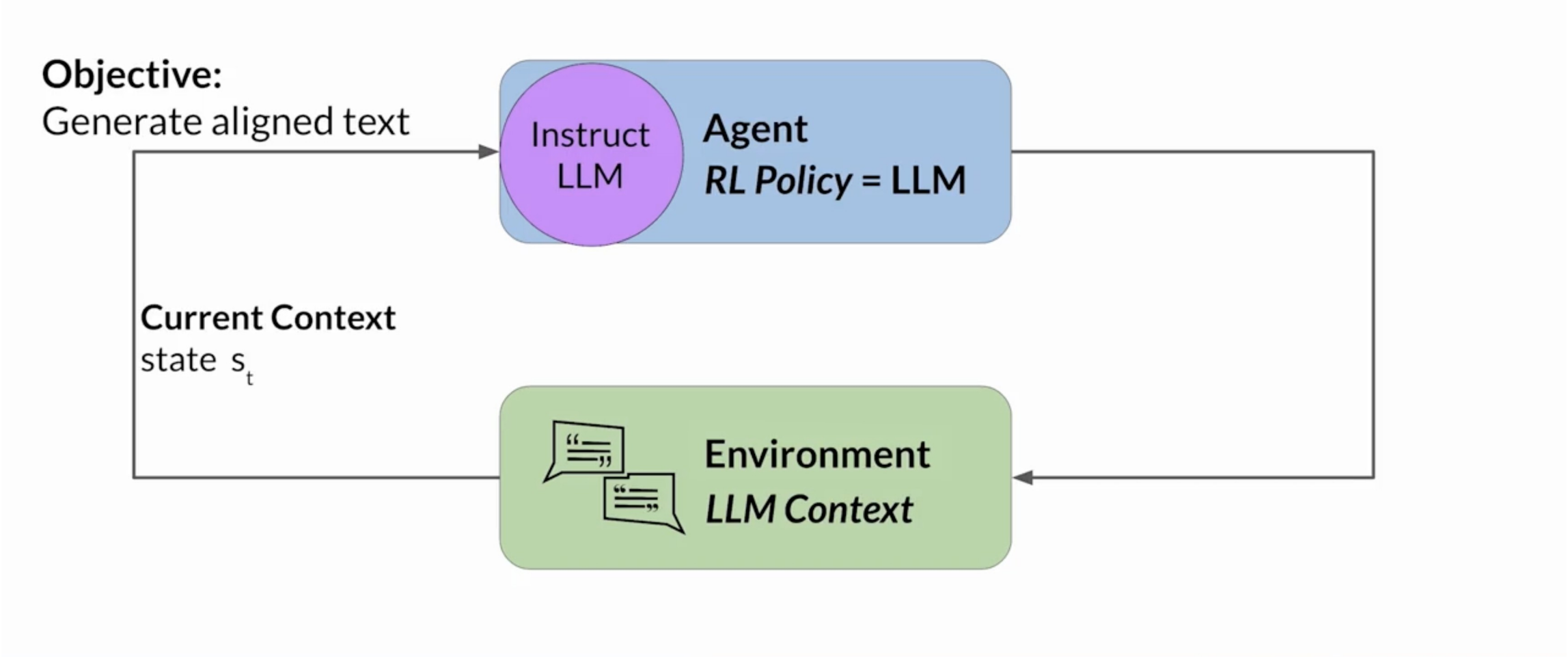
Reinforcement learning: fine-tune LLMs



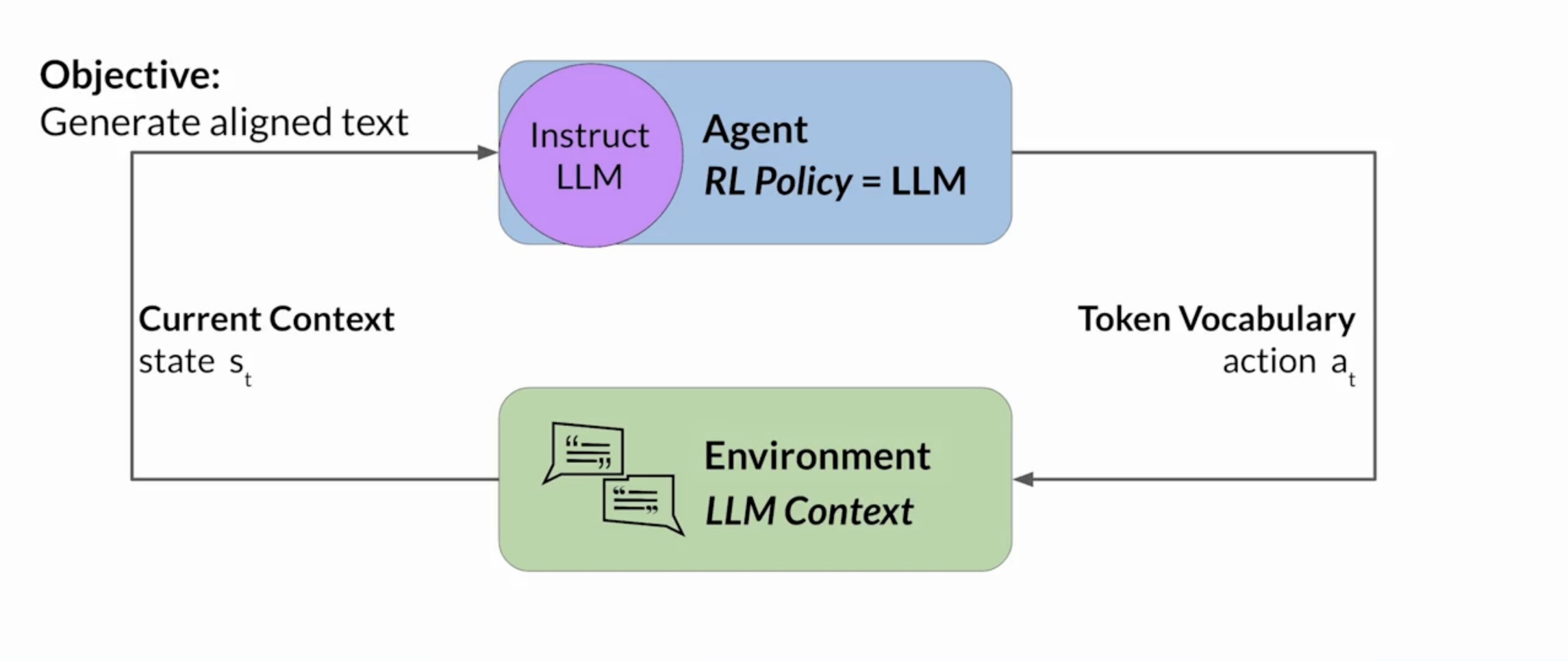
Reinforcement learning: fine-tune LLMs



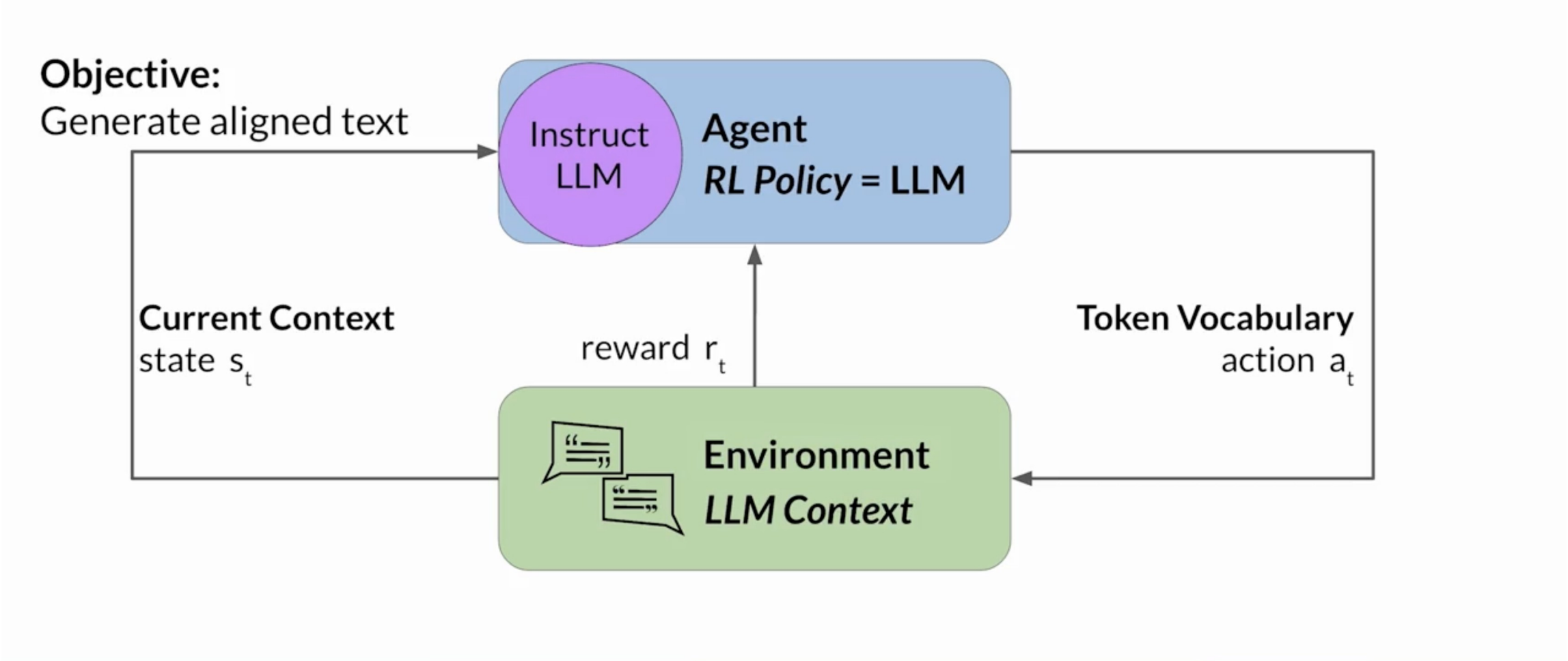
Reinforcement learning: fine-tune LLMs



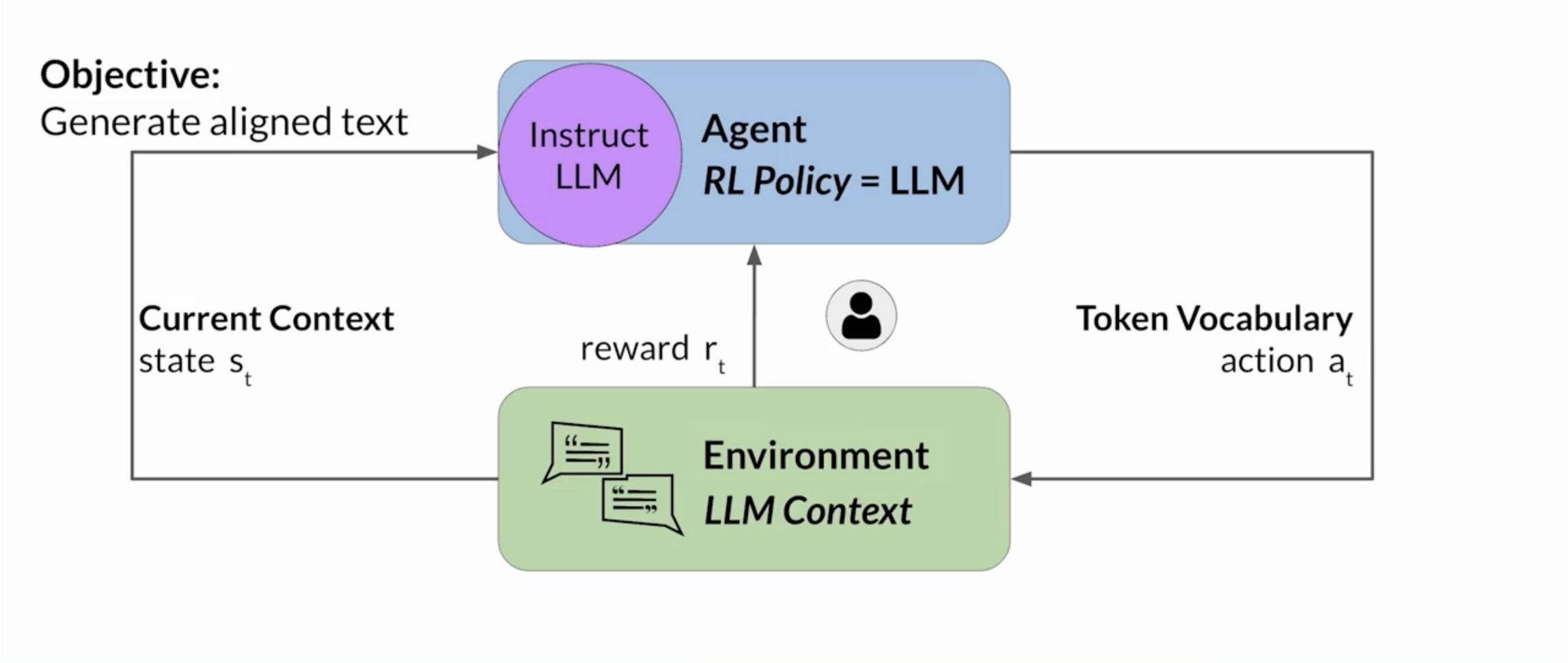
Reinforcement learning: fine-tune LLMs



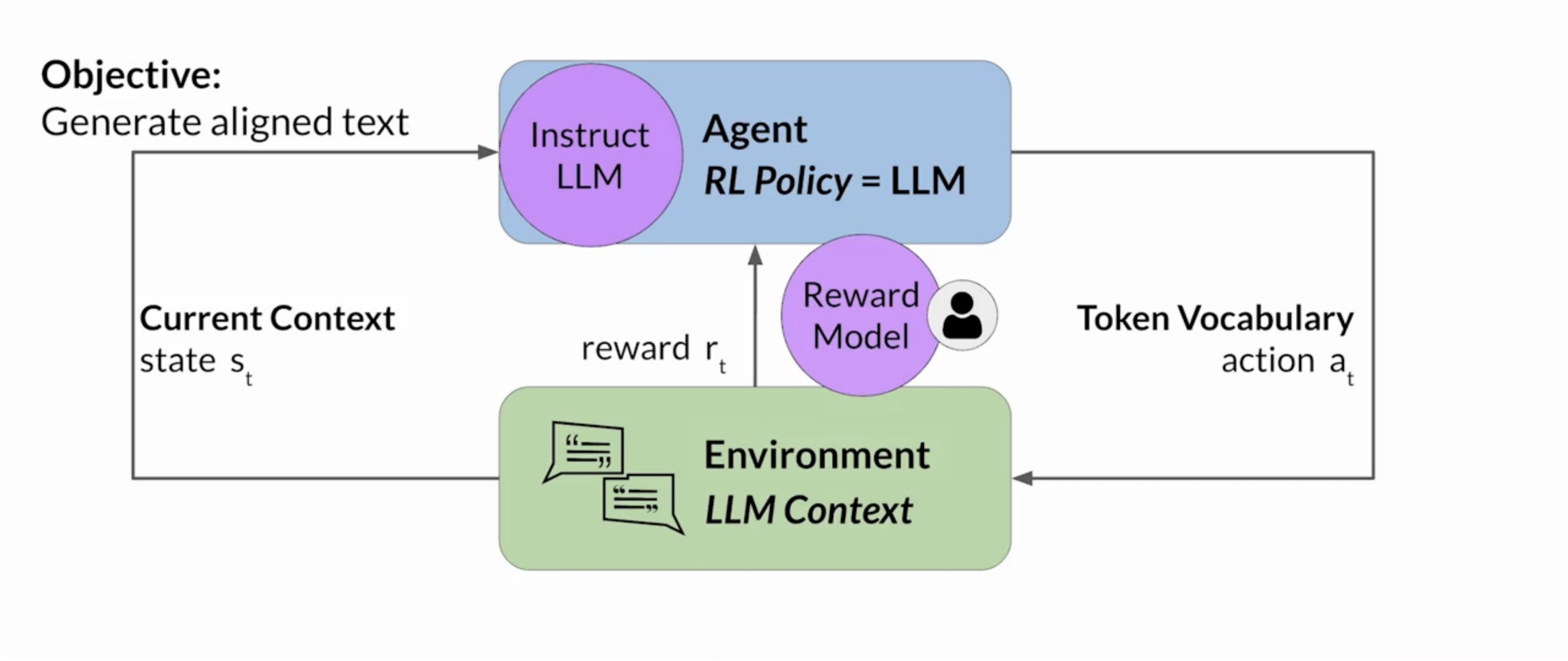
Reinforcement learning: fine-tune LLMs



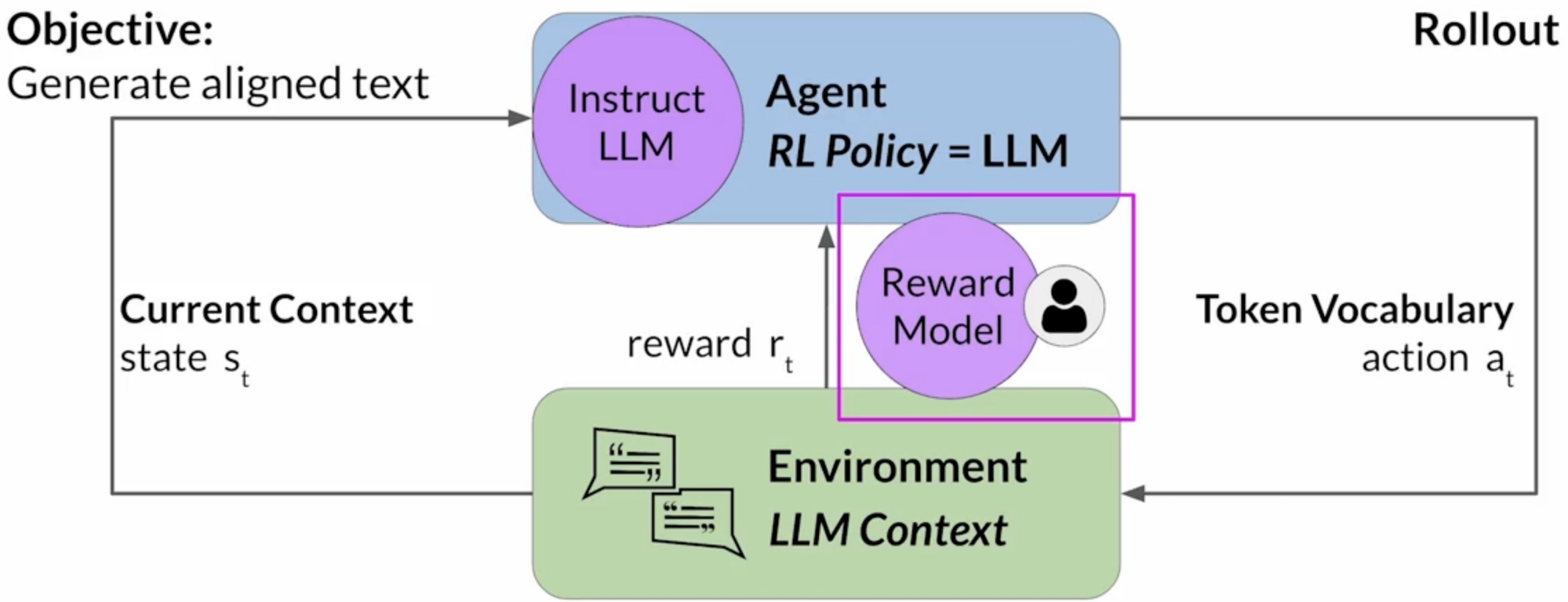
Reinforcement learning: fine-tune LLMs



Reinforcement learning: fine-tune LLMs



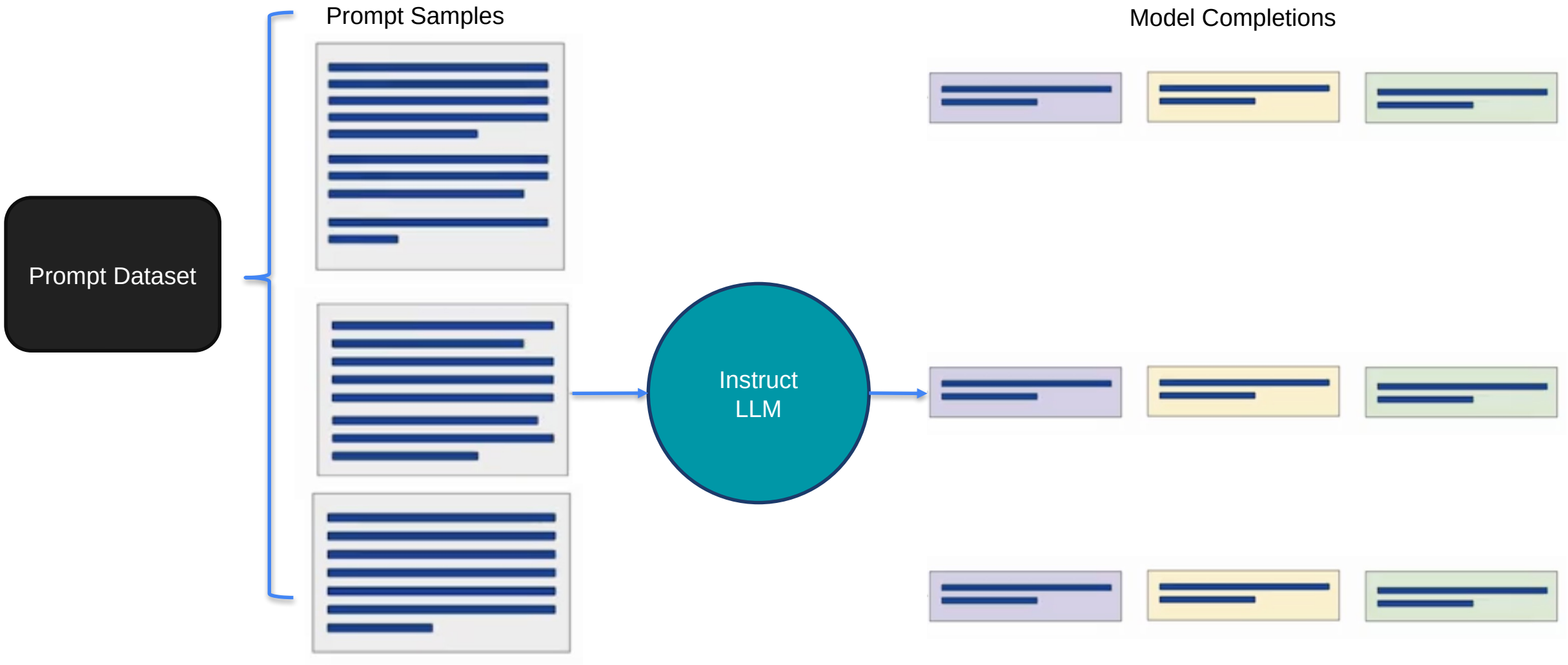
Reinforcement learning: fine-tune LLMs



Prepare dataset for human feedback

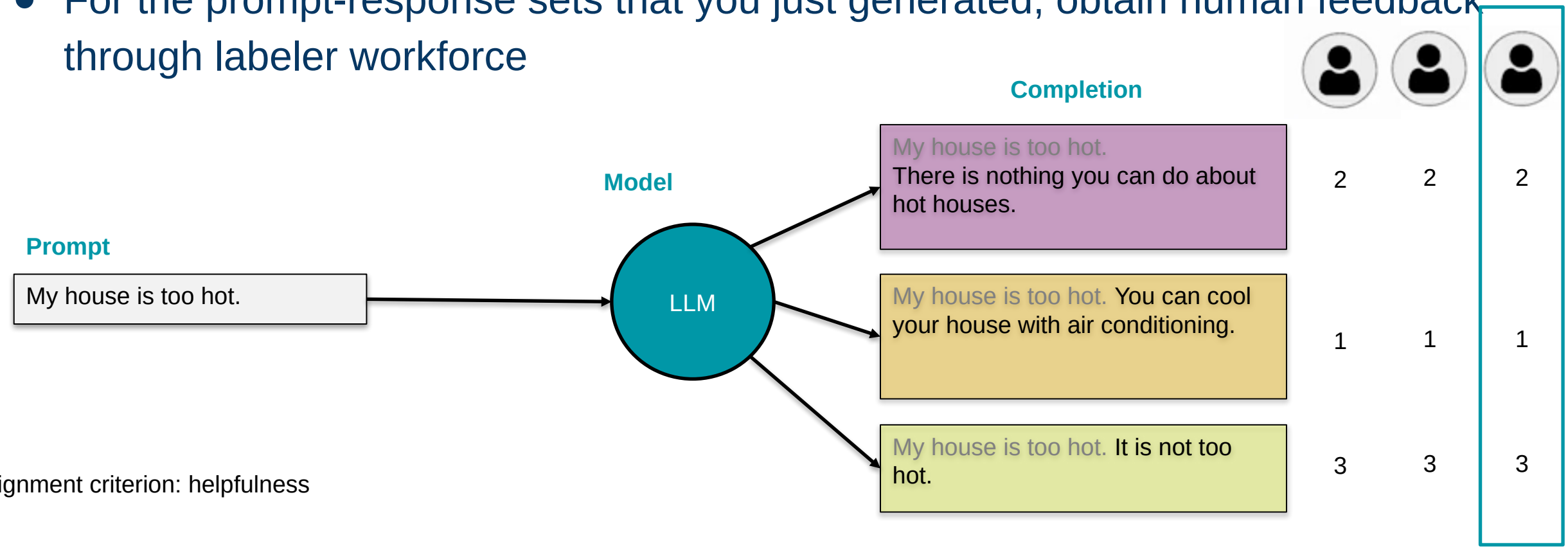


Prepare dataset for human feedback



Collect human feedback

- Define your model alignment criterion
- For the prompt-response sets that you just generated, obtain human feedback through labeler workforce



Alignment criterion: helpfulness

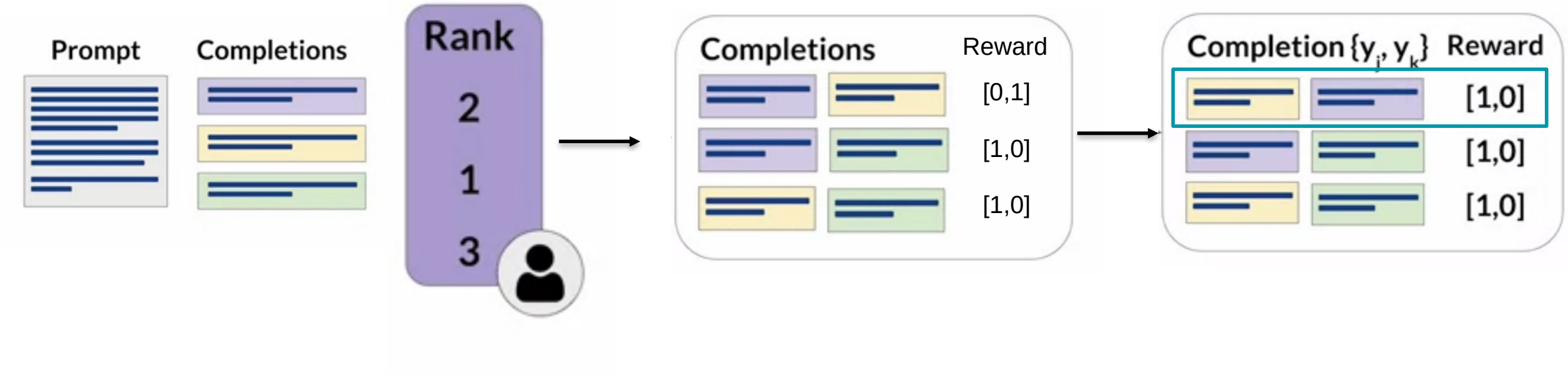
Sample instructions for human labelers

* Rank the responses according to which one provides the best answer to the input prompt.
* What is the best answer? Make a decision based on (a) the correctness of the answer, and (b) the informativeness of the response. For (a) you are allowed to search the web. Overall, use your best judgment to rank answers based on being the most useful response, which we define as one which is at least somewhat correct, and minimally informative about what the prompt is asking for.
* If two responses provide the same correctness and informativeness by your judgment, and there is no clear winner, you may rank them the same, but please only use this sparingly.
* If the answer for a given response is nonsensical, irrelevant, highly ungrammatical/confusing, or does not clearly respond to the given prompt, label it with "F" (for fail) rather than its rank.
* Long answers are not always the best. Answers which provide succinct, coherent responses may be better than longer ones, if they are at least as correct and informative.

Source: Chung et al. 2022, "Scaling Instruction-Finetuned Language Models"

Prepare labeled data for training

- Convert rankings into pairwise training data for the reward model



Stiennon, Nisan, et al. "Learning to summarize with human feedback." *Advances in Neural Information Processing Systems* 33 (2020): 3008-3021.

