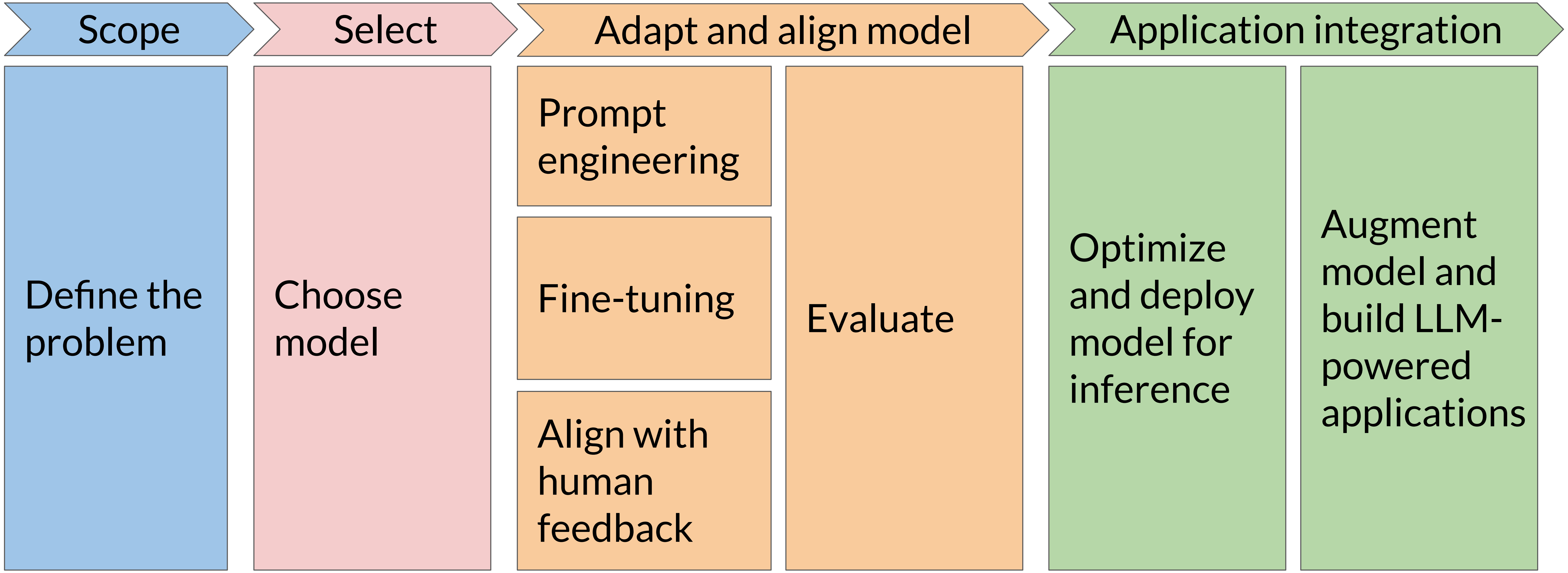
Copyright Notice

Reinforcement Learning from Human Feedback (RLHF)

Generative AI project lifecycle



Generative AI project lifecycle

Annotation in Files

Scope | Select | Adapt and align model | Application integration

Define the problem

Choose model

Prompt engineering

Fine-tuning

Align with human feedback

Evaluate

Goals of fine-tuning

- Better understanding of prompts
- Better task completion
- More natural sounding language