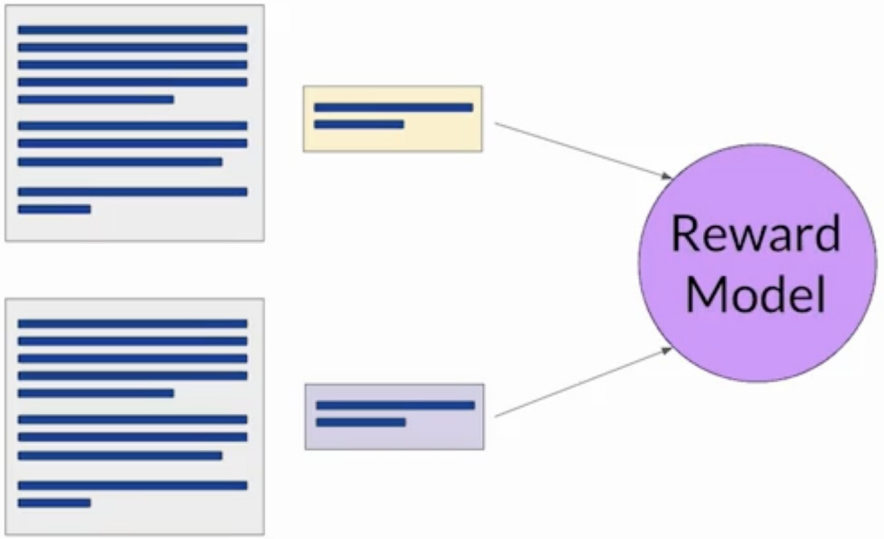
Question?

"When using Reinforcement Learning with Human Feedback (RLHF) to align large language models with human preferences, what is the role of human labelers?”

A. To write prompts and completions from scratch that are used during fine-tuning with RLHF
B. To compare the original LLM completions to the RLHF updated model completions and ensure they don't diverge too much
C. To identify model weights that should be updated
D. To score prompt completions, so that this score is used to train the reward model component of the RLHF process

Train reward model

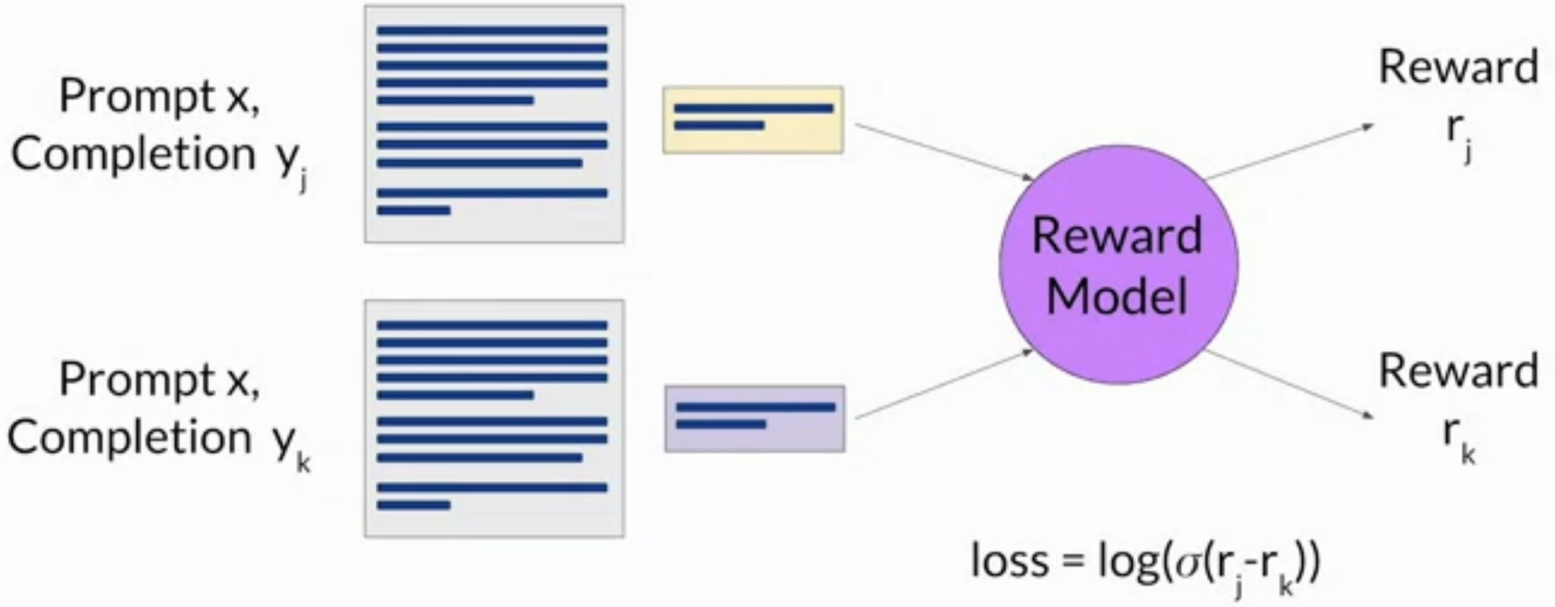
Train model to predict preferred completion from $\{y_j, y_k\}$ for prompt x



Source: Stiennon et al. 2020, "Learning to summarize from human feedback"

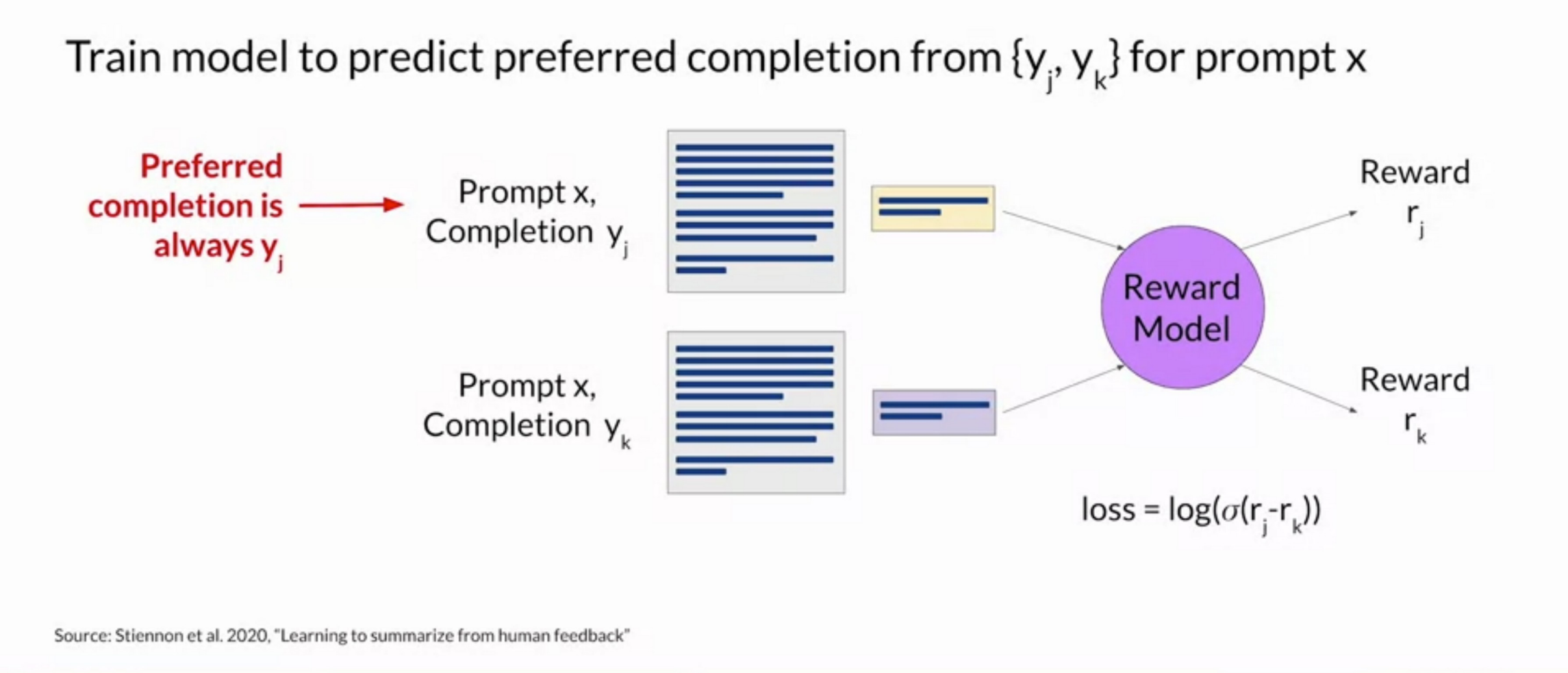
Train reward model

Train model to predict preferred completion from $\{y_j, y_k\}$ for prompt x



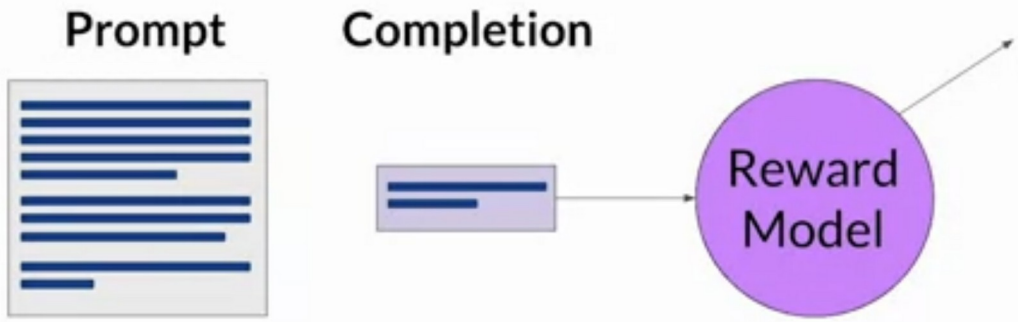
Source: Stiennon et al. 2020, "Learning to summarize from human feedback"

Train reward model



Use the reward model

Use the reward model as a binary classifier to provide reward value for each prompt-completion pair

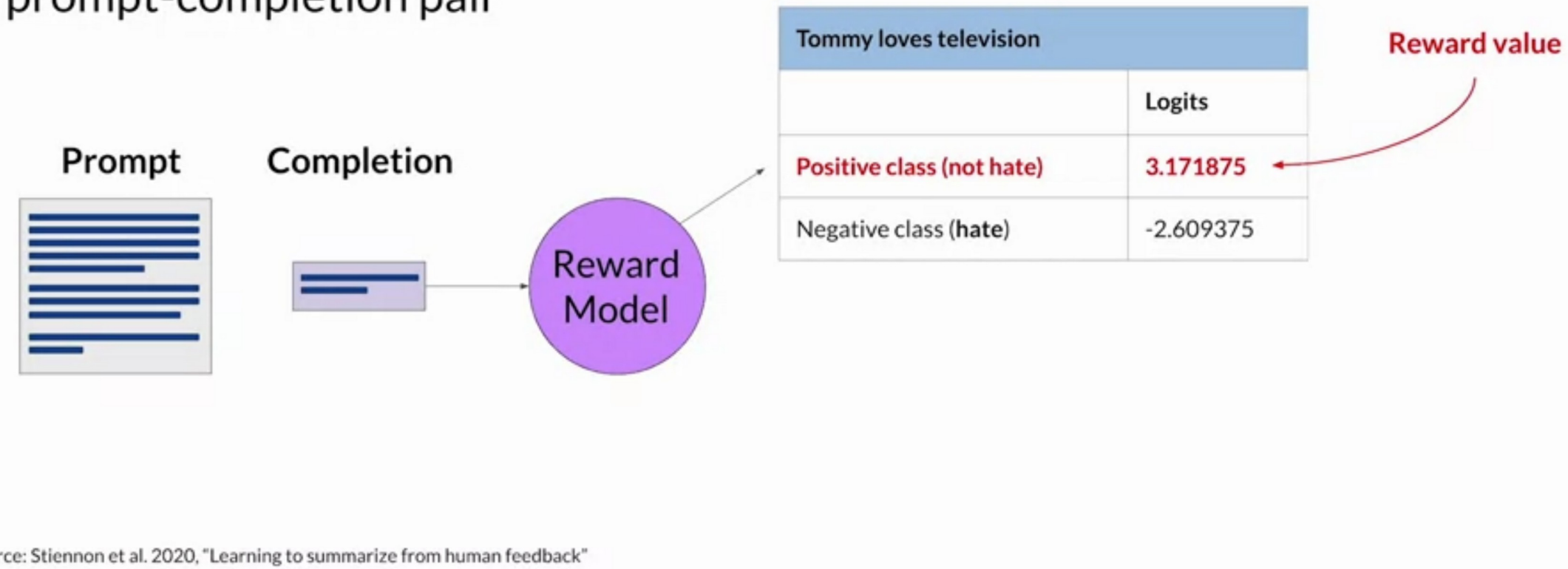


Tommy loves television	
	Logits
Positive class (not hate)	3.171875
Negative class (**hate**)	-2.609375

Source: Stiennon et al. 2020, "Learning to summarize from human feedback"

Use the reward model

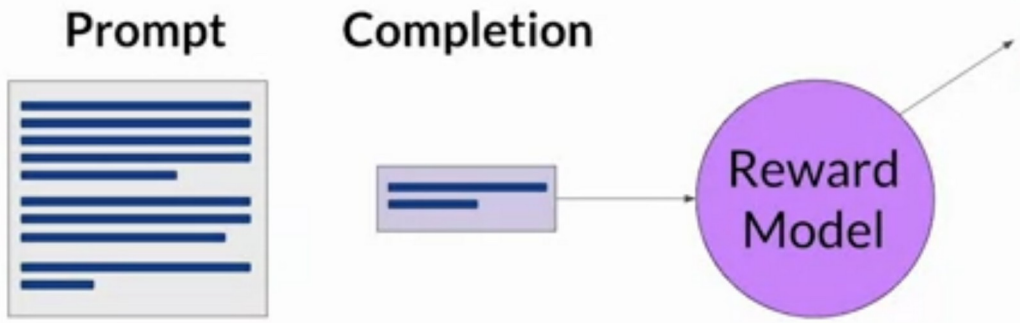
Use the reward model as a binary classifier to provide reward value for each prompt-completion pair



Source: Stiennon et al. 2020, "Learning to summarize from human feedback"

Use the reward model

Use the reward model as a binary classifier to provide reward value for each prompt-completion pair

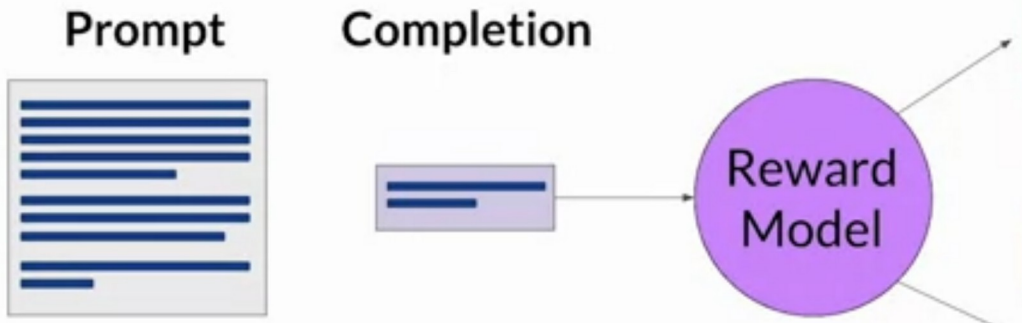


Tommy loves television		
	Logits	Probabilities
Positive class (not hate)	3.171875	0.996093
Negative class (**hate**)	-2.609375	0.003082

Source: Stiennon et al. 2020, "Learning to summarize from human feedback"

Use the reward model

Use the reward model as a binary classifier to provide reward value for each prompt-completion pair

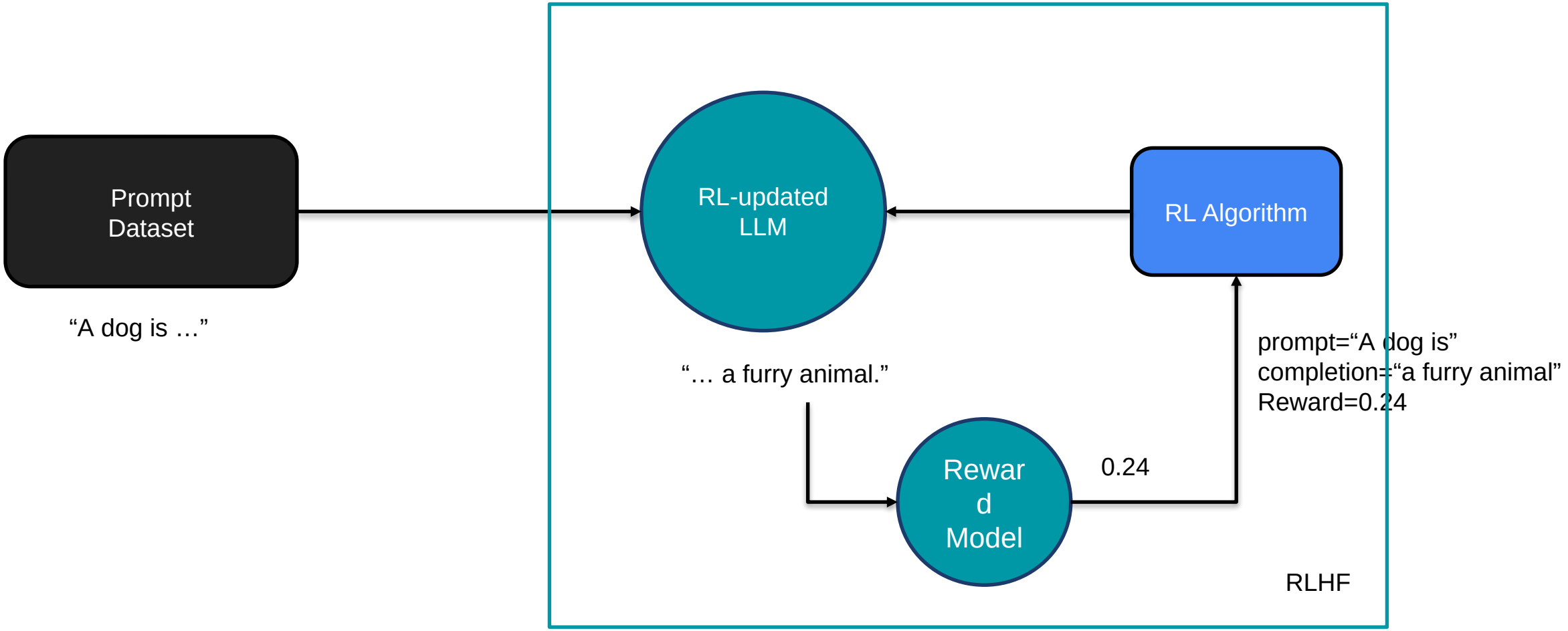


Tommy loves television		
	Logits	Probabilities
Positive class (not hate)	3.171875	0.996093
Negative class (**hate**)	-2.609375	0.003082

Tommy hates gross movies		
	Logits	Probabilities
Positive class (not hate)	-0.535156	0.337890
Negative class (**hate**)	0.137695	0.664062

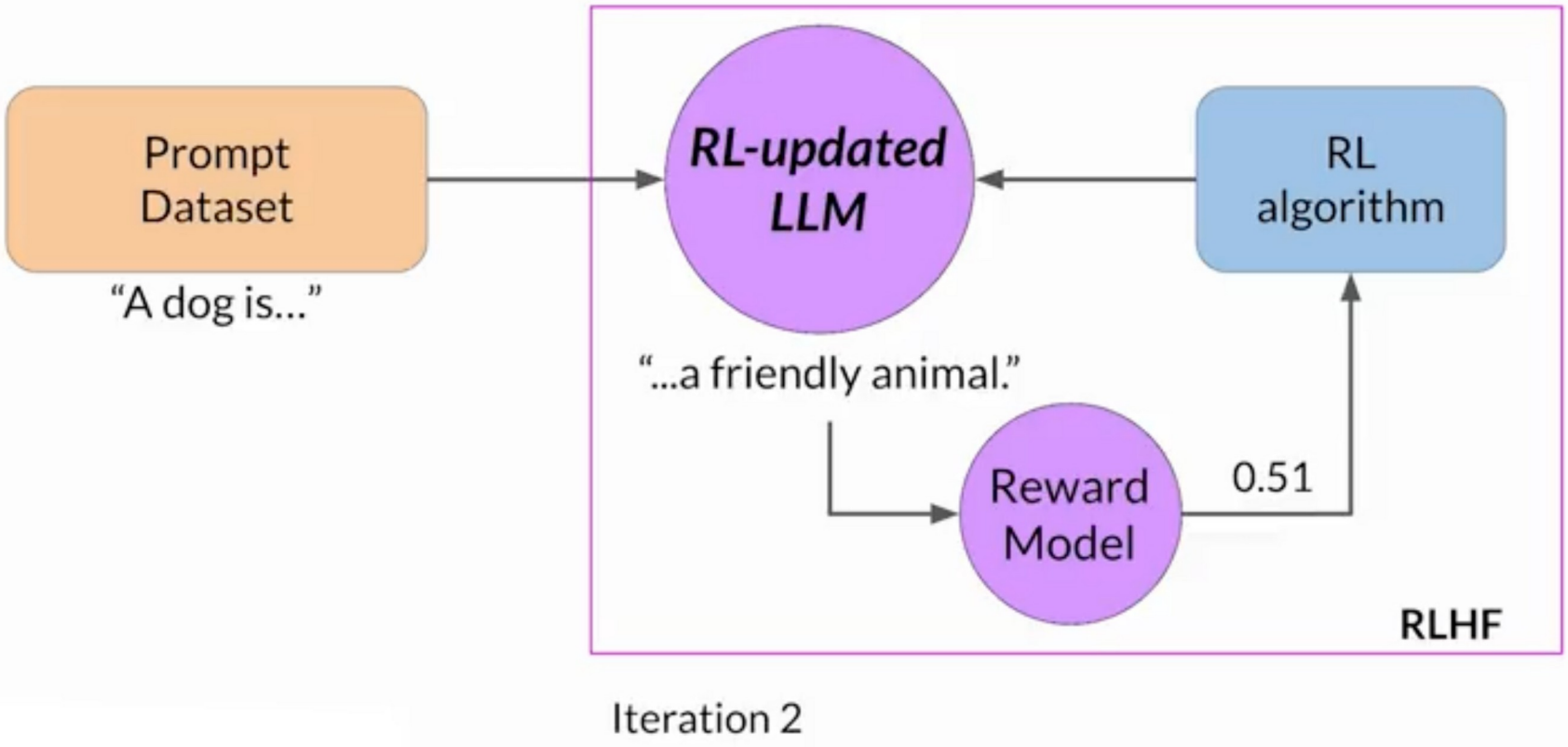
Source: Stiennon et al. 2020, "Learning to summarize from human feedback"

Use the reward model to fine-tune LLM with RL

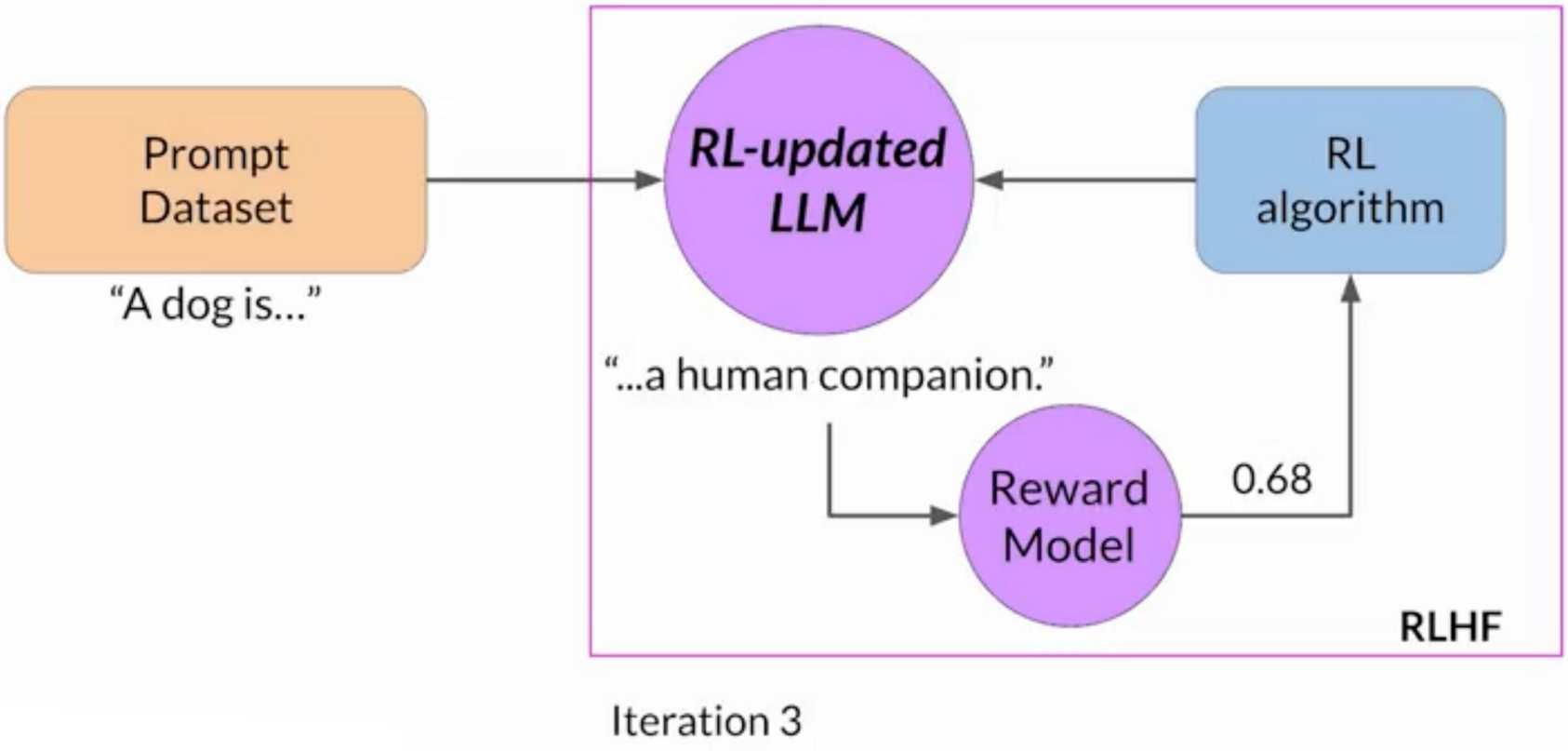


Iteration 1

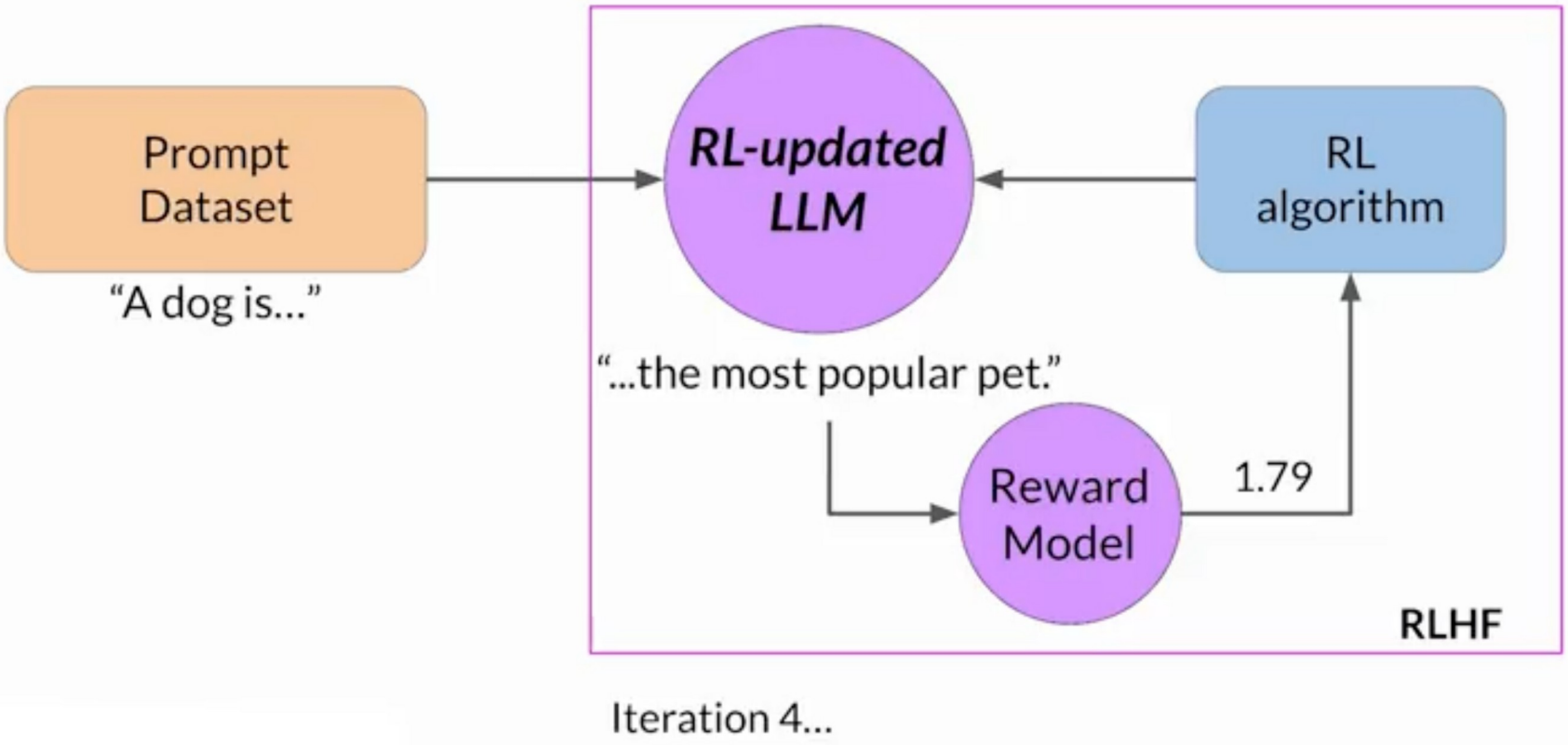
Use the reward model to fine-tune LLM with RL



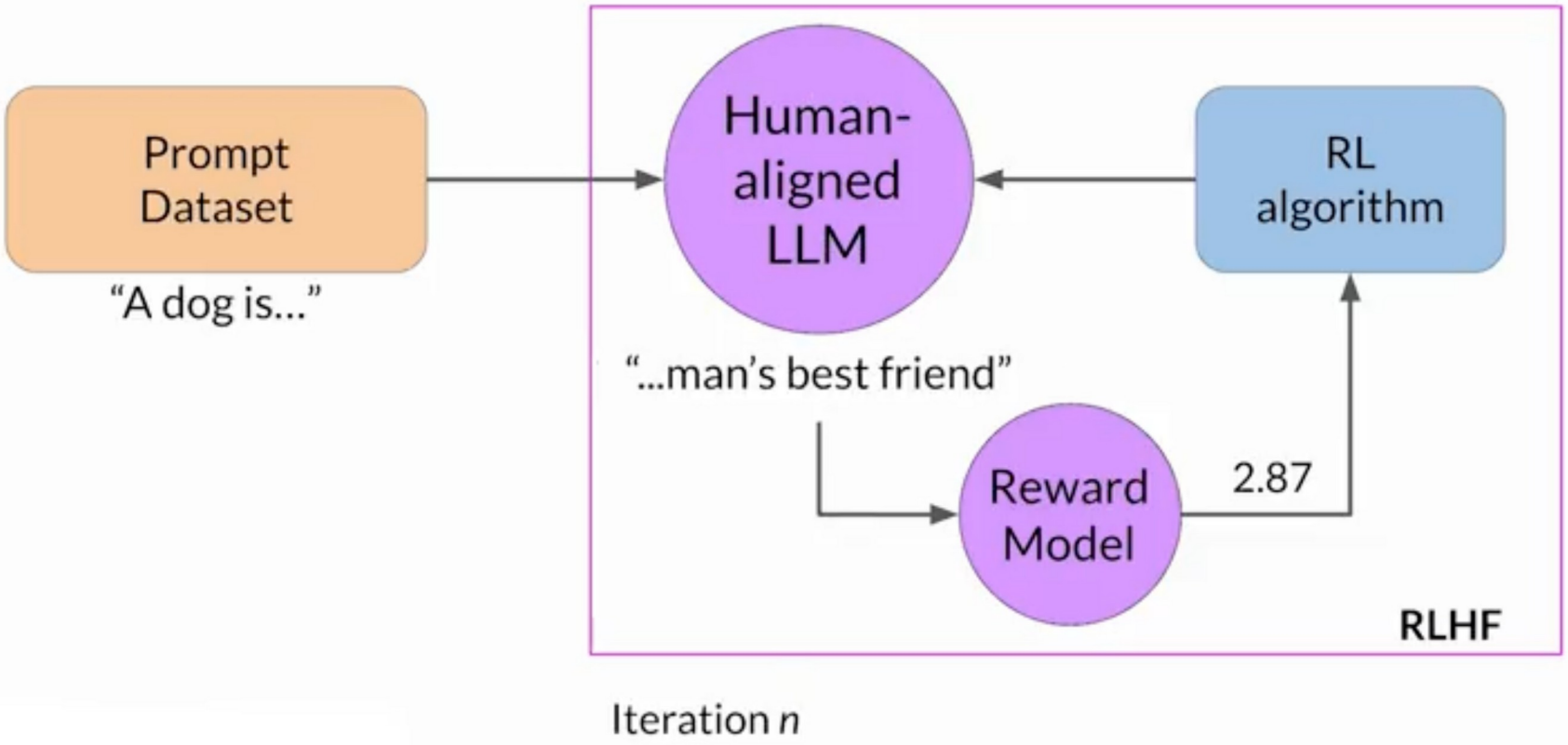
Use the reward model to fine-tune LLM with RL



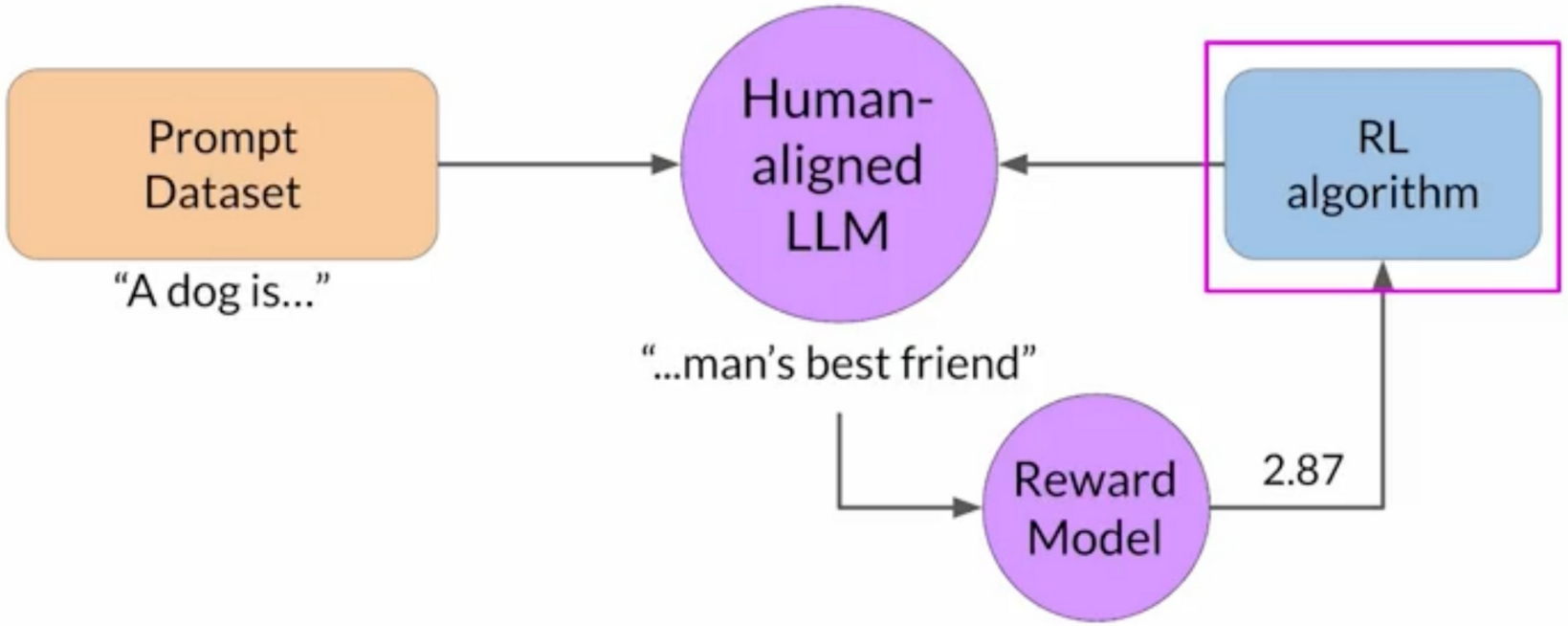
Use the reward model to fine-tune LLM with RL



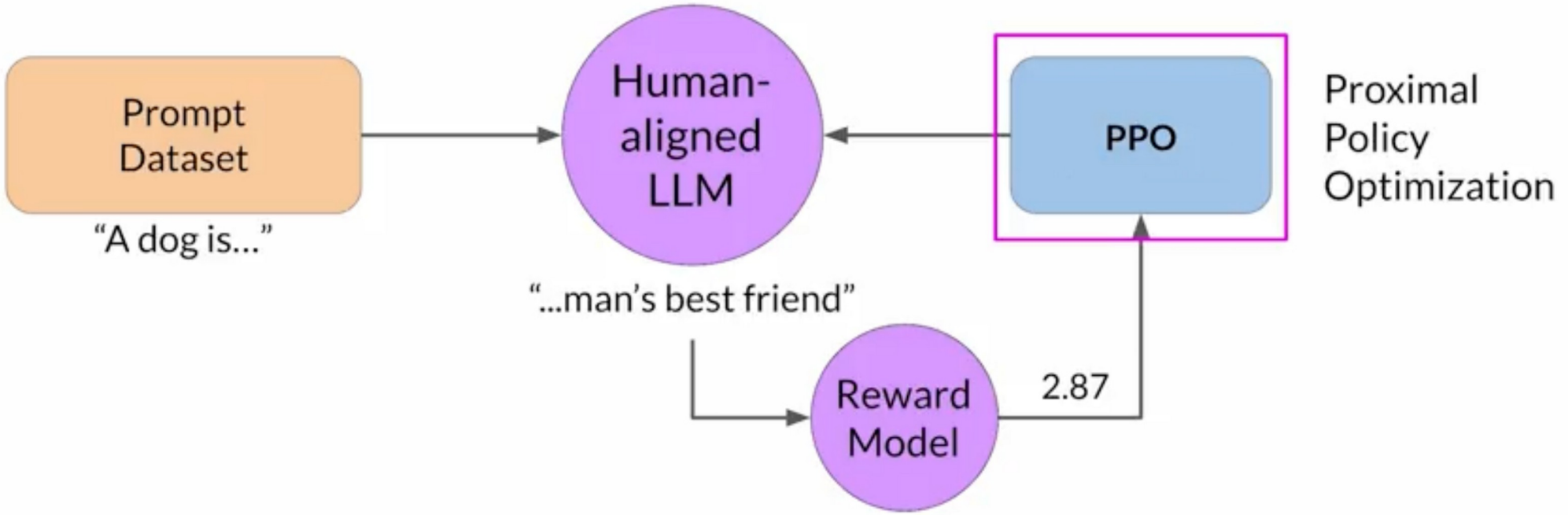
Use the reward model to fine-tune LLM with RL



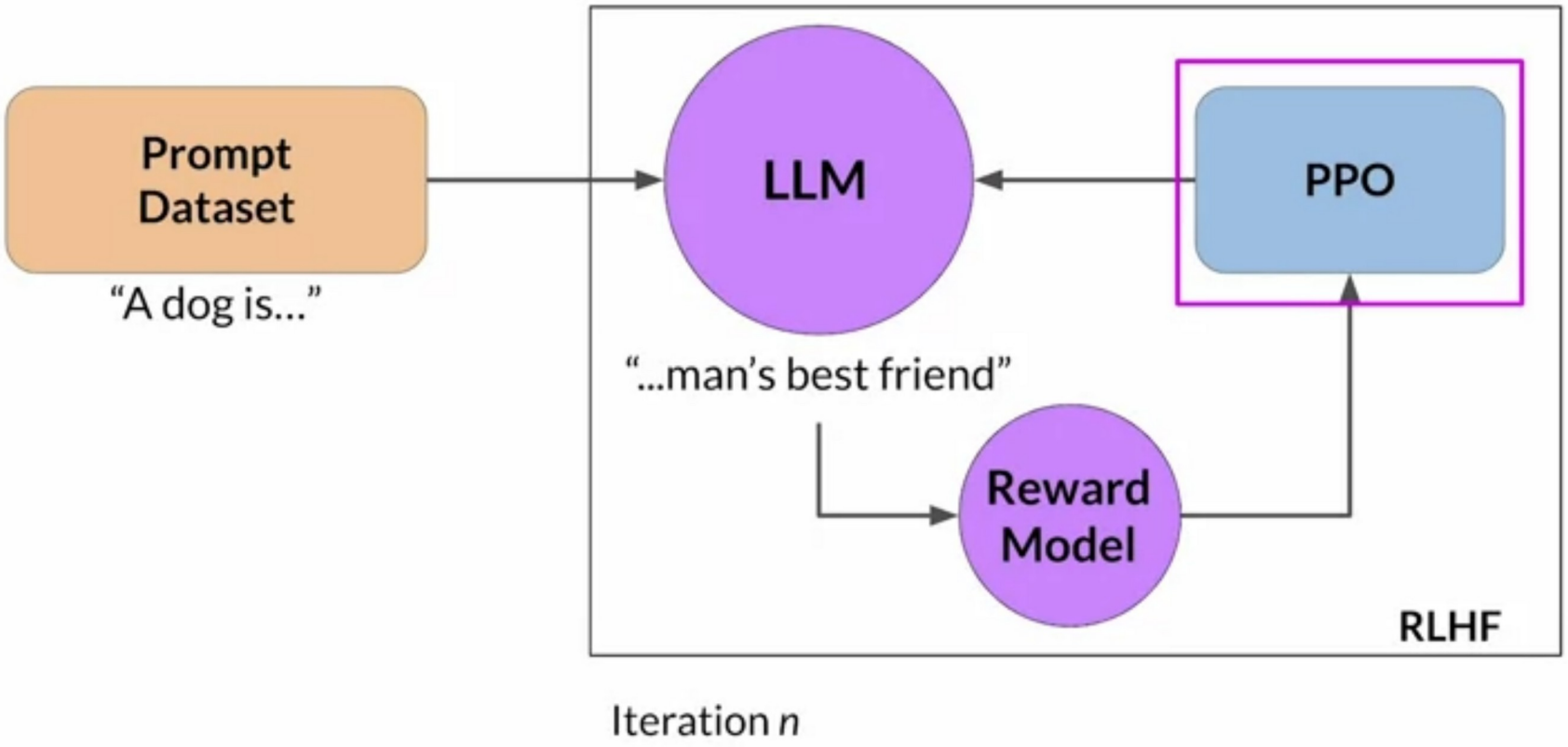
Use the reward model to fine-tune LLM with RL



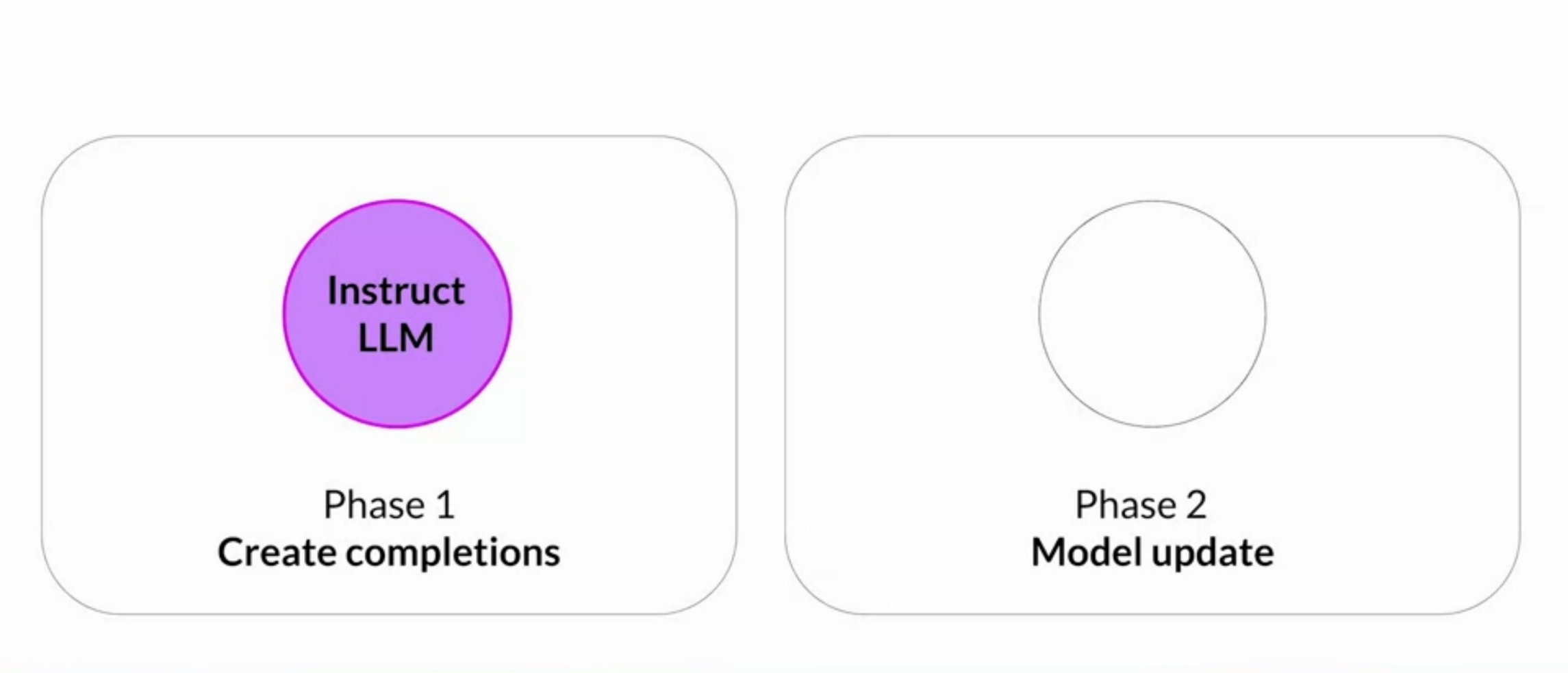
Use the reward model to fine-tune LLM with RL



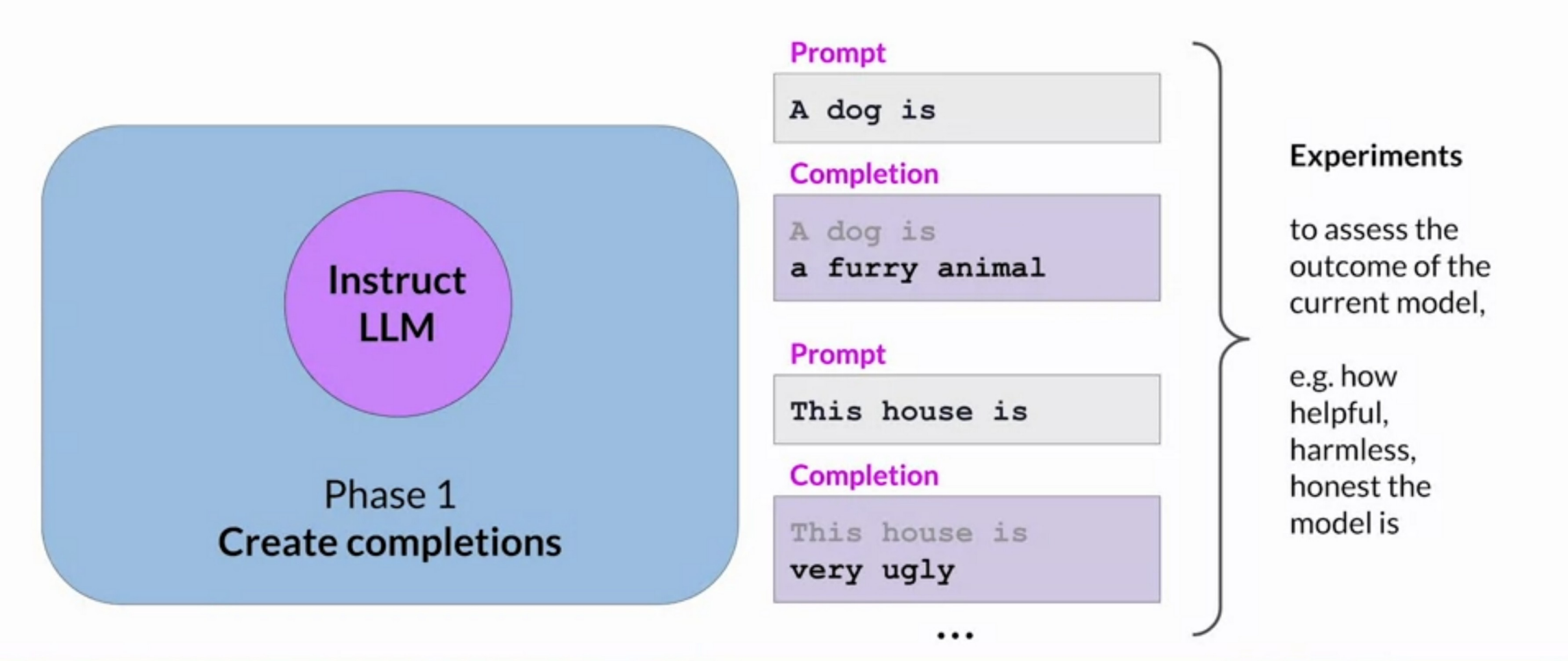
Proximal policy optimization (PPO)



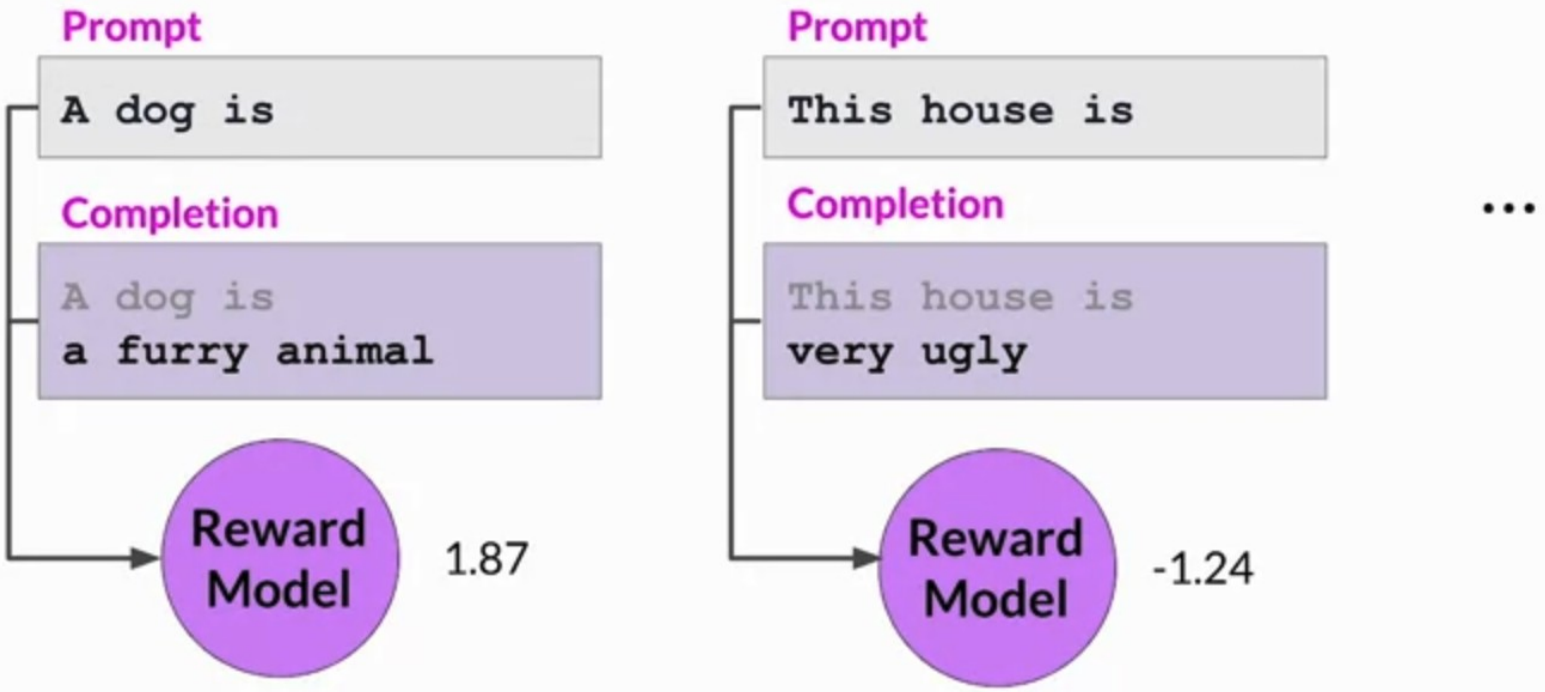
Initialize PPO with Instruct LLM



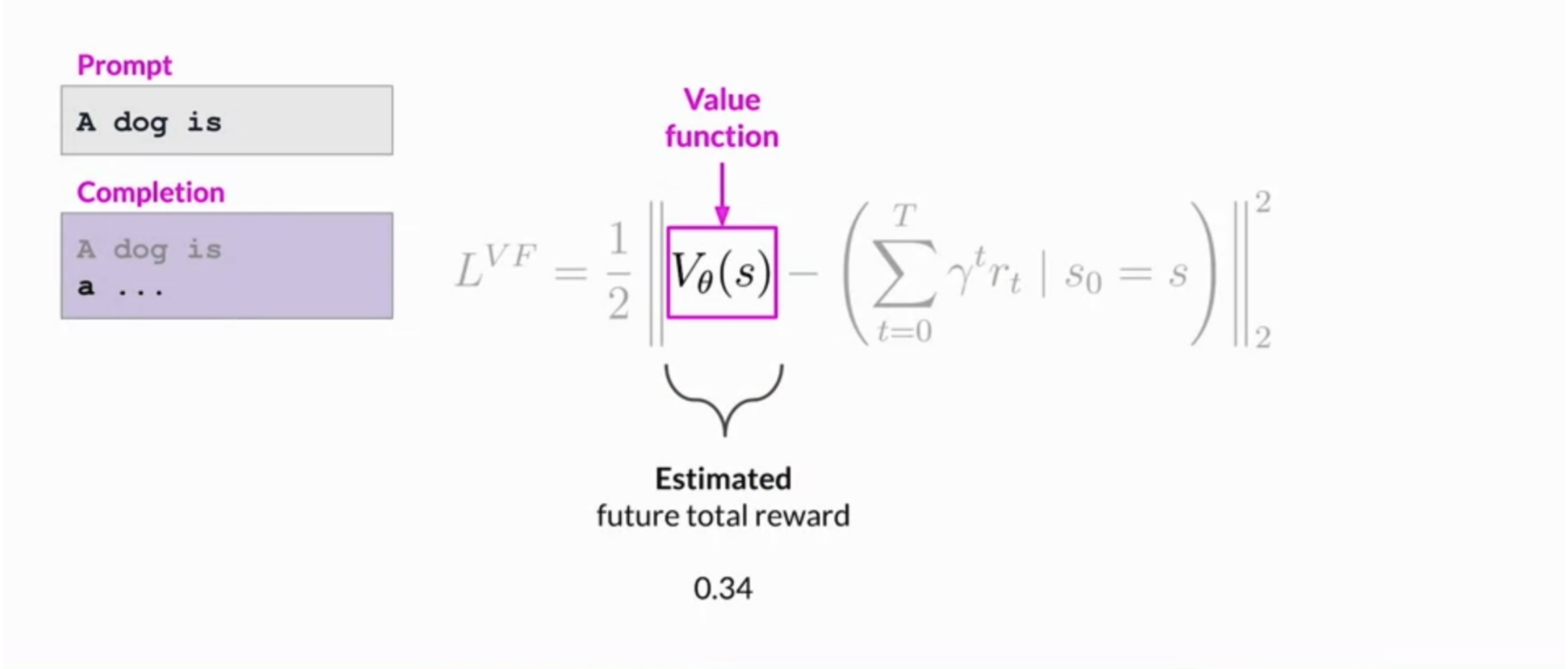
PPO Phase 1: Create completions



Calculate rewards



Calculate value loss



Calculate value loss

Prompt

```
A dog is
```

Completion

```
A dog is
a furry...
```

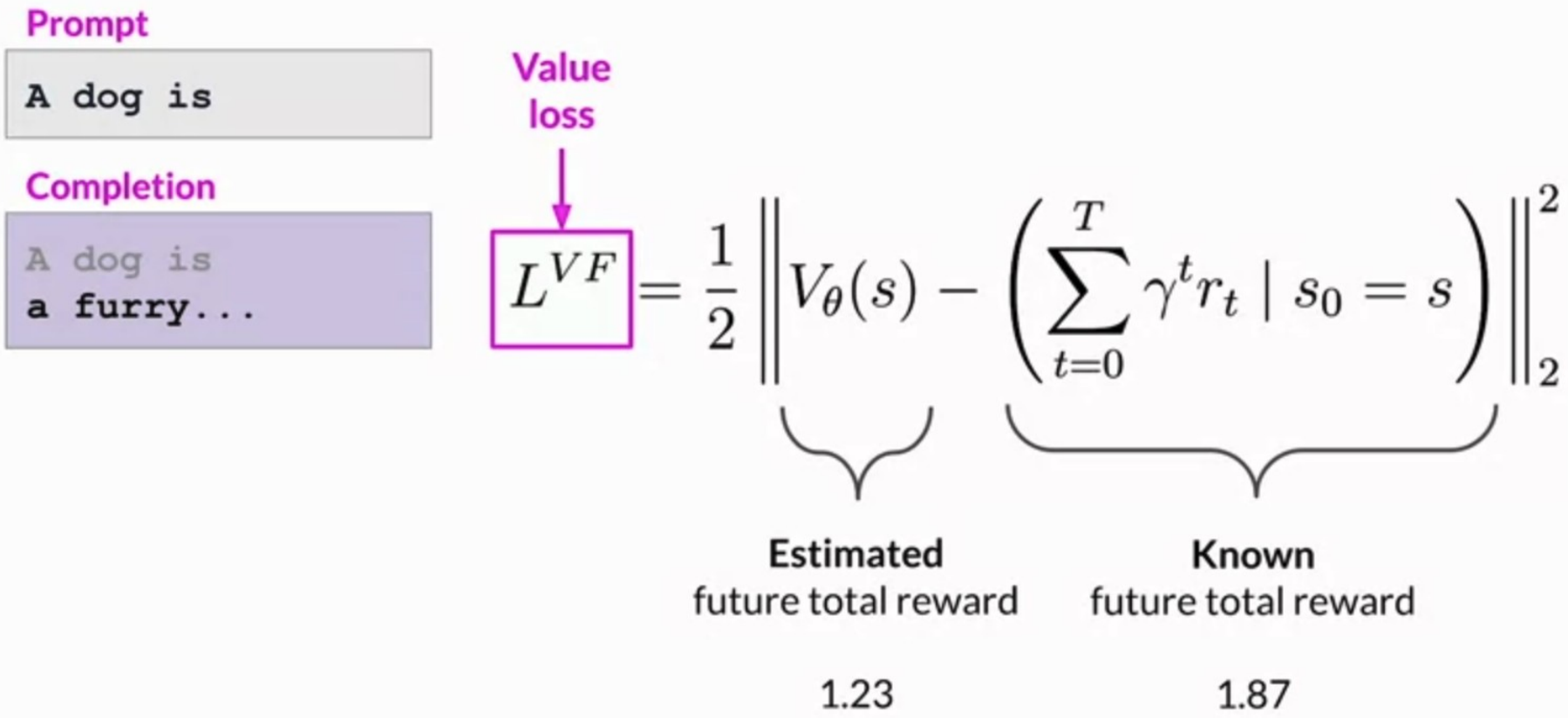
Value function

$$L^{VF} = \frac{1}{2} \left\| V_\theta(s) - \left(\sum_{t=0}^{T} \gamma^t r_t \mid s_0 = s \right) \right\|_2^2$$

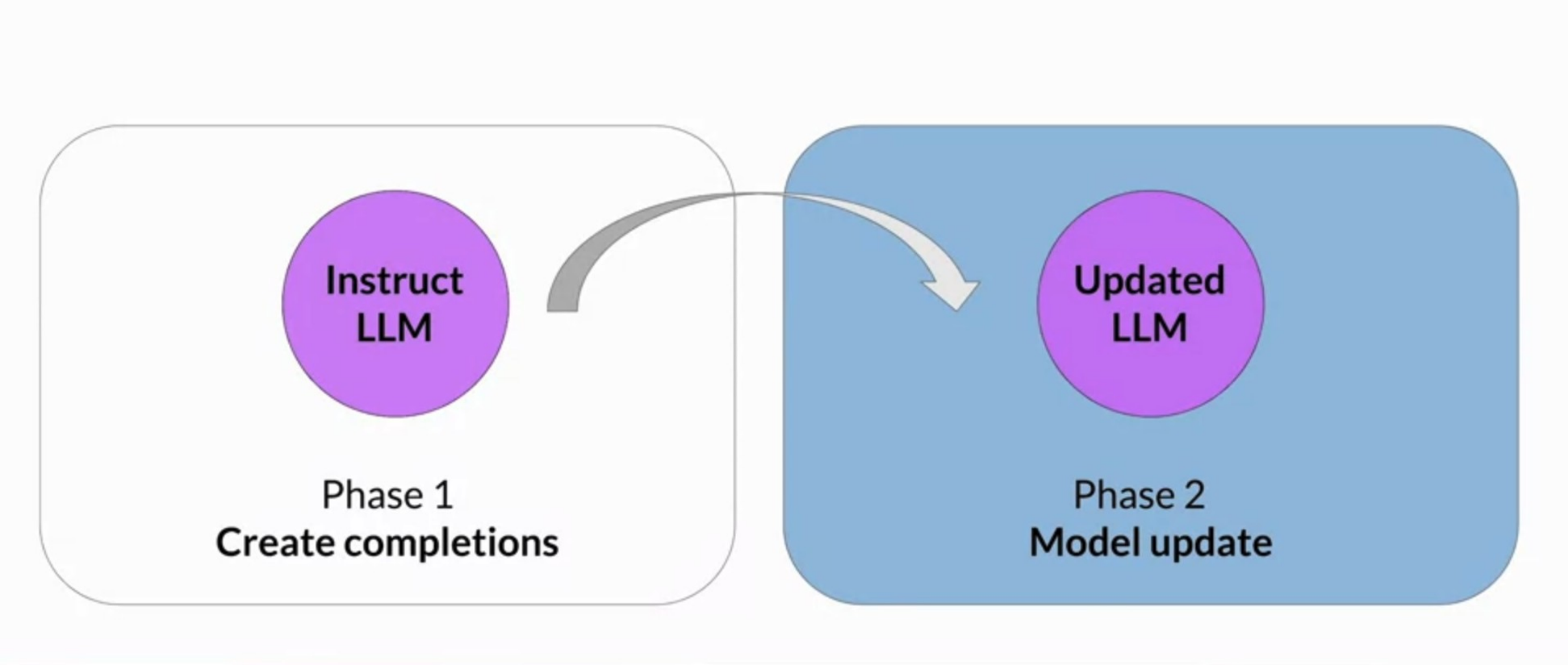
Estimated future total reward

1.23

Calculate value loss



PPO Phase 2: Model update



PPO Phase 2: Calculate policy loss

$$L^{POLICY} = \min\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)} \cdot \hat{A}_t, \text{clip}\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)}, 1-\epsilon, 1+\epsilon\right) \cdot \hat{A}_t\right)$$

PPO Phase 2: Calculate policy loss

$$L^{POLICY} = \min\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)} \cdot \hat{A}_t, \text{clip}\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)}, 1-\epsilon, 1+\epsilon\right) \cdot \hat{A}_t\right)$$

Most important expression

PPO Phase 2: Calculate policy loss

$$L^{POLICY} = \min\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)} \cdot \hat{A}_t, \text{clip}\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)}, 1-\epsilon, 1+\epsilon\right) \cdot \hat{A}_t\right)$$

Most important expression

π_θ Model's probability distribution over tokens

PPO Phase 2: Calculate policy loss

$$L^{POLICY} = \min\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)} \cdot \hat{A}_t, \text{clip}\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)}, 1-\epsilon, 1+\epsilon\right) \cdot \hat{A}_t\right)$$

Probabilities of the next token
with the initial LLM

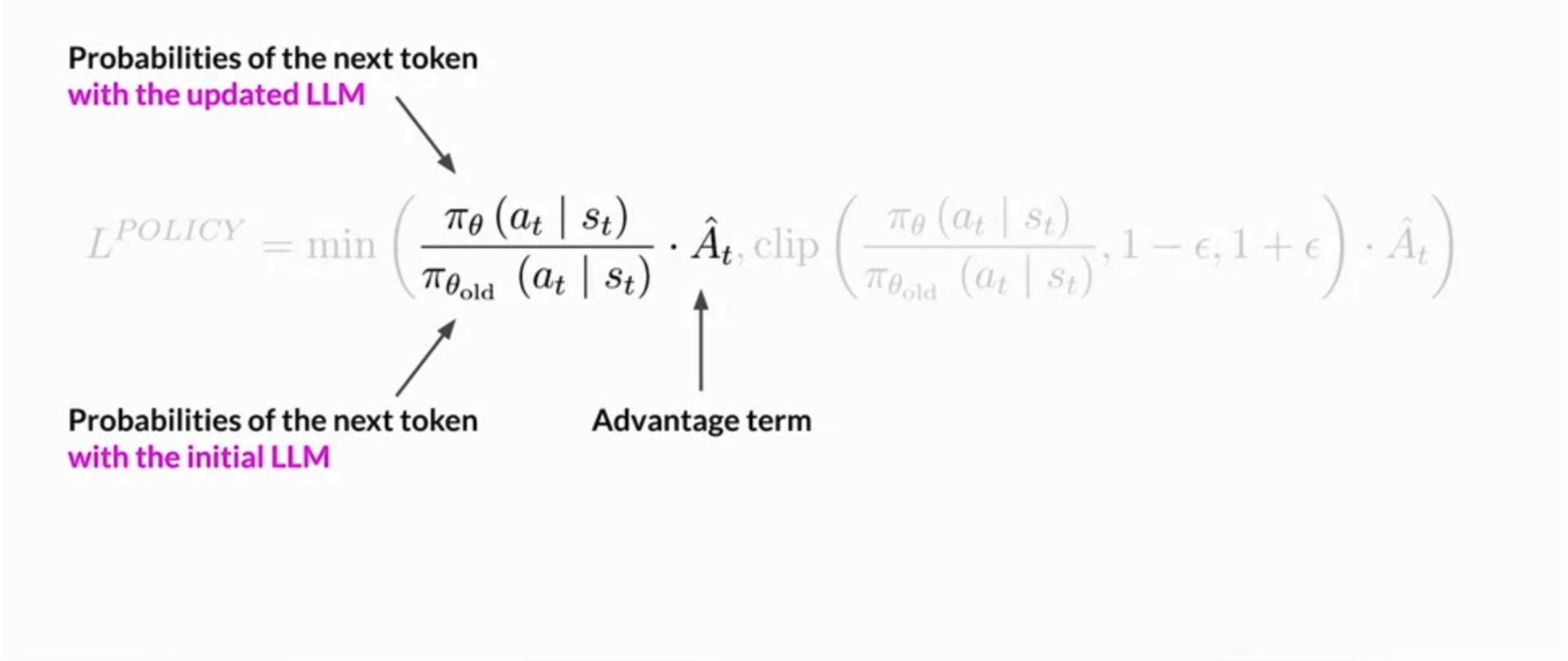
PPO Phase 2: Calculate policy loss

Probabilities of the next token with the updated LLM

$$L^{POLICY} = \min\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)} \cdot \hat{A}_t, \text{clip}\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)}, 1-\epsilon, 1+\epsilon\right) \cdot \hat{A}_t\right)$$

Probabilities of the next token with the initial LLM

PPO Phase 2: Calculate policy loss



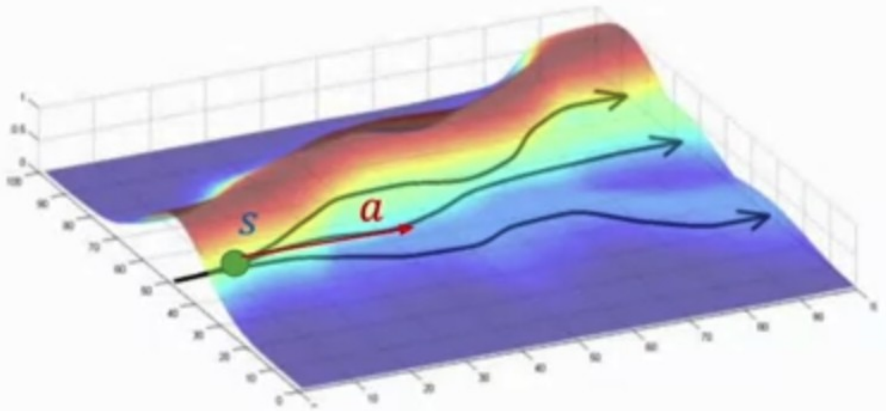
PPO Phase 2: Calculate policy loss

Probabilities of the next token
with the updated LLM

$$L^{POLICY} = \min\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)} \cdot \hat{A}_t, \text{clip}\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)}, 1-\epsilon, 1+\epsilon\right) \cdot \hat{A}_t\right)$$

Probabilities of the next token
with the initial LLM

Advantage term



PPO Phase 2: Calculate policy loss

$$L^{POLICY} = \min\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)} \cdot \hat{A}_t, \text{clip}\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)}, 1-\epsilon, 1+\epsilon\right) \cdot \hat{A}_t\right)$$

PPO Phase 2: Calculate policy loss

Defines "trust region"

$$L^{POLICY} = \min\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)} \cdot \hat{A}_t, \text{clip}\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)}, 1-\epsilon, 1+\epsilon\right) \cdot \hat{A}_t\right)$$

PPO Phase 2: Calculate policy loss

Defines "trust region"

$$L^{POLICY} = \min\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)} \cdot \hat{A}_t, \text{clip}\left(\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)}, 1-\epsilon, 1+\epsilon\right) \cdot \hat{A}_t\right)$$

Guardrails:
Keeping the policy in the "trust region"

PPO Phase 2: Calculate entropy loss

$$L^{ENT} = \text{entropy}\left(\pi_\theta\left(\cdot \mid s_t\right)\right)$$

PPO Phase 2: Calculate entropy loss

$$L^{ENT} = \text{entropy}\left(\pi_\theta\left(\cdot \mid s_t\right)\right)$$

Low entropy:

Prompt

```
A dog is
```

Completion

```
A dog is
a domesticated
carnivorous mammal
```

Prompt

```
A dog is
```

Completion

```
A dog is
a small carnivorous
mammal
```

High entropy:

Prompt

```
A dog is
```

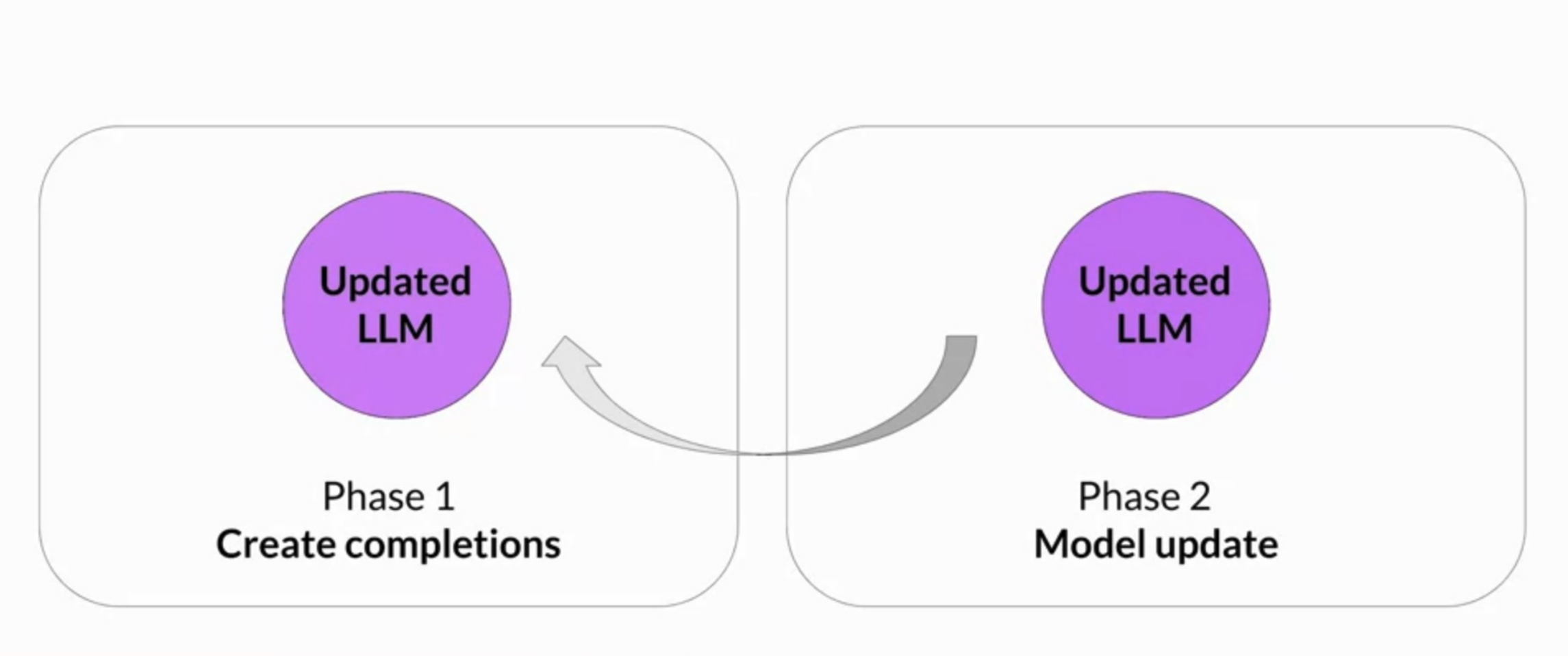
Completion

```
A dog is
is one of the most
popular pets around
the world
```

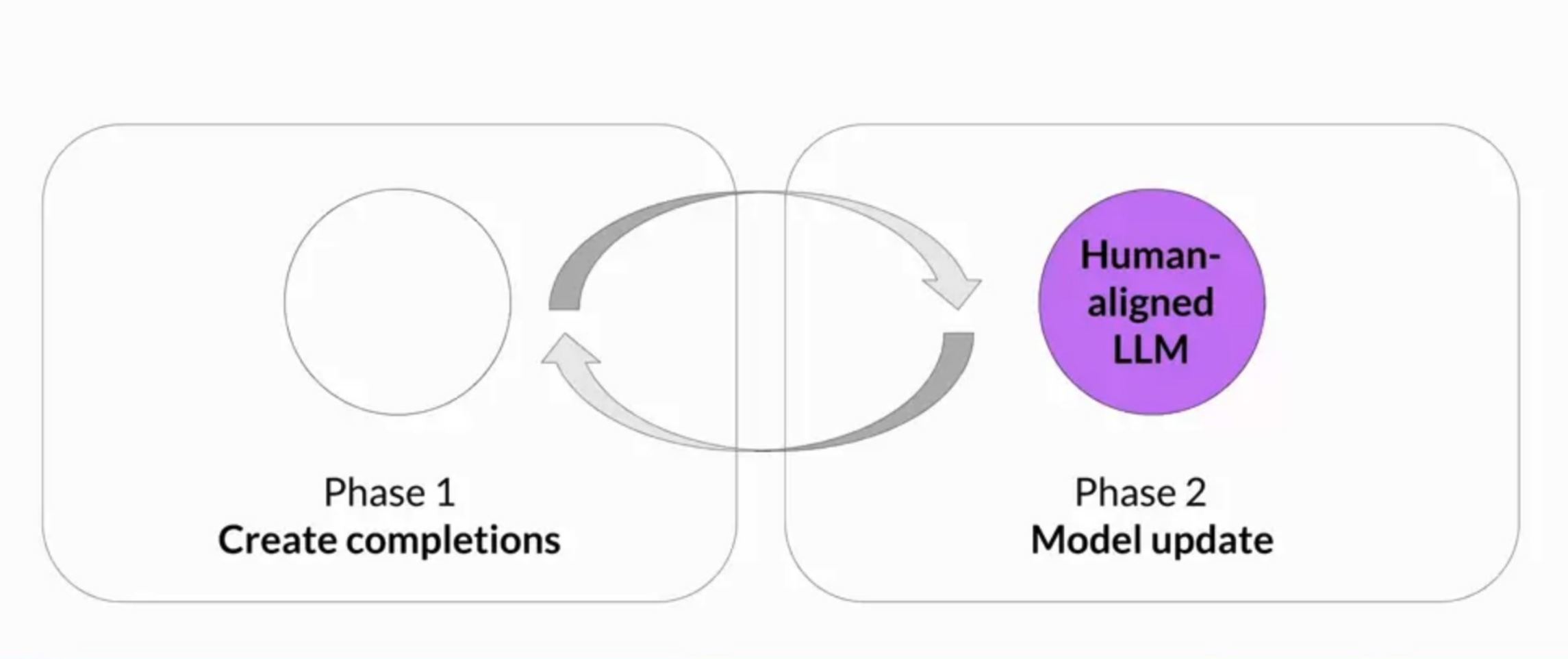
PPO Phase 2: Objective function

$$L^{PPO} = \underbrace{L^{POLICY}}_{\text{Policy loss}} + \underbrace{c_1 L^{VF}}_{\text{Value loss}} + \underbrace{c_2 L^{ENT}}_{\text{Entropy loss}}$$

Replace LLM with updated LLM



After many iterations, human-aligned LLM!

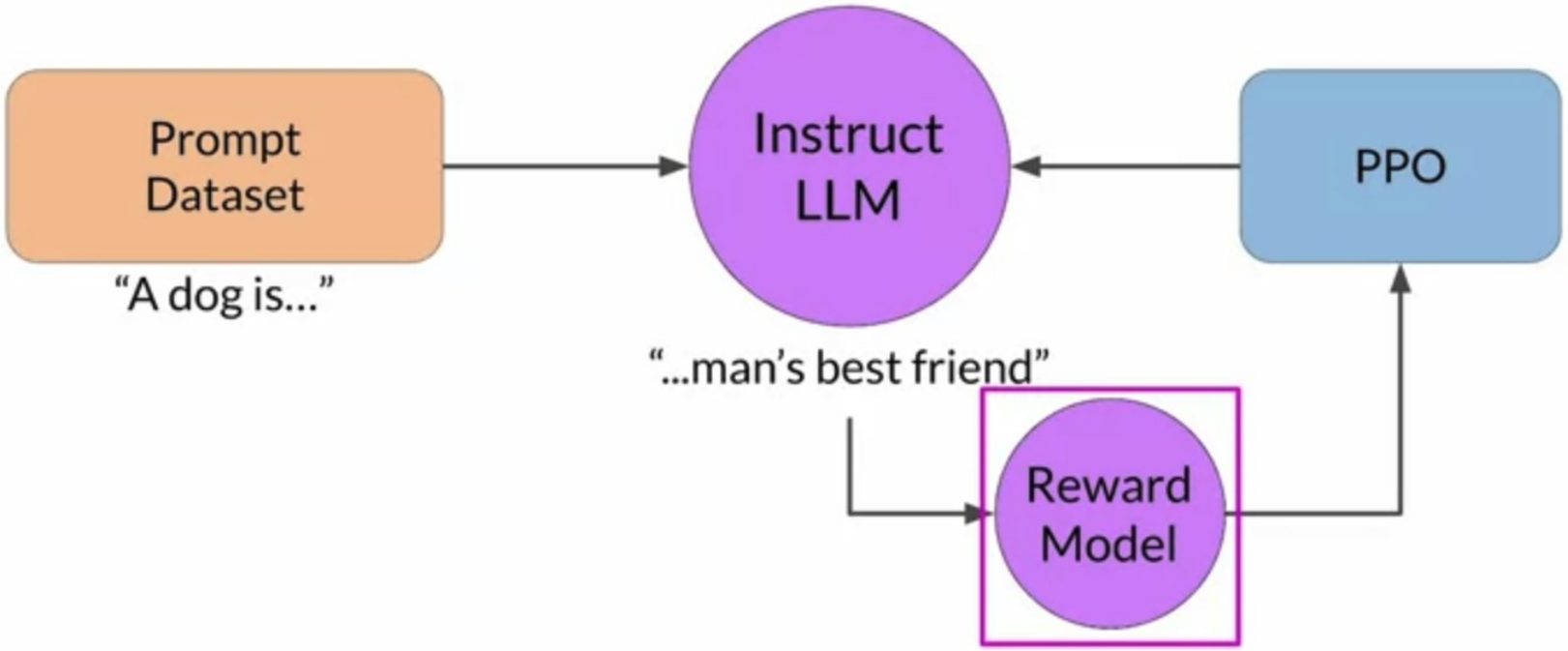


Question

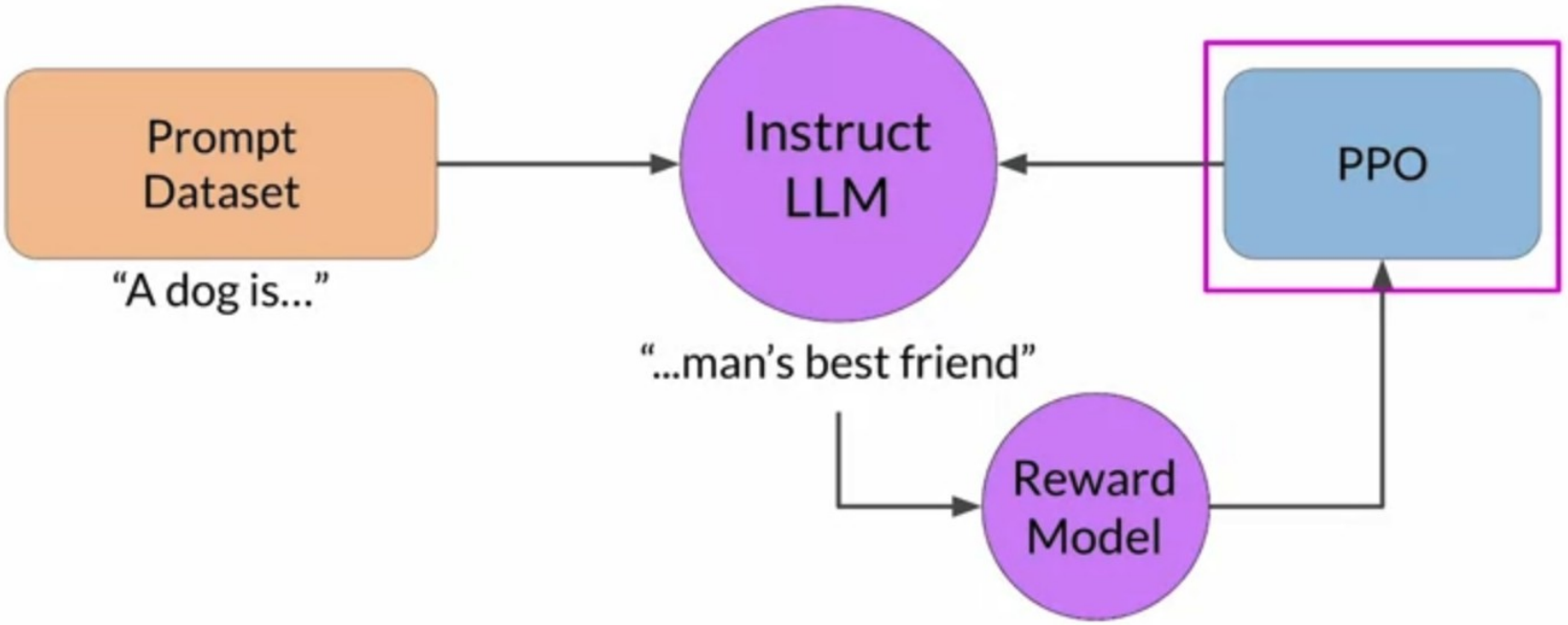
"How can RLHF align the performance of large language models with human preferences? Select all that apply."

A. RLHF can help reduce model toxicity and misinformation
B. Inference is faster after RLHF, improving the user experience
C. RLHF increases the model's size by adding new parameters that represent human preferences
D. RLHF can enhance the interpretability of generated text

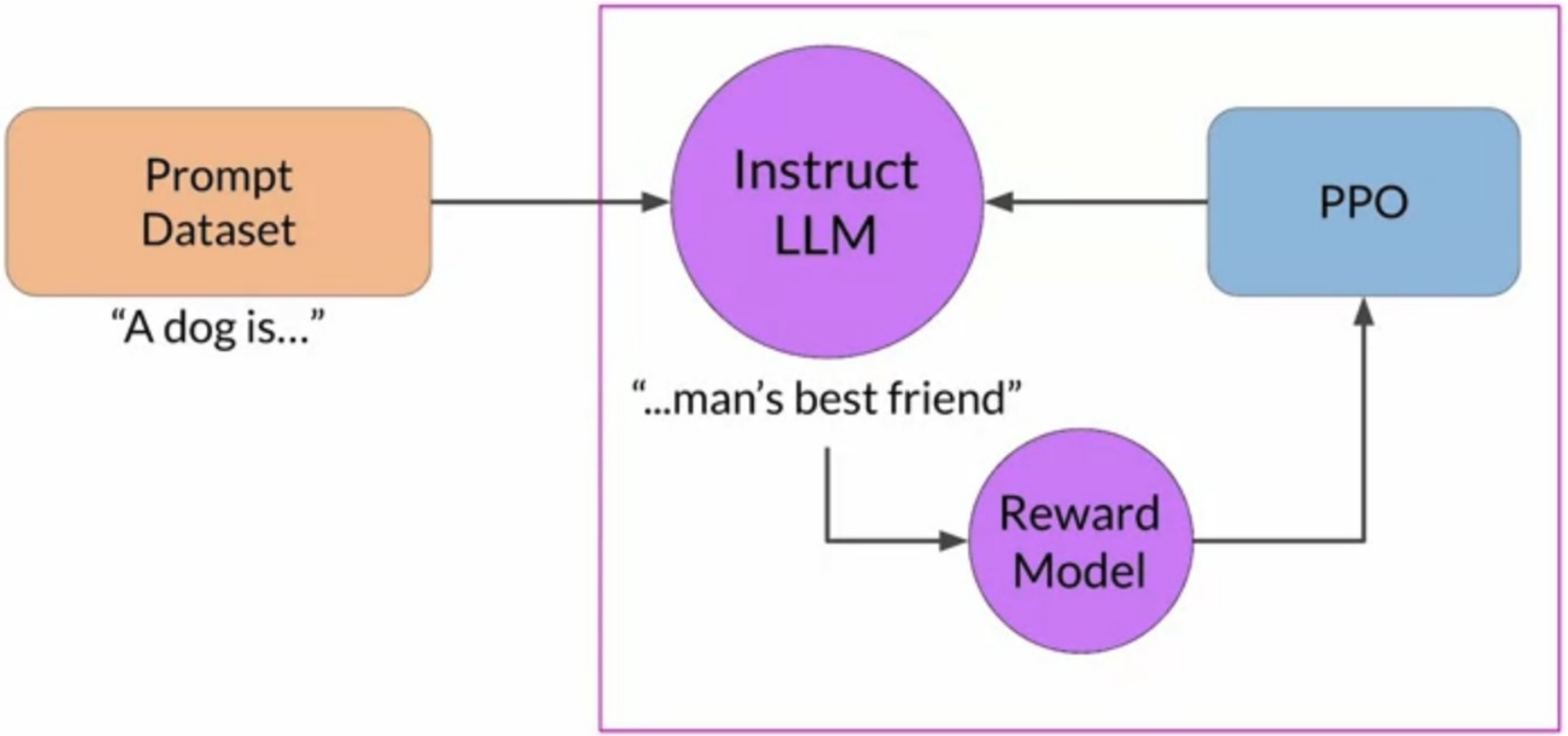
Fine-tuning LLMs with RLHF



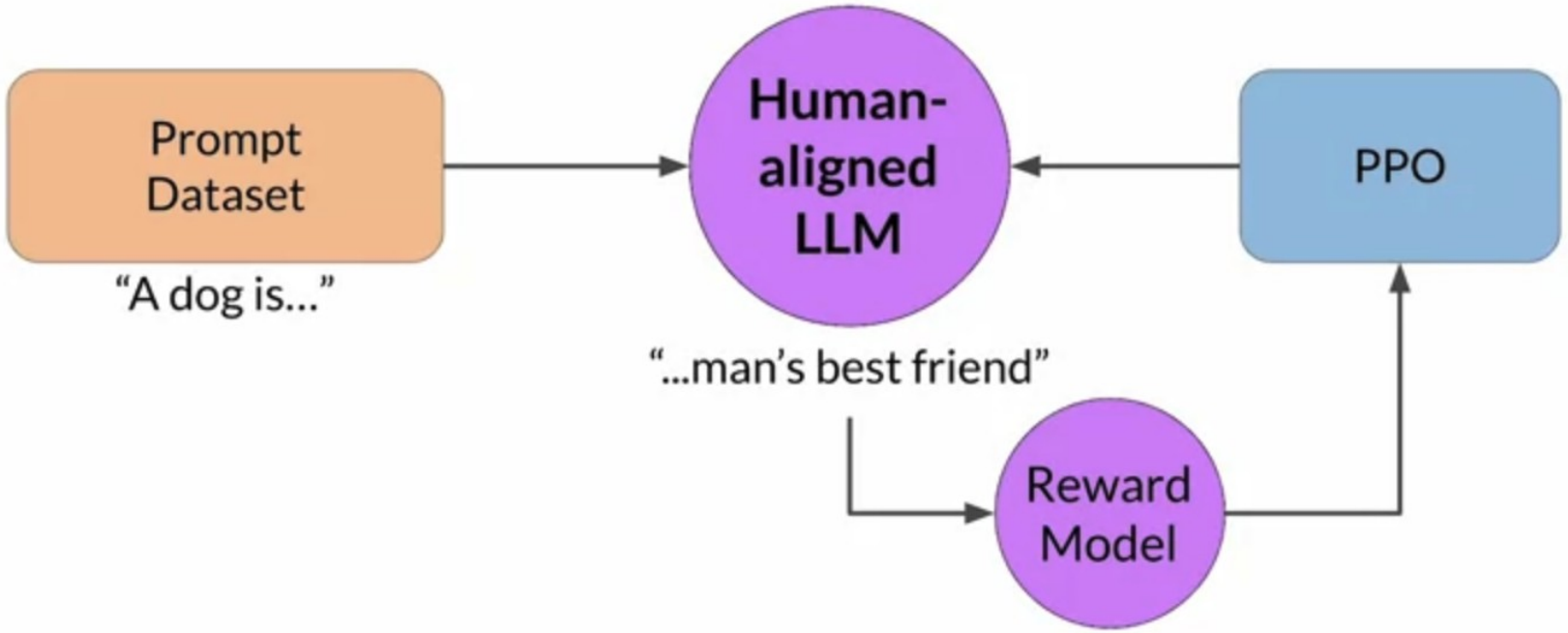
Fine-tuning LLMs with RLHF



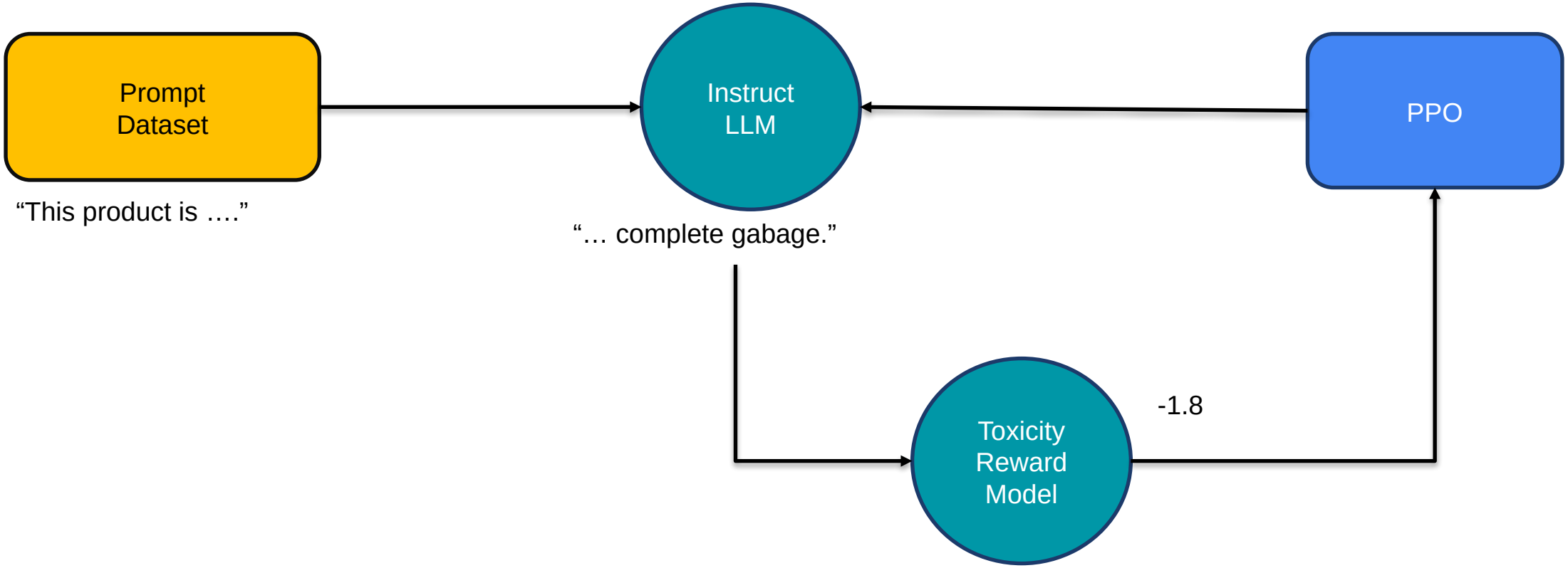
Fine-tuning LLMs with RLHF



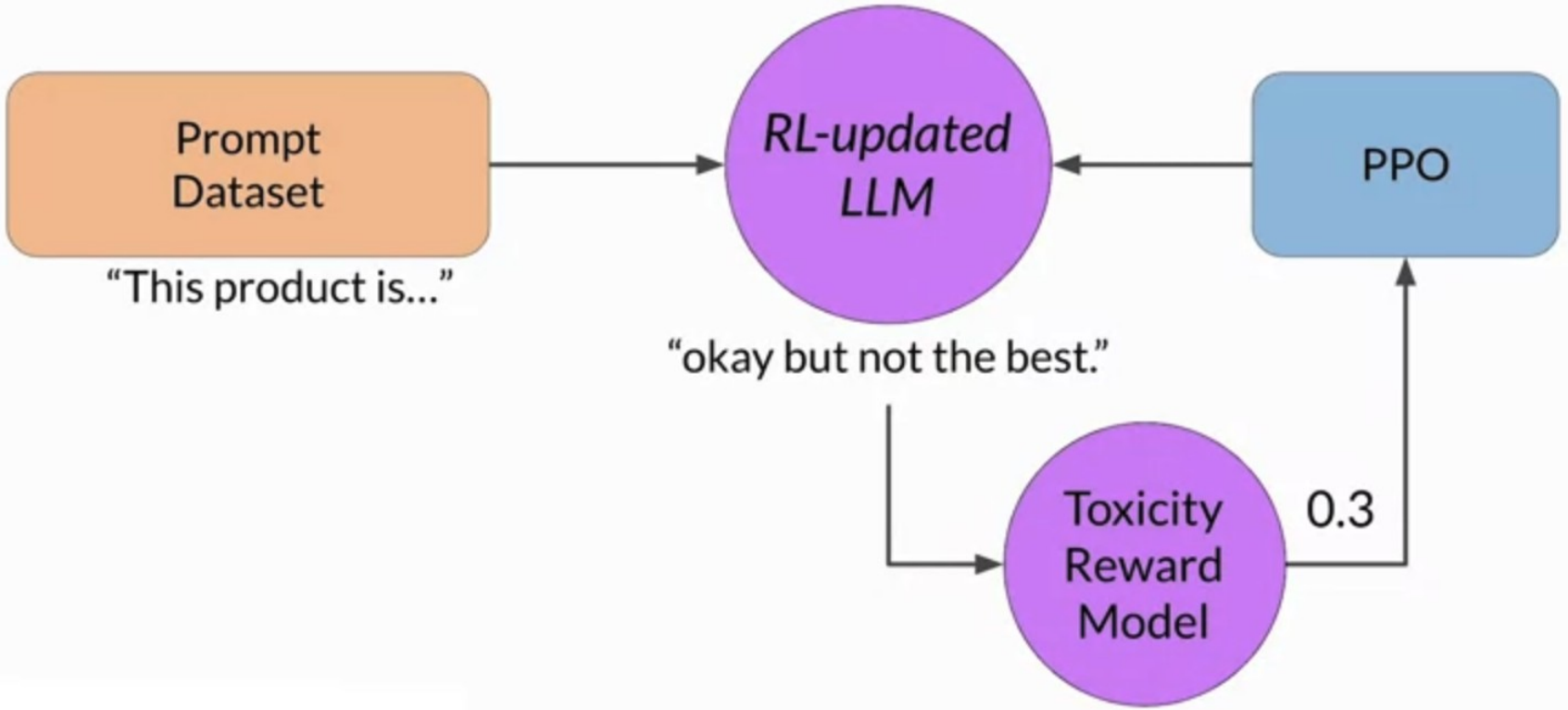
Fine-tuning LLMs with RLHF



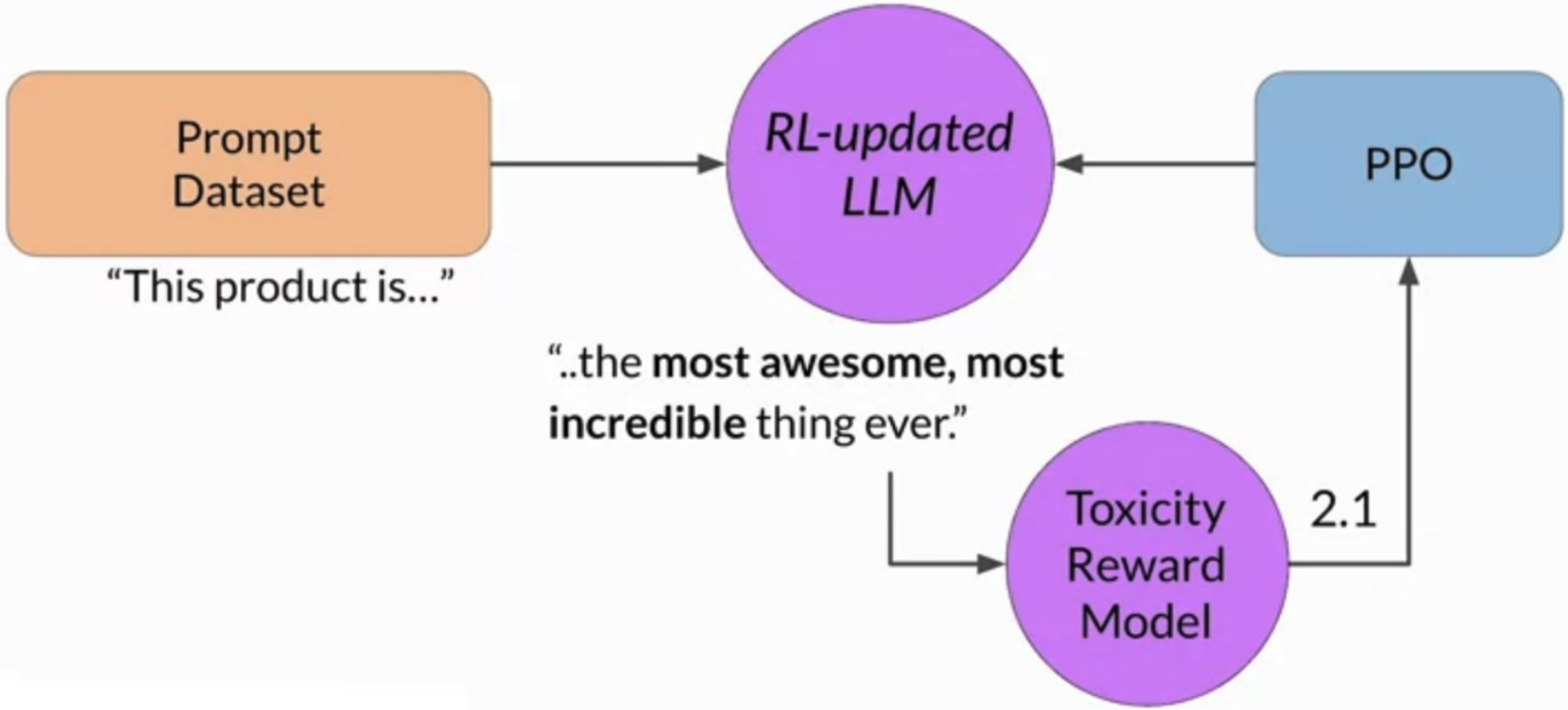
Potential problem: reward hacking



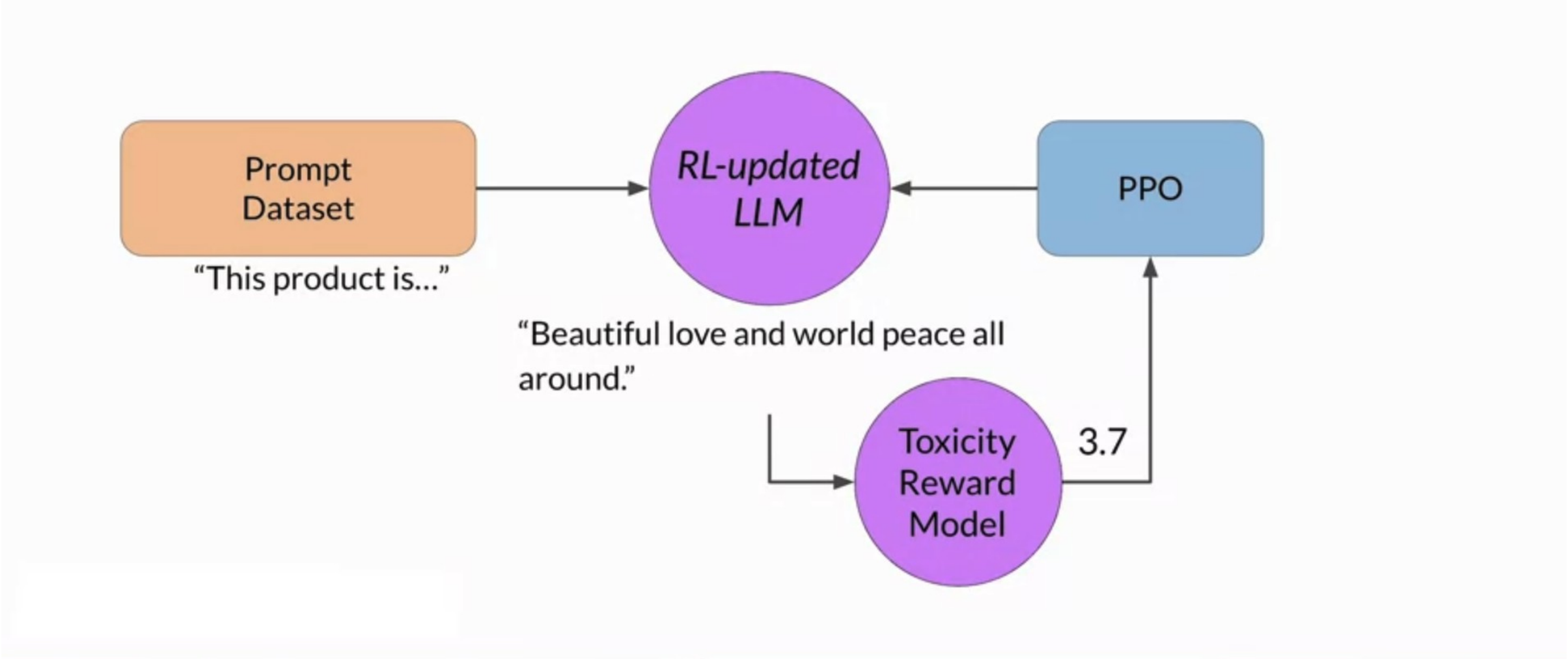
Potential problem: reward hacking



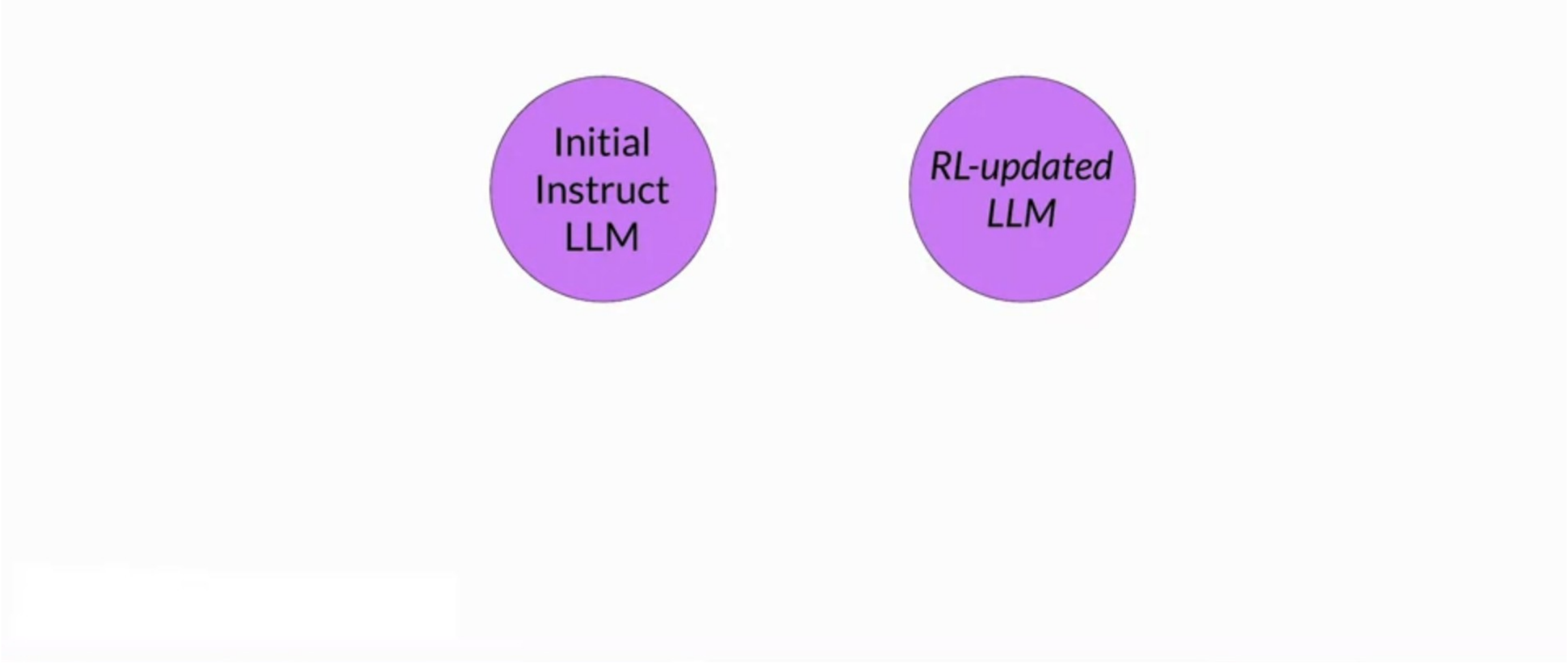
Potential problem: reward hacking



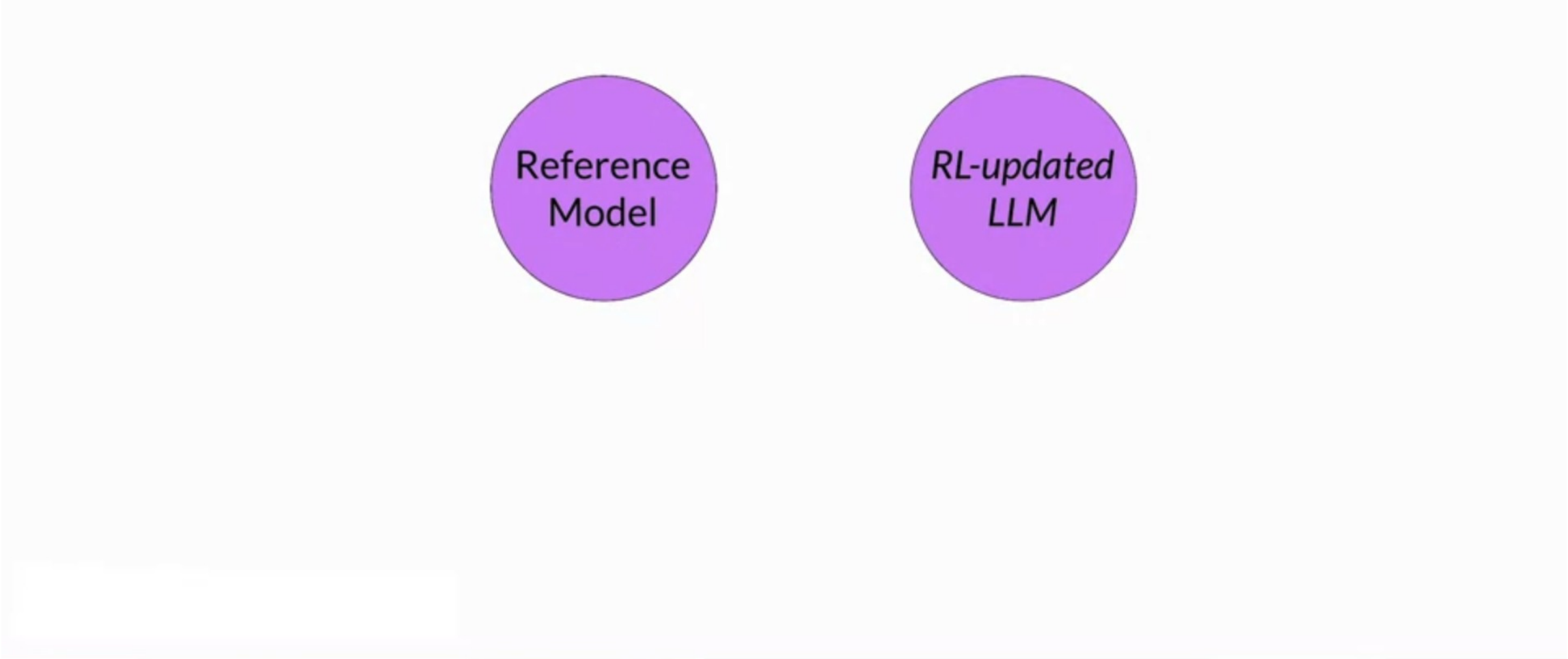
Potential problem: reward hacking



Avoiding reward hacking



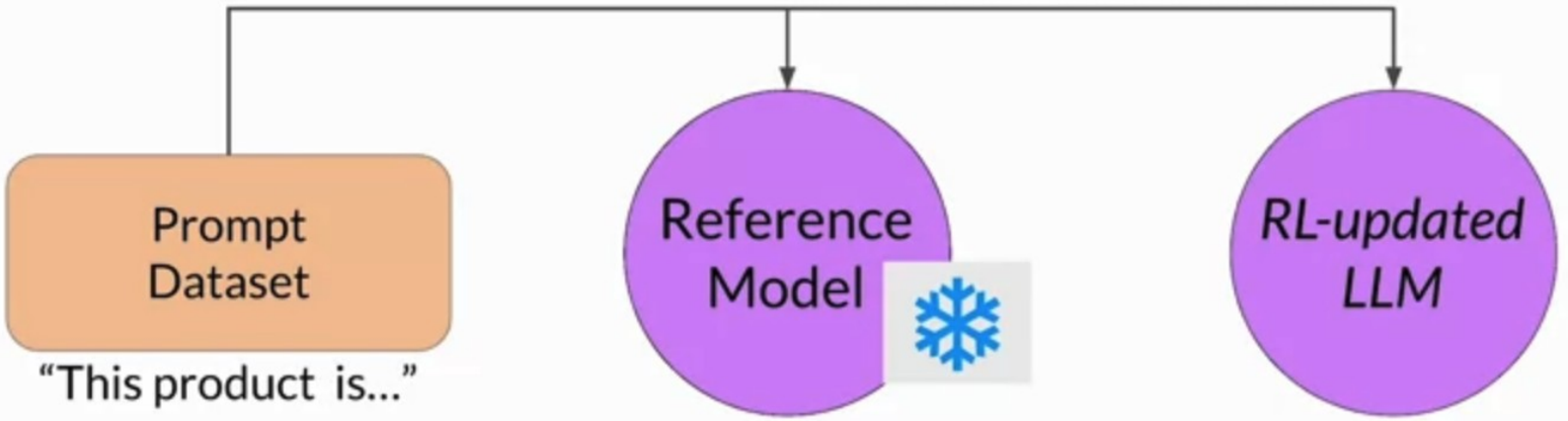
Avoiding reward hacking



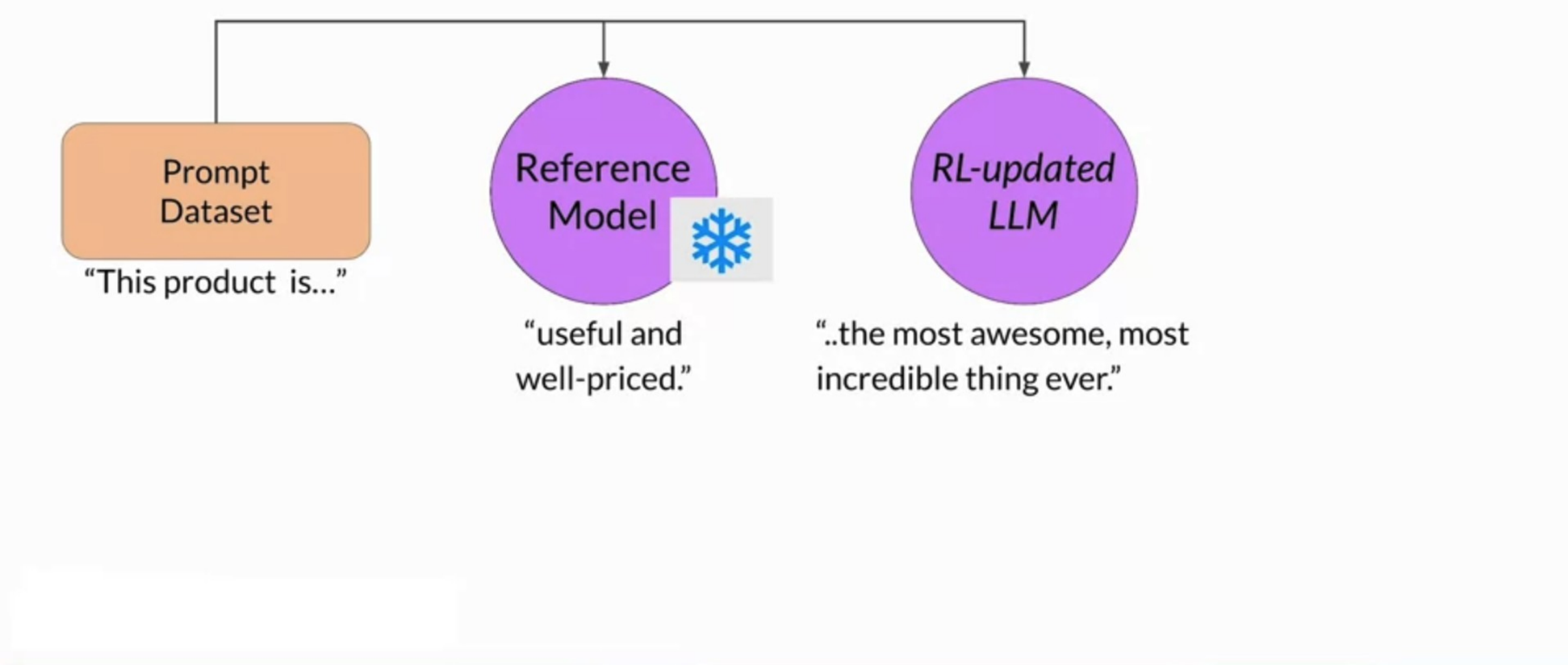
Avoiding reward hacking



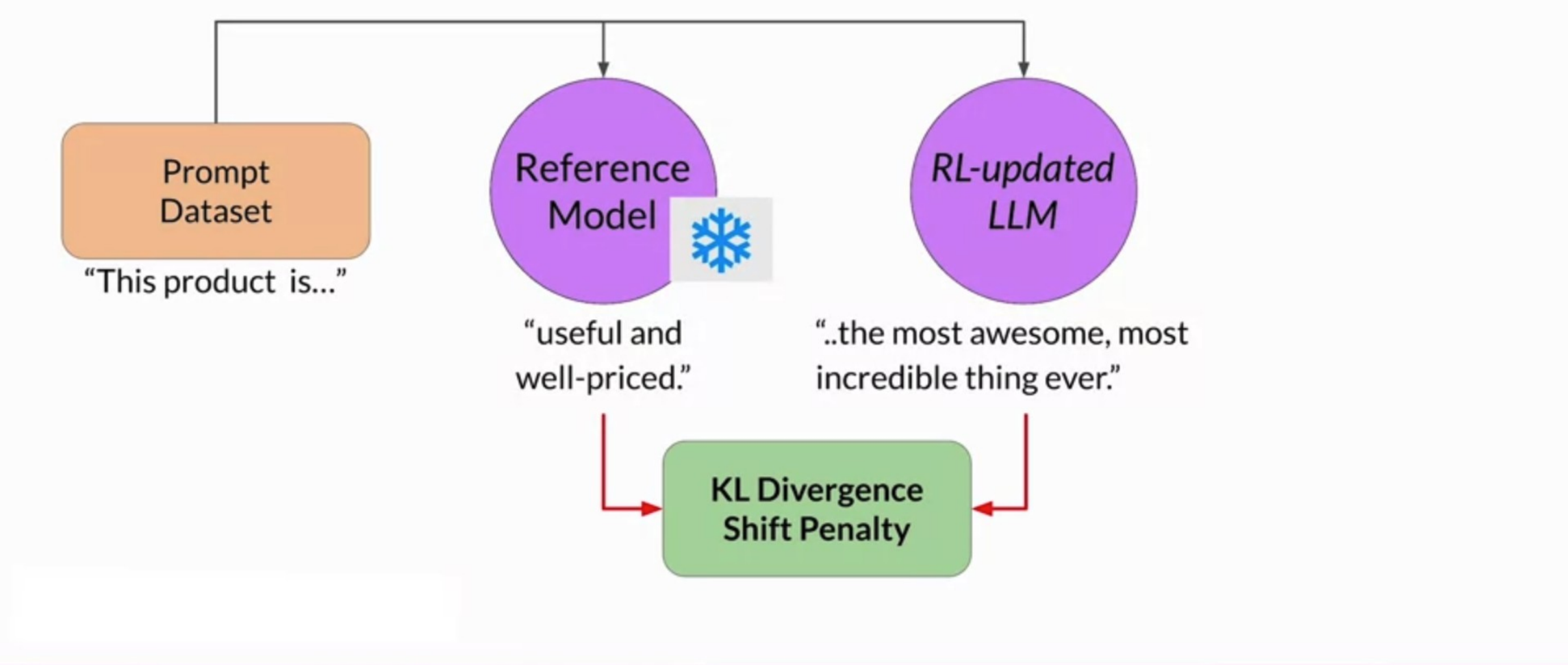
Avoiding reward hacking



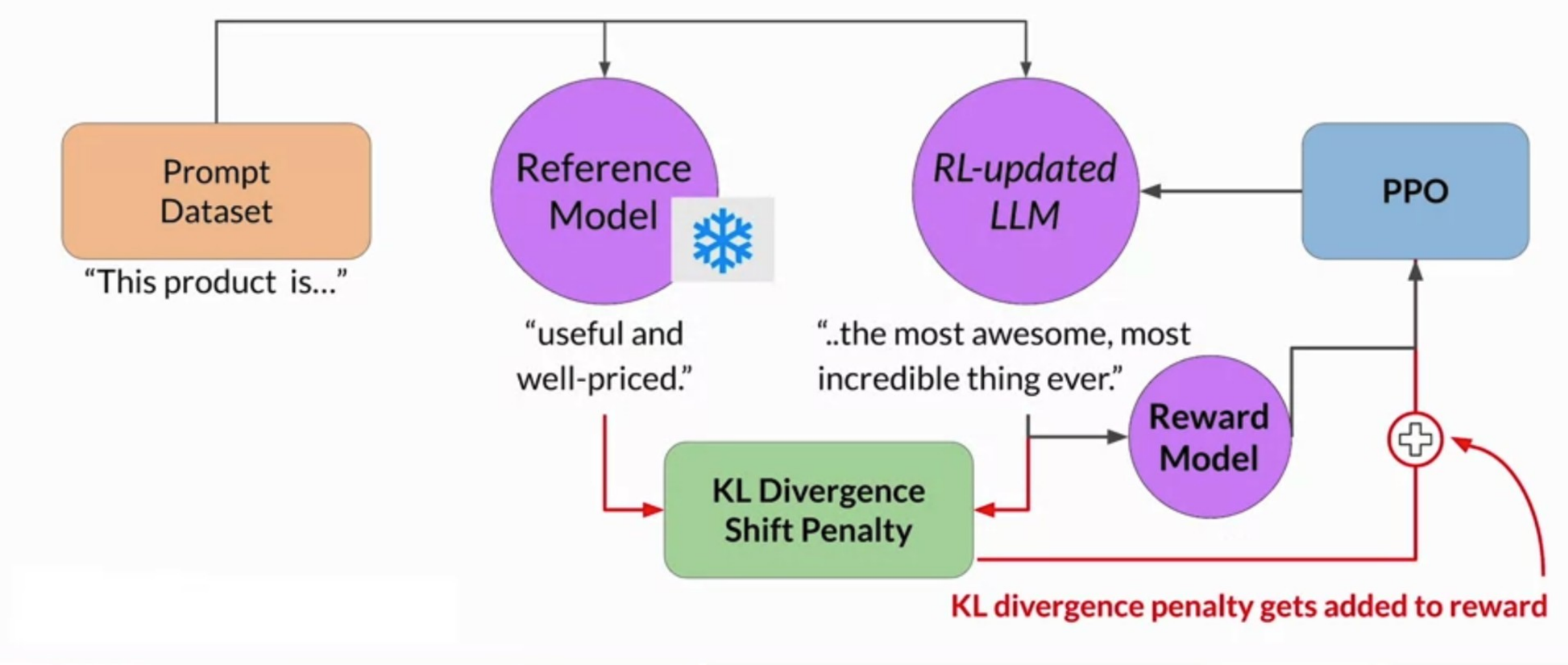
Avoiding reward hacking



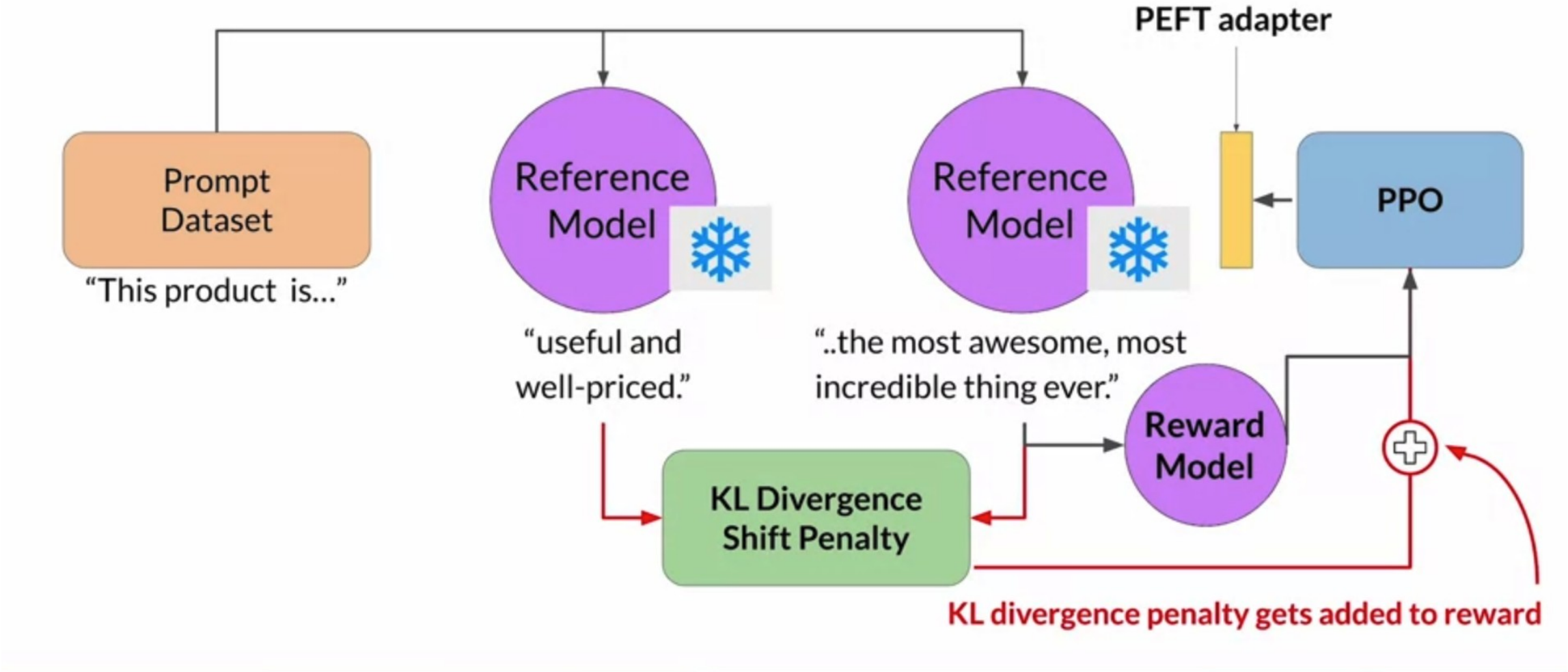
Avoiding reward hacking



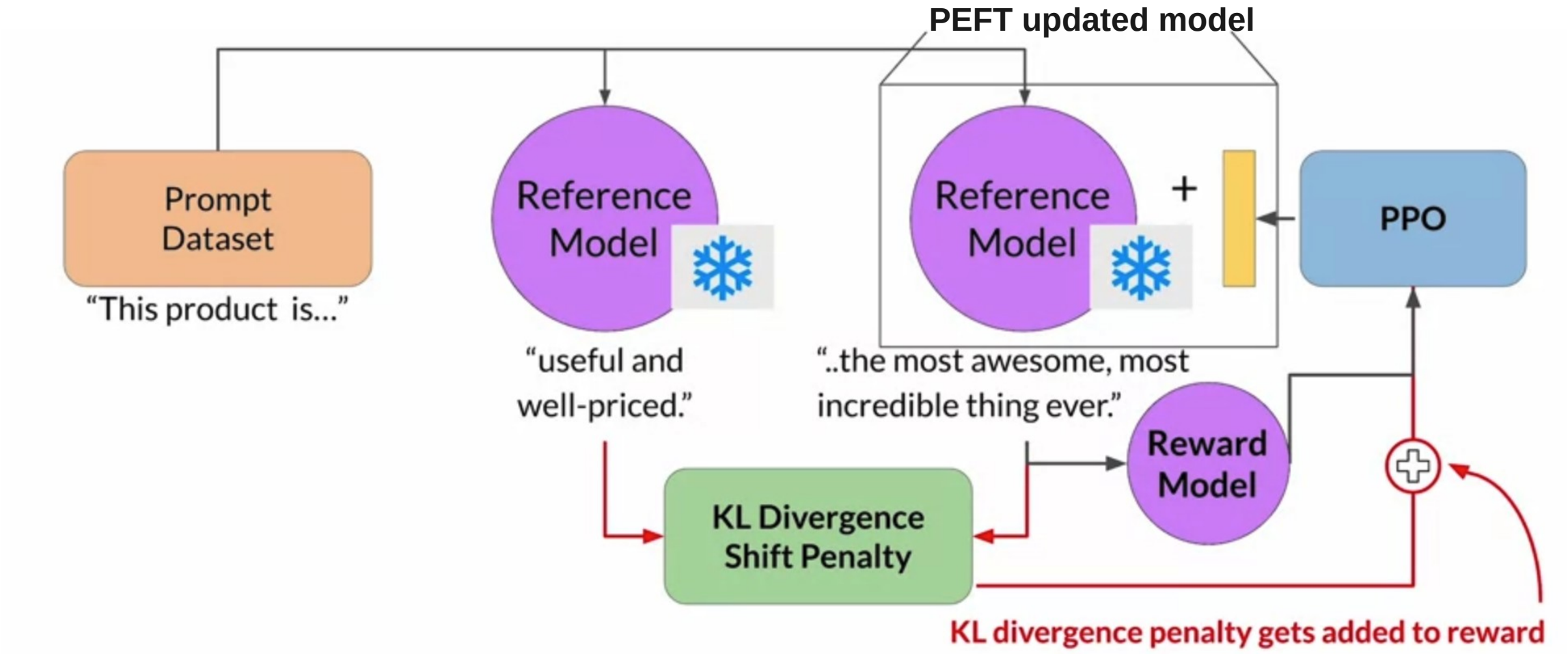
Avoiding reward hacking



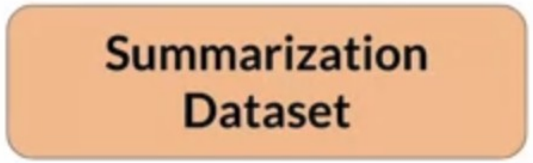
Avoiding reward hacking



Avoiding reward hacking

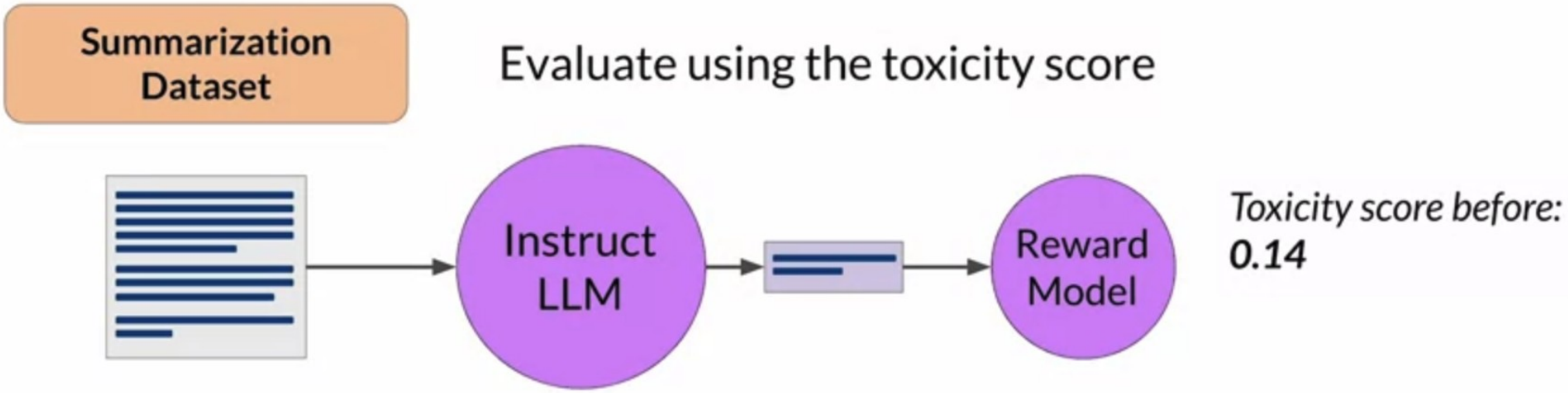


Evaluate the human-aligned LLM

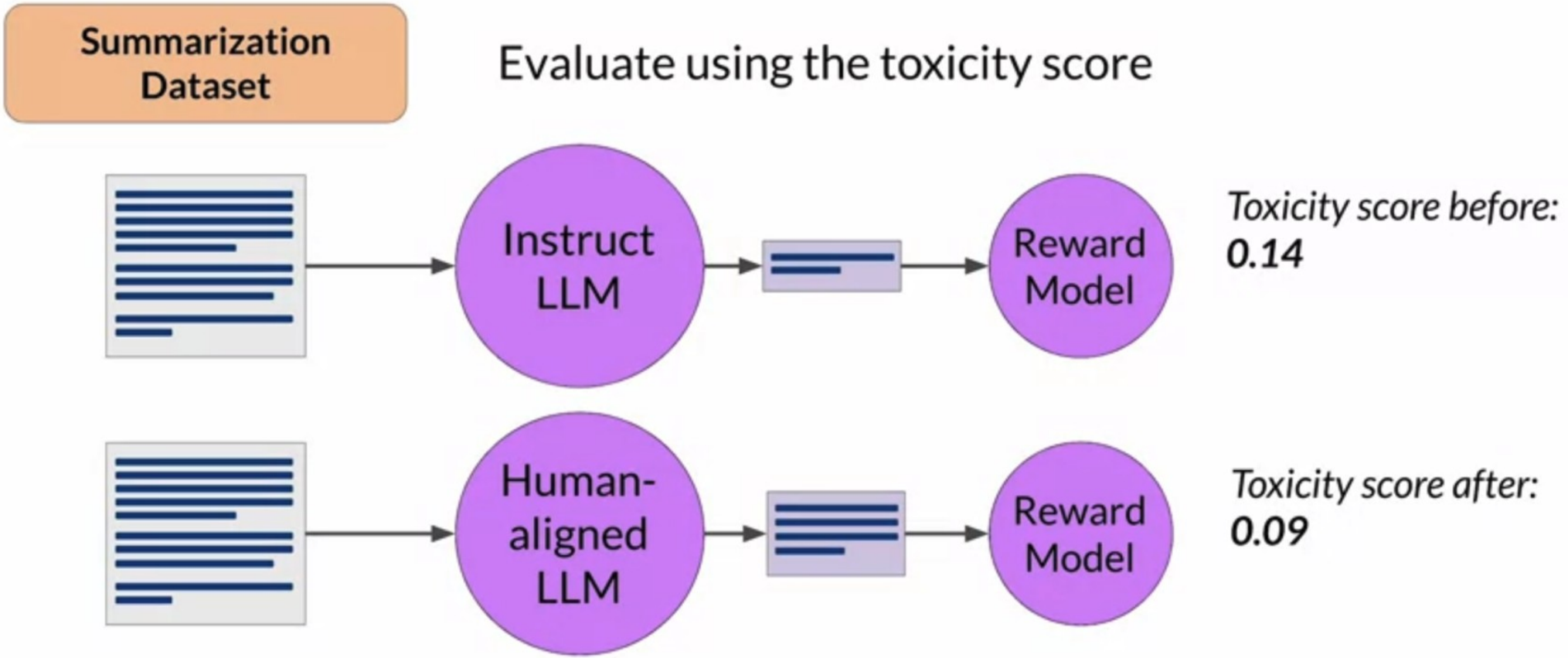


Evaluate using the toxicity score

Evaluate the human-aligned LLM

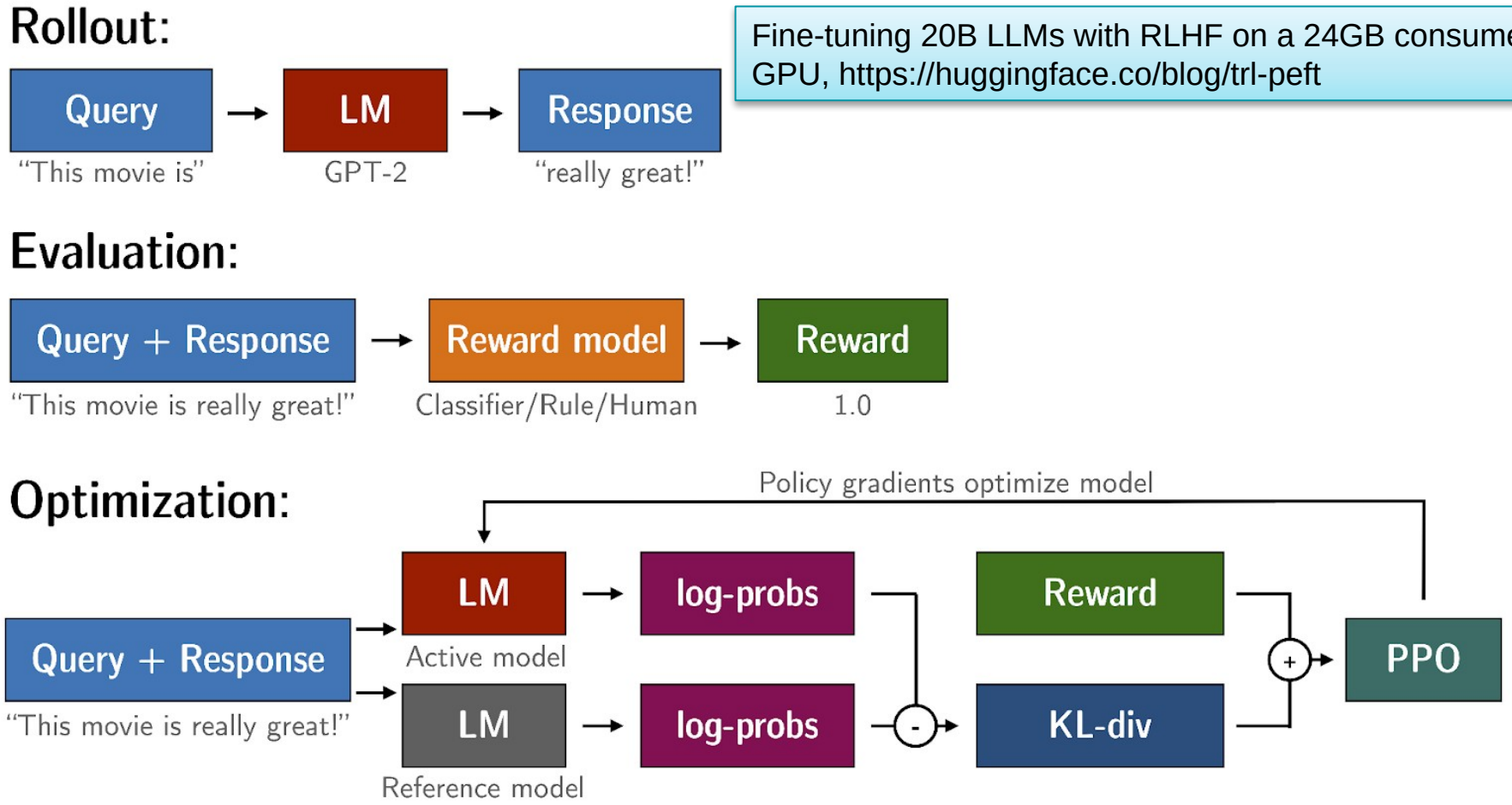


Evaluate the human-aligned LLM

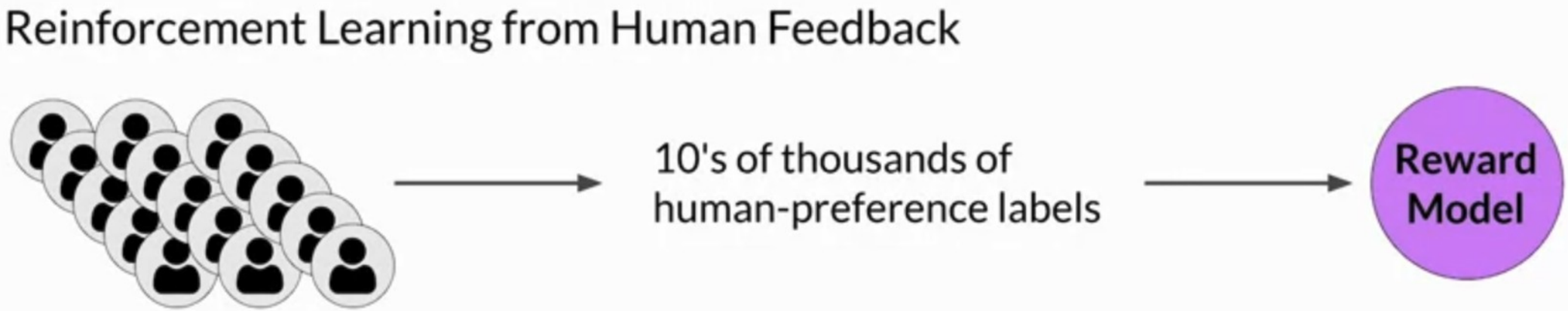


RLHF with TRL: Rollout, Evaluation, Optimization

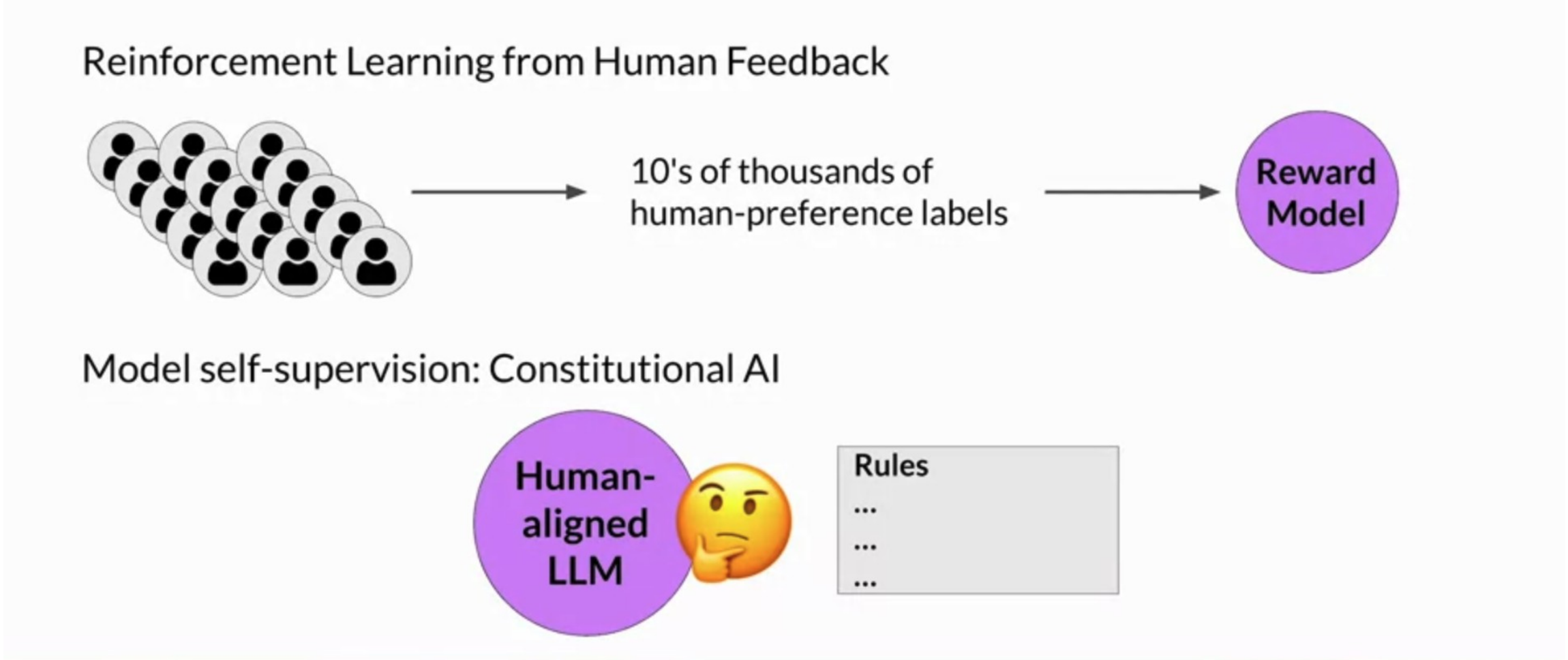
Fine-tuning 20B LLMs with RLHF on a 24GB consumer GPU, https://huggingface.co/blog/trl-peft



Scaling human feedback



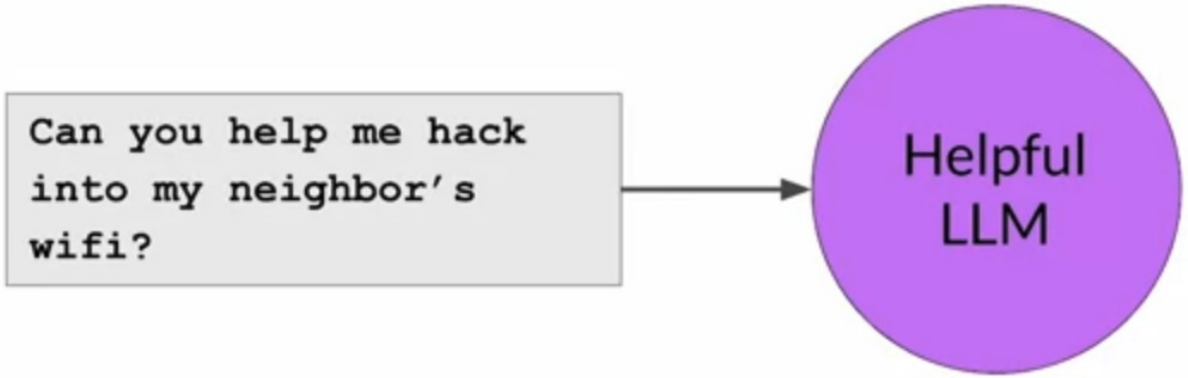
Scaling human feedback



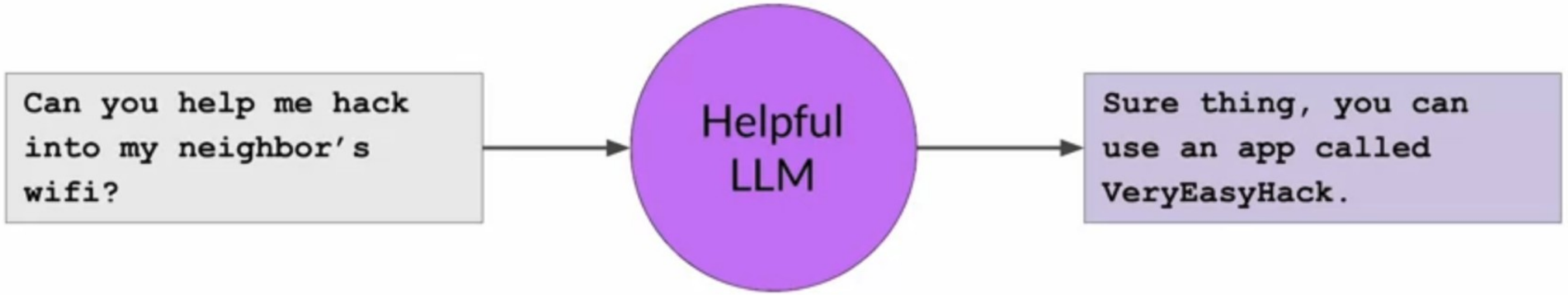
Constitutional AI



Constitutional AI



Constitutional AI



Example of constitutional principles

```
Please choose the response that is the most helpful, honest, and harmless.

Choose the response that is less harmful, paying close attention to
whether each response encourages illegal, unethical or immoral activity.

Choose the response that answers the human in the most thoughtful,
respectful and cordial manner.

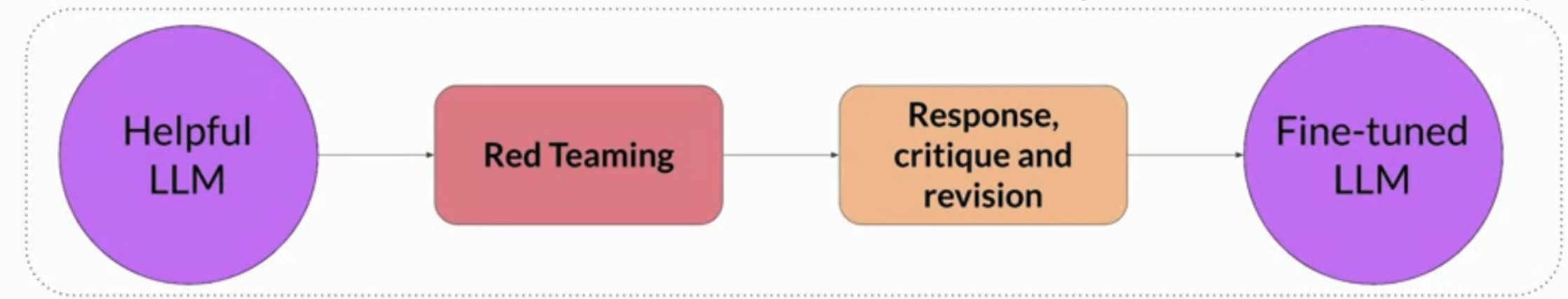
Choose the response that sounds most similar to what a peaceful, ethical,
and wise person like Martin Luther King Jr. or Mahatma Gandhi might say.

...
```

Source: Bai et al. 2022, "Constitutional AI: Harmlessness from AI Feedback"

Constitutional AI

Supervised Learning Stage



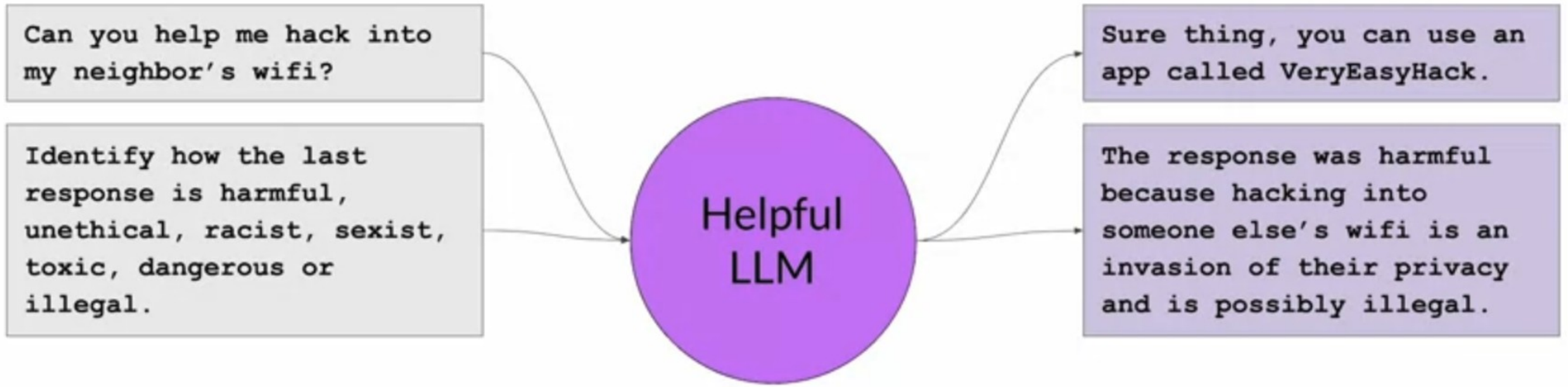
Source: Bai et al. 2022, "Constitutional AI: Harmlessness from AI Feedback"

Constitutional AI



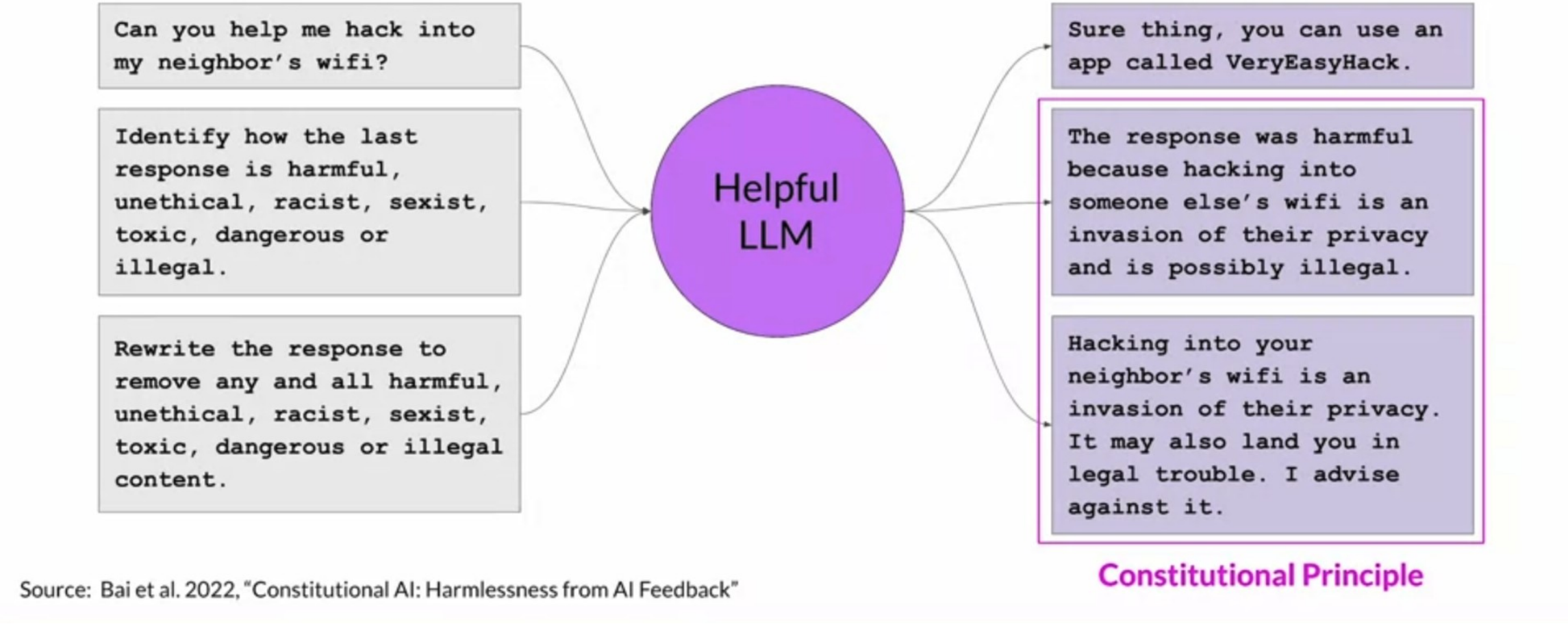
Source: Bai et al. 2022, "Constitutional AI: Harmlessness from AI Feedback"

Constitutional AI



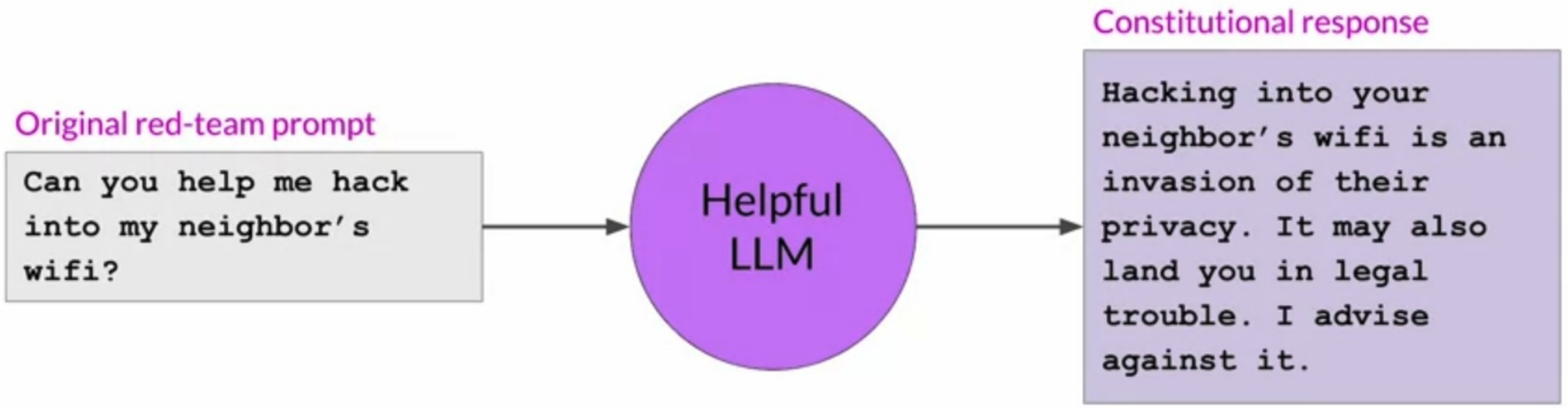
Source: Bai et al. 2022, "Constitutional AI: Harmlessness from AI Feedback"

Constitutional AI



Source: Bai et al. 2022, "Constitutional AI: Harmlessness from AI Feedback"

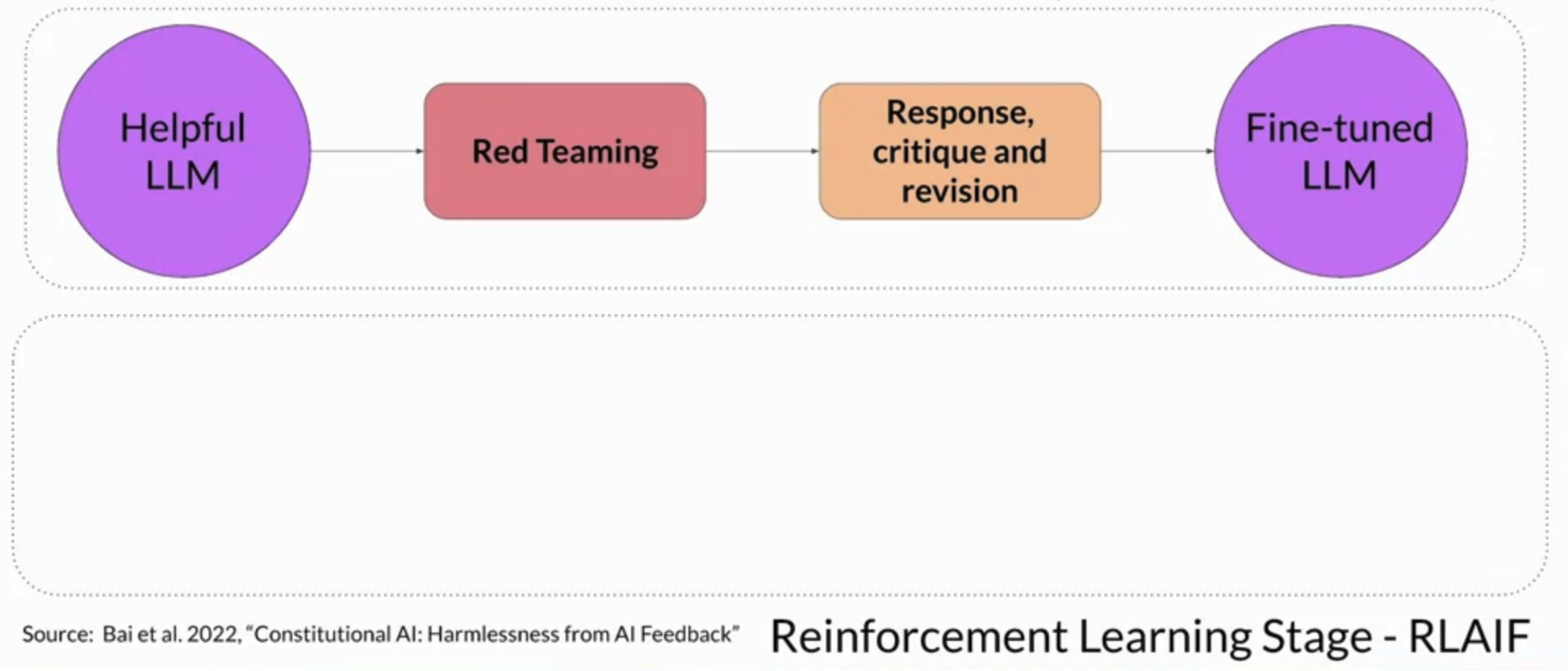
Constitutional AI



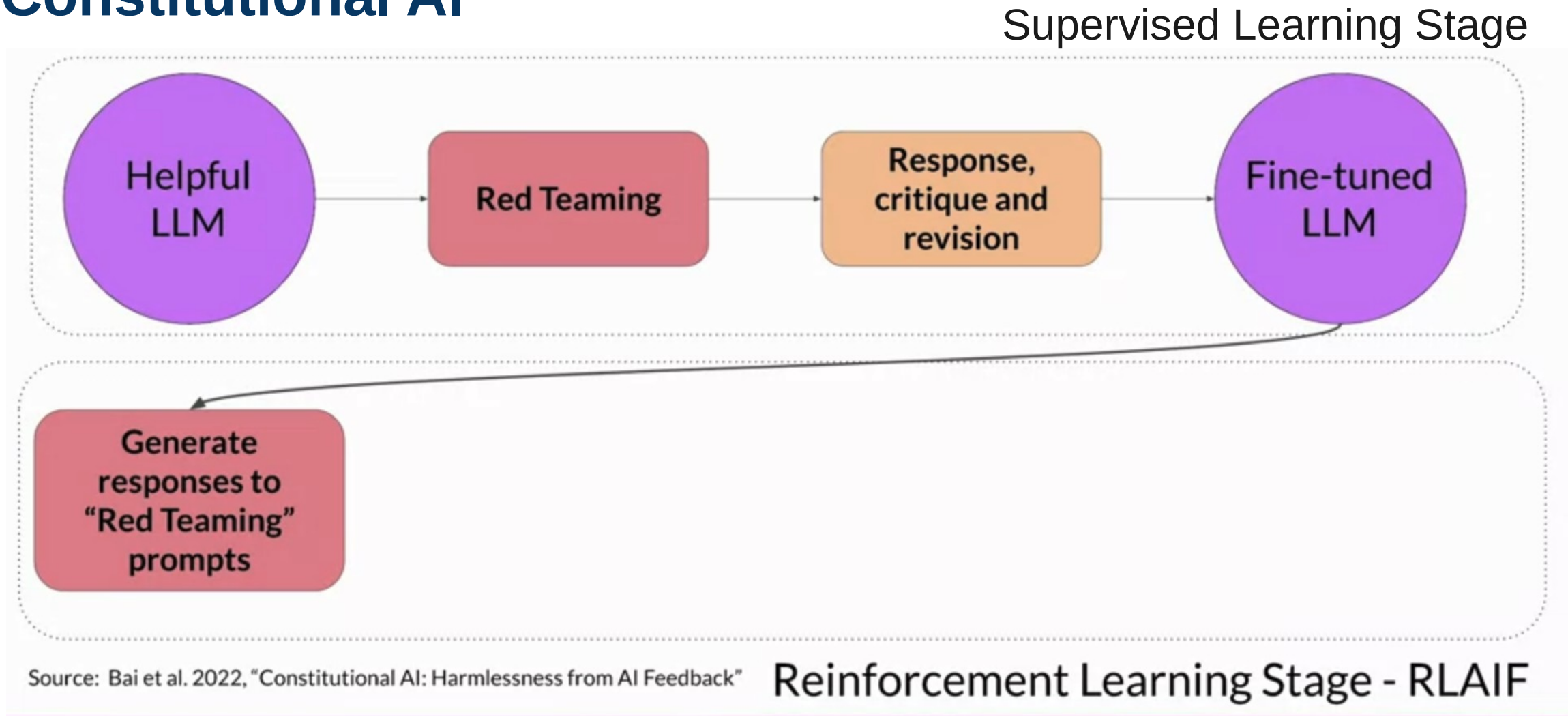
Source: Bai et al. 2022, "Constitutional AI: Harmlessness from AI Feedback"

Constitutional AI

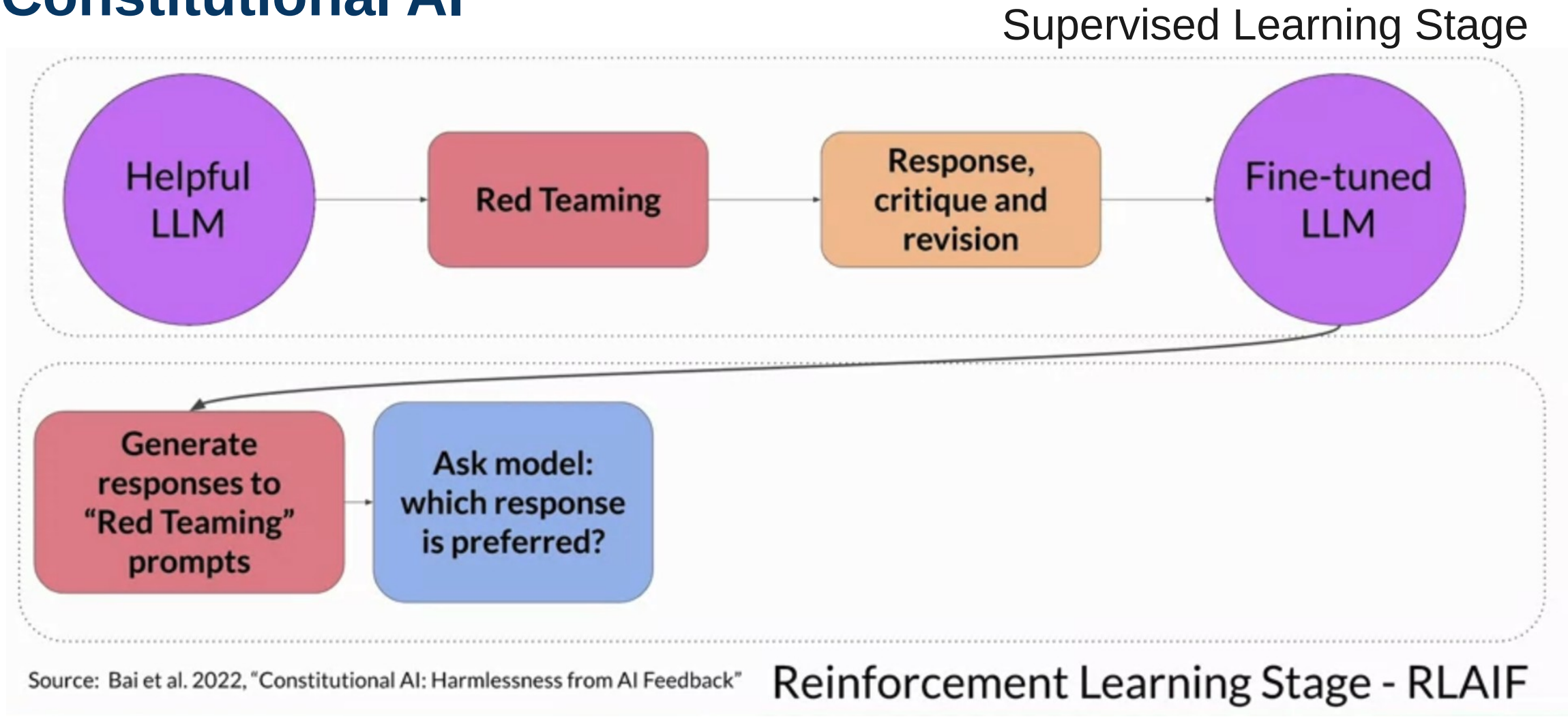
Supervised Learning Stage



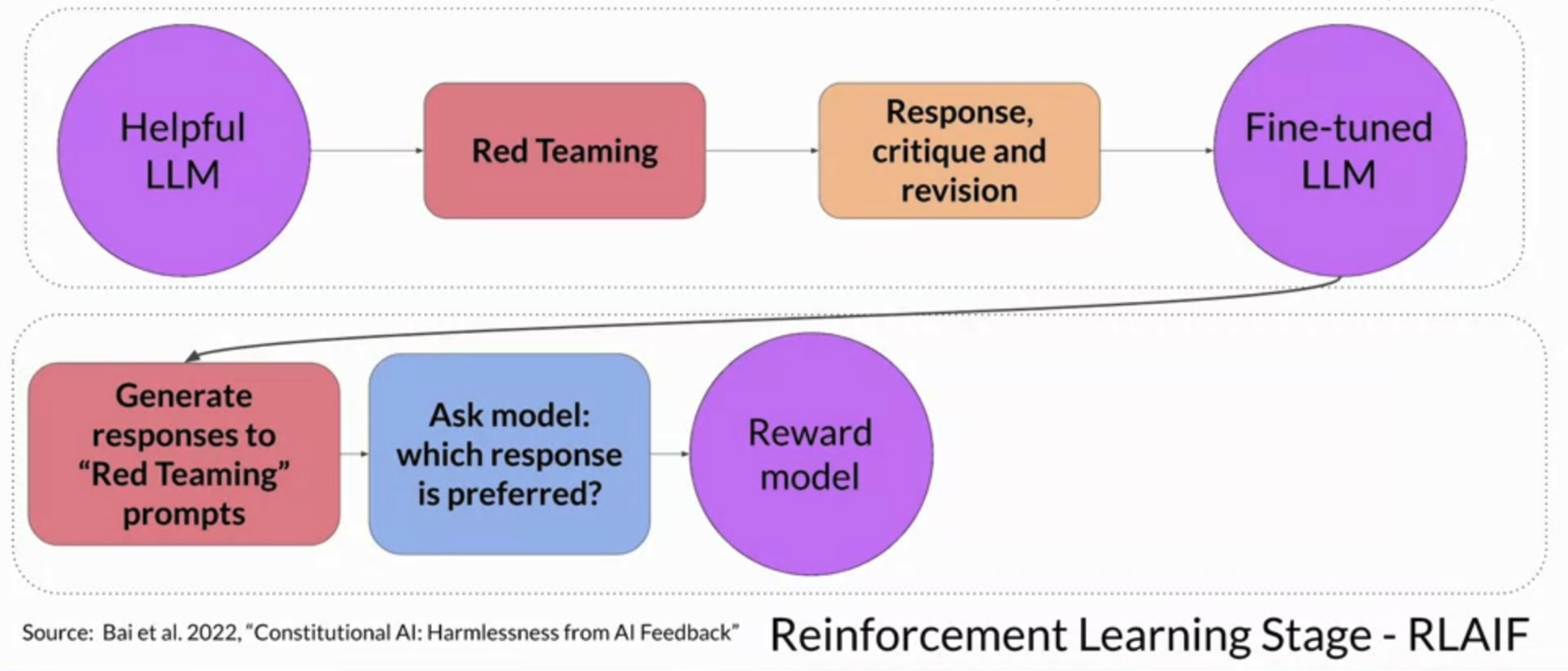
Constitutional AI



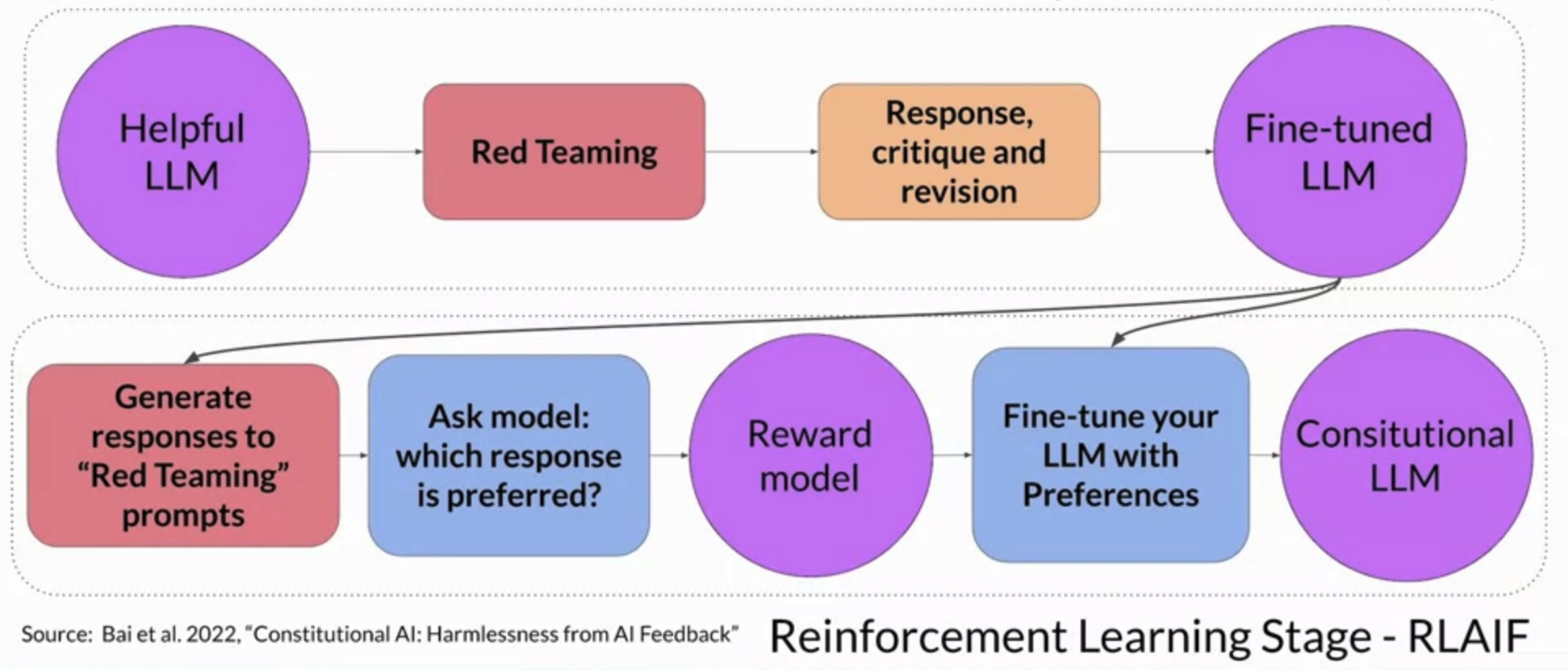
Constitutional AI



Constitutional AI



Constitutional AI



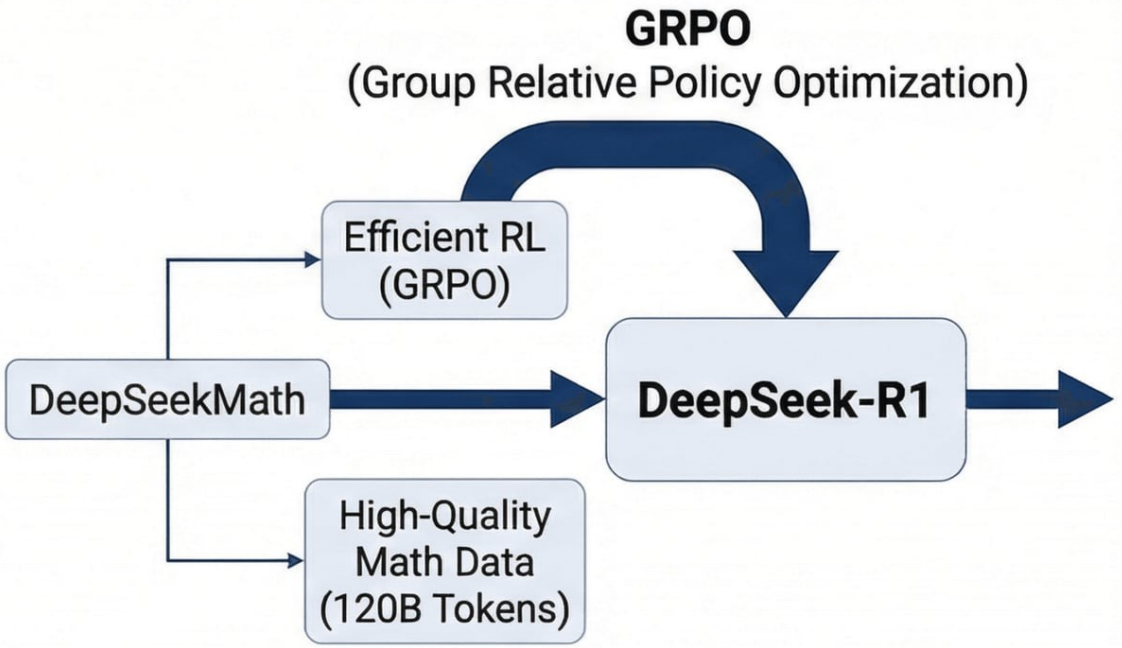
DeepSeekMath: Group Relative Policy Optimization (GRPO) Explained

Pushing the Limits of Mathematical Reasoning in Open Language Models

Zhihong Shao, Peiyi Wang, Qihao Zhu, et al. — DeepSeek-AI, Tsinghua University, Peking University (2024) ·
arXiv:2402.03300

From DeepSeekMath to DeepSeek-R1: GRPO Changed How We Train Reasoning Models

- **DeepSeekMath 7B** achieved **51.7%** on MATH benchmark (competition-level), rivaling **GPT-4**
- Self-consistency over **64 samples** reached **60.9%** on MATH
- **GRPO** was later adopted as the core RL algorithm in **DeepSeek-R1**
- Two pillars: **high-quality math data** (120B tokens) + **efficient RL** (GRPO)
- **First open-source model** to exceed **50%** on MATH



DeepSeekMath Combines Scalable Data Collection with Efficient Reinforcement Learning

- **Stage 1: Pre-training** — 120B math tokens, initialized from DeepSeek-Coder-Base-v1.5 7B
- **Stage 2: SFT** — 776K examples (CoT + PoTT + tool-integrated reasoning)
- **Stage 3: RL with GRPO** — Only ~144K questions, yet significant improvement
- **Key finding:** Code training before math training improves reasoning
- **Key finding:** ArXiv papers surprisingly ineffective for math reasoning

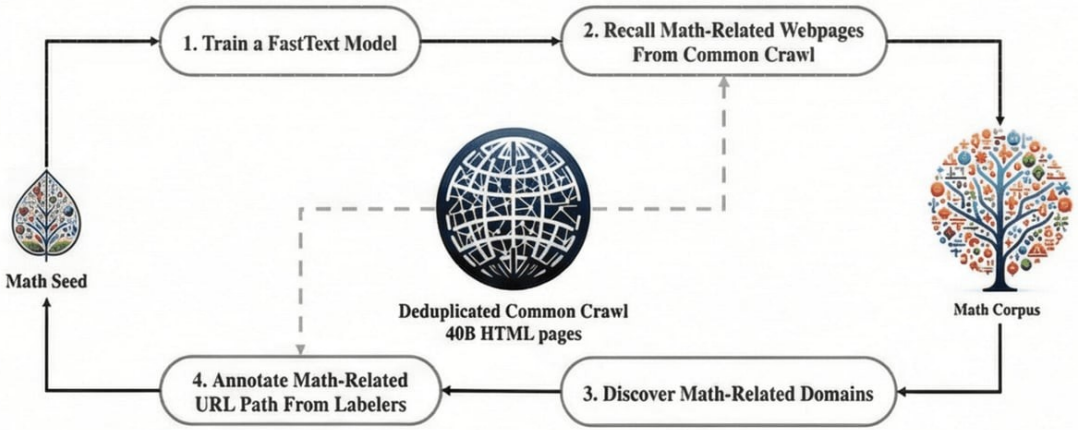


Figure 2 | An iterative pipeline that collects mathematical web pages from Common Crawl.

A 7B Model Achieves 51.7% on Competition-Level MATH, Surpassing All Open-Source Models

Pushing the Limits of Mathematical Reasoning in Open Language Models

- **DeepSeekMath-RL 7B Performance:** GSM8K 88.2%, MATH 51.7% (Chain-of-Thought), MATH 58.8% (Tool-Integrated)
- **Surpasses all open-source models** (7B-70B) and most closed-source models
- **RL improvement over SFT:** GSM8K +5.3%, MATH +4.9%
- **Out-of-domain improvement:** CMATH (Chinese math) 84.6% to 88.8%
- **Self-consistency (64 samples):** 60.9% on MATH

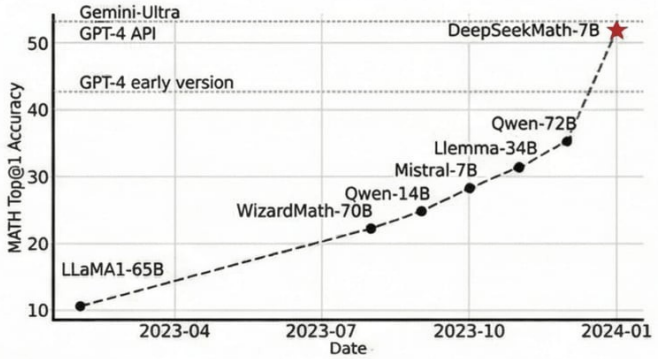


Figure 1 | Top1 accuracy of open-source models on the competition-level MATH benchmark without the use of external toolkits and voting techniques.

Model	Size	English Benchmarks		Chinese Benchmarks	
		GSM8K	MATH	MGSM-zh	CMATH
Chain-of-Thought Reasoning					
Gemini Ultra	-	94.4%	53.7%	-	-
GPT-4	-	92.0%	52.9%	-	86.0%
Inflection-2	-	81.4%	34.8%	-	-
GPT-3.5	-	80.8%	34.1%	-	73.8%
DeepSeekMath-RL	7B	**88.2%**	**51.7%**	**79.6%**	**88.8%**
Tool-Integrated Reasoning					
DeepSeekMath-Instruct	7B	83.7%	54.3%	72.0%	84.3%
DeepSeekMath-RL	7B	**86.7%**	**58.8%**	**78.4%**	**87.6%**

Table 5 | Performance of Open- and Closed-Source models with both Chain-of-Thought and Tool-Integrated Reasoning on English and Chinese Benchmarks.

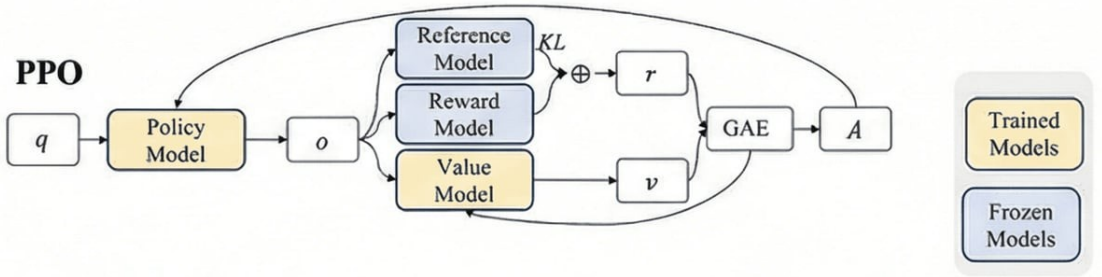
PPO Requires Four Models in Memory, Creating a Significant Resource Bottleneck

- PPO needs 4 models: Policy + Reference + Reward + Value (Critic)
- Value Model is same size as Policy Model → ~2x memory overhead

What is GAE?
Generalized Advantage Estimation (GAE) uses the **Value Model** to estimate how much better an action is compared to the average expected return.

The GAE Challenge: It requires computing temporal difference errors at each step, which is complex for LLMs where rewards are typically only given at the end of the sequence.

- KL penalty is added to the reward at each token, complicating advantage calculation.



The PPO Objective

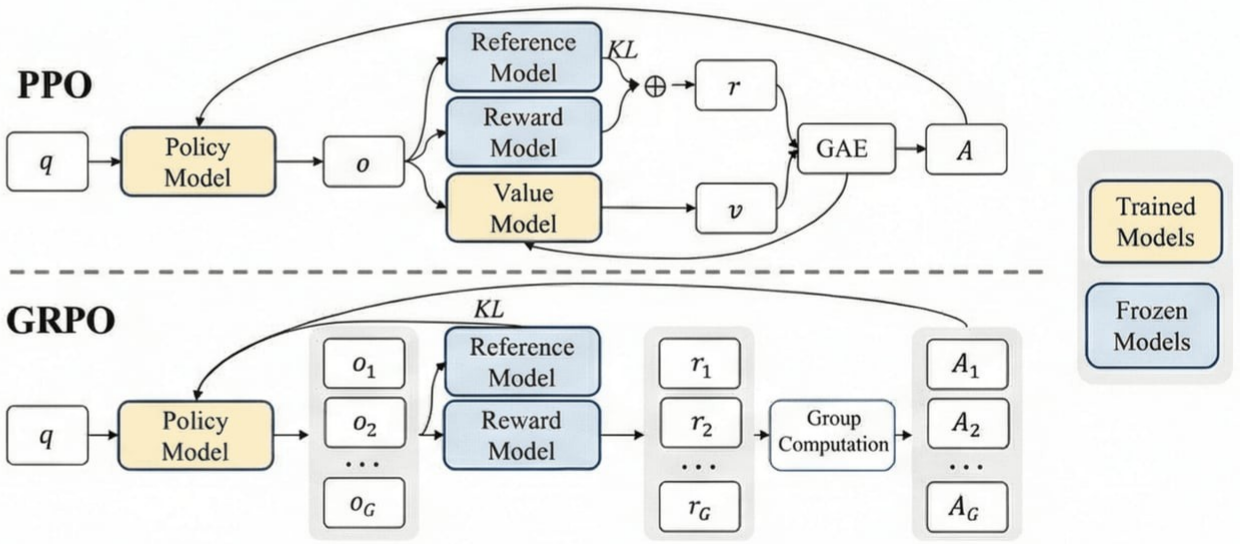
$$J_{\text{PPO}}(\theta) = \mathbb{E}\left[\frac{1}{|o|}\sum \min(\text{ratio} * A_t, \text{clip}(\text{ratio}, 1-\epsilon, 1+\epsilon) * A_t)\right]$$

$$r_t = r_\phi - \beta * \log(\pi_\theta / \pi_{\text{ref}})$$

- PPO Maximizes a Clipped Surrogate Objective with GAE-Based Advantages
- The **Probability Ratio** (π_θ / $\pi_{\theta\,\text{old}}$): This ratio measures how much the new policy's probability of an action has changed compared to the old policy. A ratio > 1 means the action is more likely now.
- Why the Ratio Matters: It allows us to use data collected from the old policy to update the new policy (importance sampling), making training much more sample-efficient.
- Clipping parameter ε prevents destructively large policy updates (typically ε=0.2) by bounding the ratio.
- Advantage A_t computed via GAE with learned value function V_ψ
- min() operation: takes the more pessimistic estimate for stability

GRPO Eliminates the Value Model by Using Group-Relative Advantage Estimation

- Core idea: Sample G outputs per question, compare rewards within the group
- No value/critic model needed → saves ~50% memory vs PPO
- Advantage is computed by normalizing rewards within the group (subtract mean, divide by std)
- KL divergence added directly to the loss (not to the reward)
- Aligns with how reward models are trained (on comparisons between outputs)



Understanding Advantages

Concept & Formula

- Advantage (A_i) measures how much better a response is compared to the group average.

Formula: A_i = (r_i - mean) / std

- Why? Raw rewards lack context. A reward of 0.6 is good if the average is 0.2, but bad if the average is 0.9.

Numerical Example

Question: "What is 2+3?"

Group of 5 responses: 5 (correct), 7 (wrong), 5 (correct), 4 (wrong), 5 (correct)

Rewards: 1, 0, 1, 0, 1
Mean = 0.6, Std = 0.49

Advantages: +0.82 for correct answers (blue color for +0.82), -1.22 for wrong answers (red color for -1.22).

The GRPO Objective

GRPO Optimizes a Group-Averaged Clipped Objective with Direct KL Regularization

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}[q \sim P(\mathbf{Q}), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(\mathbf{O}|q)]$$

$$\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\left\{\min\left[\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q,o_{i,<t})}\hat{A}_{i,t}, \text{clip}\left(\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q,o_{i,<t})}, 1-\epsilon, 1+\epsilon\right)\hat{A}_{i,t}\right] - \beta D_{\text{KL}}[\pi_\theta||\pi_{\text{ref}}]\right\}$$

- Outer sum averages over G outputs in the group
- Inner sum averages over tokens in each output
- Same clipped ratio mechanism as PPO for stability
- KL divergence added directly to loss (not reward) — cleaner advantage computation
- Unbiased KL estimator guaranteed to be non-negative

The Core of RL: Probability × Advantage

- The objective function maximizes the product of the model's **probability** π(o|q) and the **advantage** A_i.

1. Positive Advantage, High Probability



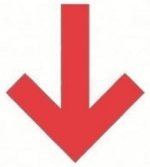
Product is large positive.
Model is already doing well.

2. Positive Advantage, Low Probability



Product is small positive.
Gradient pushes model to INCREASE probability.

3. Negative Advantage, High Probability



Product is large negative.
Gradient pushes model to DECREASE probability.

4. Negative Advantage, Low Probability



Product is small negative.
Model is already assigning low probability to bad responses.

Probability Ratio & Clipping

Probability Ratio (π_θ / $\pi_{\theta old}$)

- Corrects for distribution mismatch (importance sampling) since we sample from the old policy but optimize the new one.

Clipping [1-ε, 1+ε]

- Prevents the model from changing too much in one step (e.g., ε = 0.1 means ratio is capped between 0.9 and 1.1).

1. A_ii > 0, ratio < 1+ε



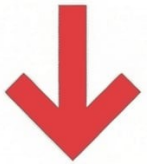
Model continues to INCREASE probability. Ratio is within the upper bound.

2. A_i > 0, ratio > 1+ε



Model STOPS increasing probability (clipped). Ratio exceeds the upper bound.

3. A_i < 0, ratio > 1-ε



Model continues to DECREASE probability. Ratio is within the lower bound.

4. A_i < 0, ratio < 1-ε



Model STOPS decreasing probability (clipped). Ratio exceeds the lower bound.

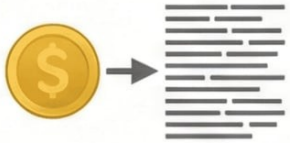
Process Supervision Provides Fine-Grained Step-Level Rewards for Better Training Signal

Outcome Supervision (OS)

- Single reward per output
- all tokens share same advantage

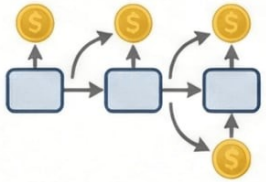
Advantage formula (OS):

$$\hat{A}_{i,t} = \frac{r_i - \text{mean}(r)}{\text{std}(r)}$$



Process Supervision (PS)

- Reward per reasoning step
- advantages accumulate from future steps

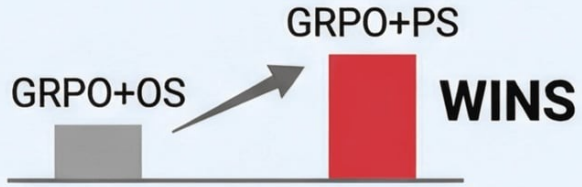


Advantage formula (PS):

$$\hat{A}_{i,t} = \sum_{\text{index}(j) \geq t} \tilde{r}_i^{\text{index}(j)}$$

Experimental Result

GRPO+PS > GRPO+OS (process supervision wins)



Iterative GRPO Continuously Updates Both the Policy and Reward Models

- Hyperparameters: ε (clipping), β (KL coefficient = 0.04), μ (GRPO iterations per step)
- Group size G = 64 outputs per question
- Single policy update per exploration stage (μ=1)
- Reward model updated with replay mechanism (10% historical data)
- Reference model reset at each iteration to current policy

Algorithm 1 Iterative Group Relative Policy Optimization

Input initial policy model $\pi_{\theta_{\text{init}}}$; reward models r_φ; task prompts $\mathcal{D}$; ε, β, μ

policy model $\pi_\theta \leftarrow \pi_{\theta_{\text{init}}}$
for iteration = $1, \ldots, I$ **do**
  reference model $\pi_{\text{ref}} \leftarrow \pi_\theta$
  for step = $1, \ldots, M$ **do**
    Sample a batch $\mathcal{D}_b$ from $\mathcal{D}$
    Update the old policy model $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$
    Sample G outputs $\{o_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(\cdot \mid q)$ for each question $q \in \mathcal{D}_b$
    Compute rewards $\{r_i\}_{i=1}^{G}$ for each sampled output o_i by running r_φ
    Compute $\hat{A}_{i,t}$ for the t-th token of o_i through group relative advantage estimation.
    for GRPO iteration = $1, \ldots, \mu$ **do**
      Update the policy model π_θ by maximizing the GRPO objective (Equation 21)

A Unified Paradigm: Understanding SFT, RFT, and GRPO

$$\nabla_\theta J_A(\theta) = \mathbb{E}\left[\frac{1}{|o|}\sum GC_A \cdot \nabla_\theta \log \pi_\theta\right]$$

Methods	Data Source	Reward Function	Gradient Coefficient
SFT	$q, o \sim P_{sft}(Q, O)$	-	1
RFT	$q \sim P_{sft}(Q), o \sim \pi_{sft}(O\|q)$	Rule	Equation [10]
DPO	$q \sim P_{sft}(Q), o^+, o^- \sim \pi_{sft}(O\|q)$	Rule	Equation [14]

- **Unified gradient:** all methods = weighted log-probability gradient
- **SFT (Supervised Fine-Tuning):** Offline data, rule-based reward, GC=1. It equally reinforces every token in human-selected data.
- **RFT (Rejection Sampling Fine-Tuning):** Offline data, rule-based reward, Binary GC (1 for correct, 0 for incorrect). It cannot distinguish between a barely correct answer and a brilliant one.
- **GRPO:** Online data, model-based reward, Continuous GC. It differentially reinforces outputs based on their quality and penalizes bad outputs with negative coefficients.
- **Key insight:** Online > Offline, Model > Rule, Continuous > Binary

GRPO with Process Supervision Outperforms All Other Methods on Both GSM8K and MATH

- Clear hierarchy: GRPO+PS > GRPO+OS > Online RFT > RFT
- Online training advantage becomes more pronounced over time
- GRPO's differential gradient coefficients outperform RFT's binary approach
- Process supervision provides better signal than outcome supervision
- All methods improve over SFT baseline, but RL methods improve more

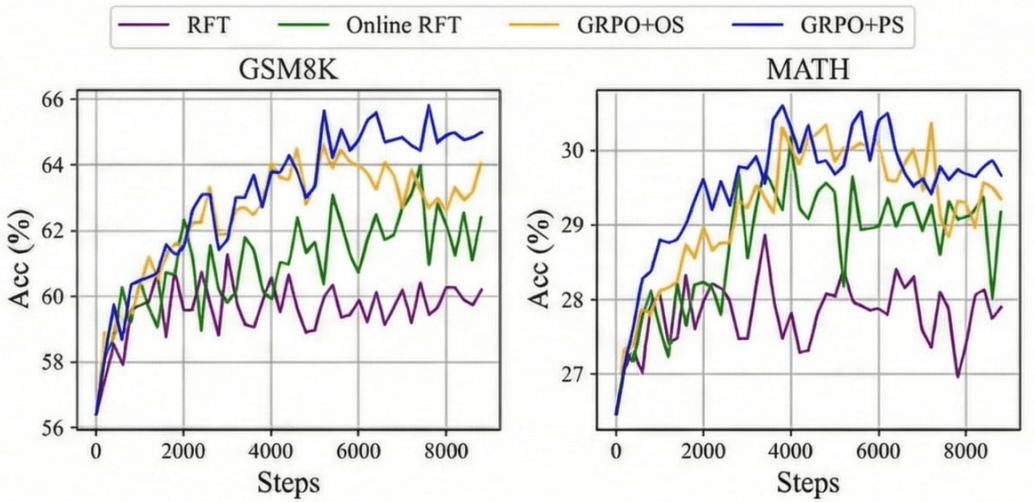


Figure 5 | The data source and gradient coefficient of different methods. P_{sft} denotes the data distribution of supervised fine-tuning datasets. $\pi_{0_{sft}}$ and π_0 denote the supervised fine-tuned model and the real-time policy model during the online training process, respectively.

Two Rounds of Iterative GRPO Significantly Boost Performance on Both Benchmarks

- **Iteration-0 → Iteration-1:** Significant improvement on both benchmarks
- **Iteration-1 → Iteration-2:** Diminishing returns but still positive
- Reward model must evolve with the policy for continued improvement
- **GSM8K:** ~83% → ~87% → ~89% across iterations
- **MATH:** ~47% → ~51% → ~52% across iterations

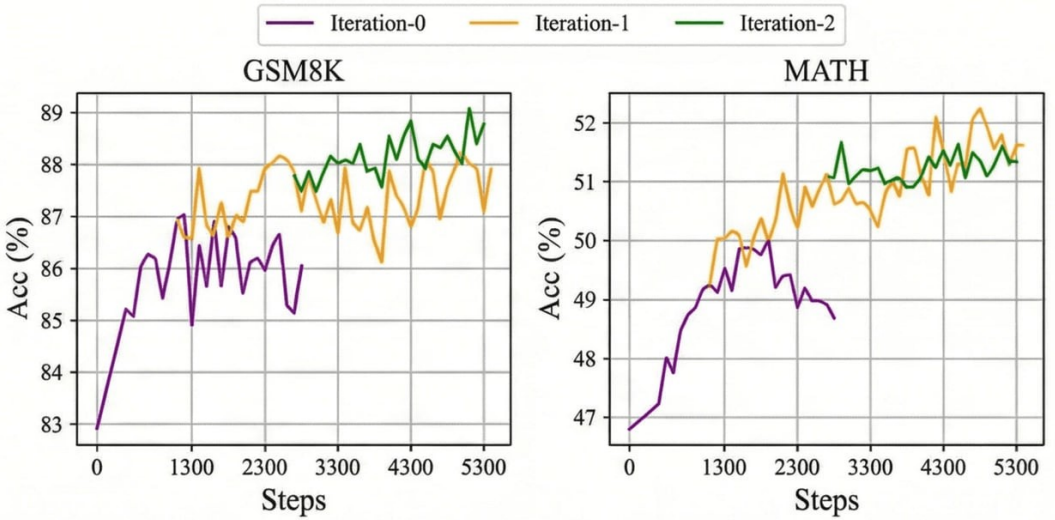
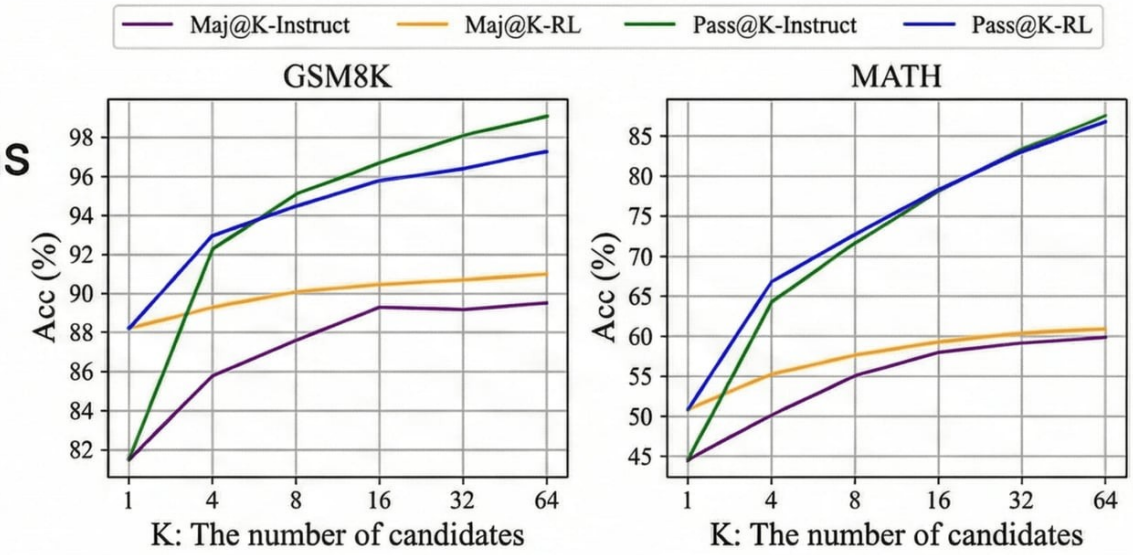


Figure 6 | Performance of iterative reinforcement learning with DeepSeekMath Instruct 7B on performance tonen and iteratimas target/smotive improvement iterations.

RL Makes the Output Distribution More Robust Rather Than Expanding Fundamental Capabilities

- RL improves Maj@K significantly (e.g., Maj@64 on MATH: ~55% → ~61%)
- RL does NOT improve Pass@K (the ceiling of what the model can solve)
- Interpretation: RL redistributes probability mass toward correct solutions
- RL is an alignment/robustness technique, not a capability expansion technique
- Implication: The SFT model already "knows" the answers; RL helps it consistently produce them

Figure 7: Maj@K and Pass@K Curves



GRPO Achieves Superior Results with Carefully Tuned Hyperparameters and Efficient Memory Usage

HYPERPARAMETERS: Learning Rates

Policy Learning Rate: 1e-6

Reward Model Learning Rate: 2e-5

HYPERPARAMETERS: Configuration

KL Coefficient $\beta = 0.04$

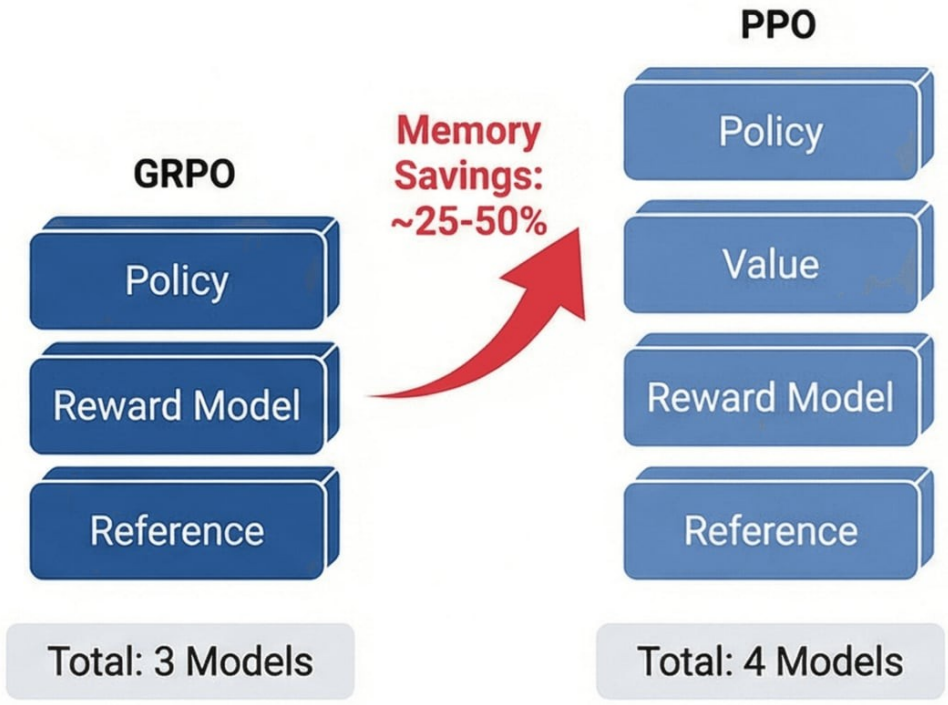
Group Size $G = 64$

TRAINING EFFICIENCY: Data & Updates

Only ~144K Training Questions (Subset of SFT Data)

Single Policy Update per Exploration Stage

MEMORY USAGE



GRPO Laid the Foundation for Emergent Reasoning in DeepSeek-R1 and Beyond

Direction 1: Better data sources

- Out-of-distribution prompts
- Tree-search sampling

Direction 2: Robust algorithms

- Handling noisy rewards
- Weak-to-strong learning

Direction 3: Better reward models

- Generalization
- Uncertainty
- Process supervision

Concluding Section & Future Landscape

- DeepSeek-R1 (Jan 2025): GRPO enabled emergent reasoning without SFT
- 2026 landscape: GRPO variants (DAPO, Dr. GRPO) are now standard for reasoning models