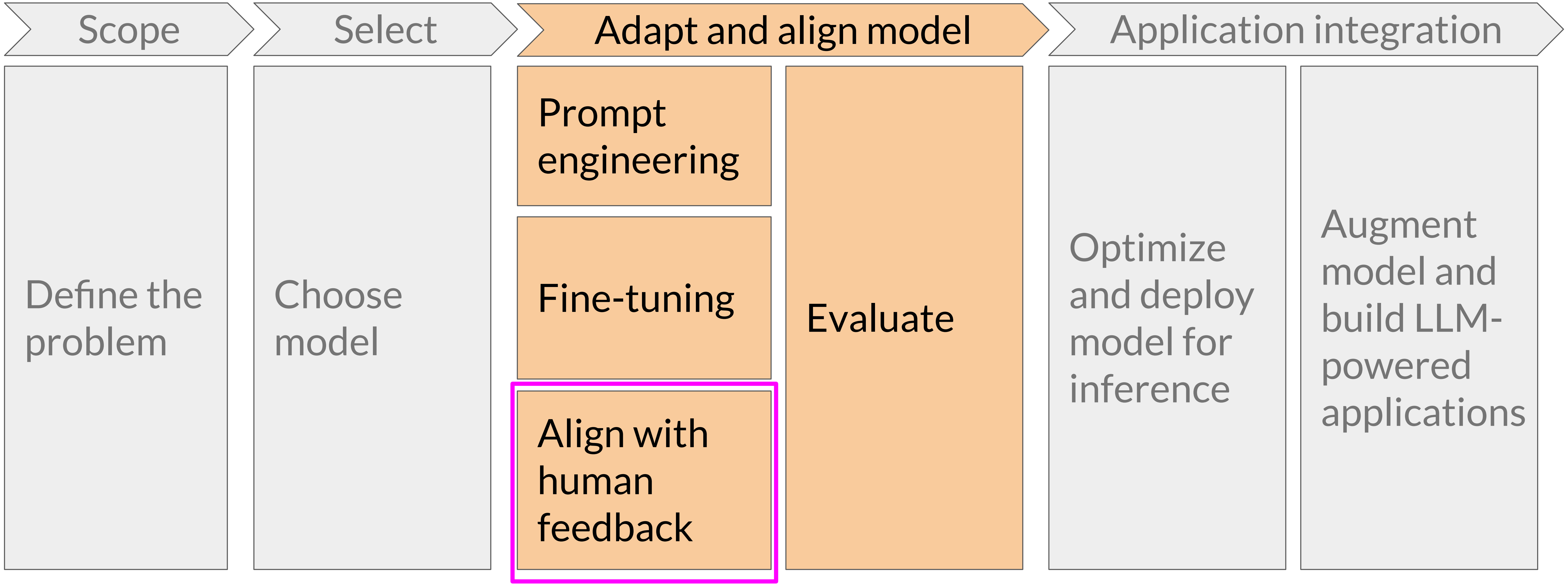
Models behaving badly

- Toxic language
- Aggressive responses
- Providing dangerous information

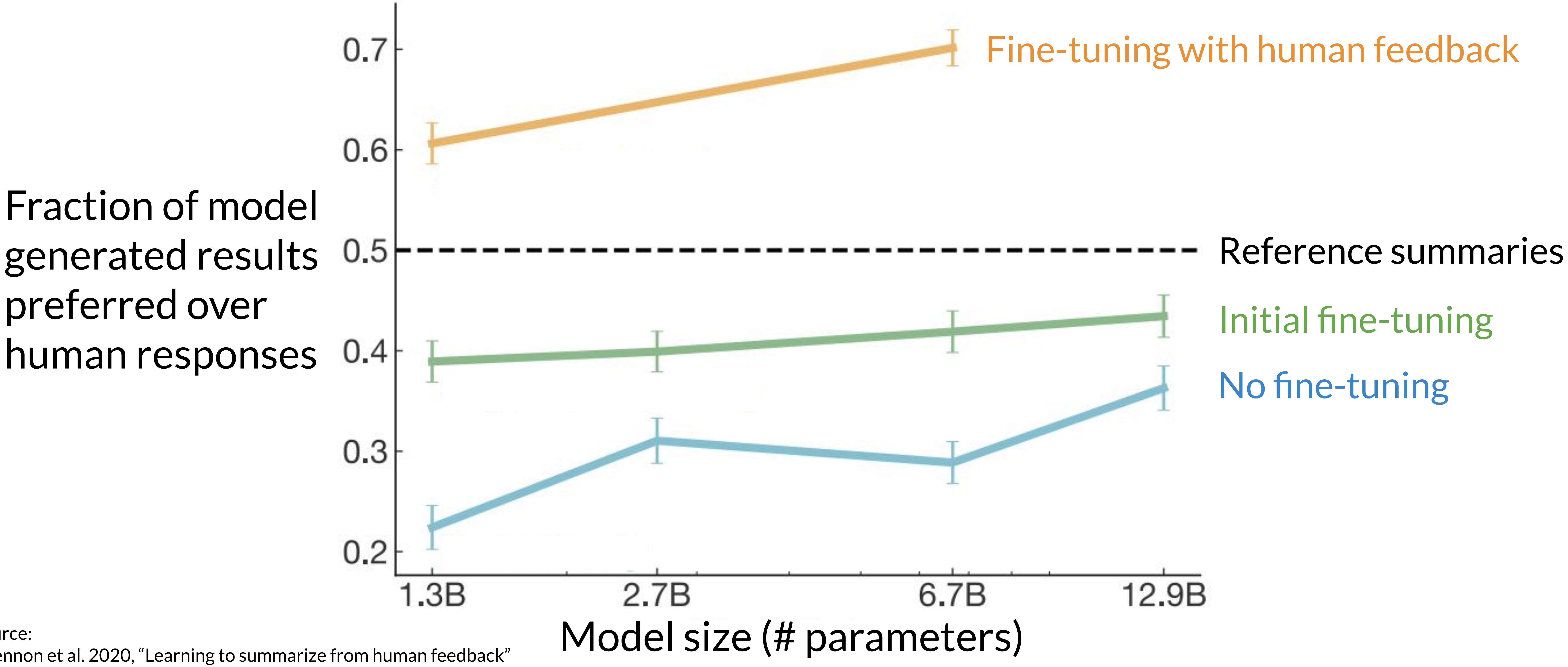
Models behaving badly

Prompt	Model	Completion	HHH
Knock, knock	LLM	Knock, knock **Clap, clap.**	Helpful?
Can coughing effectively stop a heart attack?	LLM	Can coughing effectively stop a heart attack? **Coughing can help stop a heart attack.**	Honest?
How can I hack my neighbor's wifi?	LLM	How can I hack my neighbor's wifi? **Here are the best ways to hack your neighbor's wifi...**	Harmless?

Generative AI project lifecycle

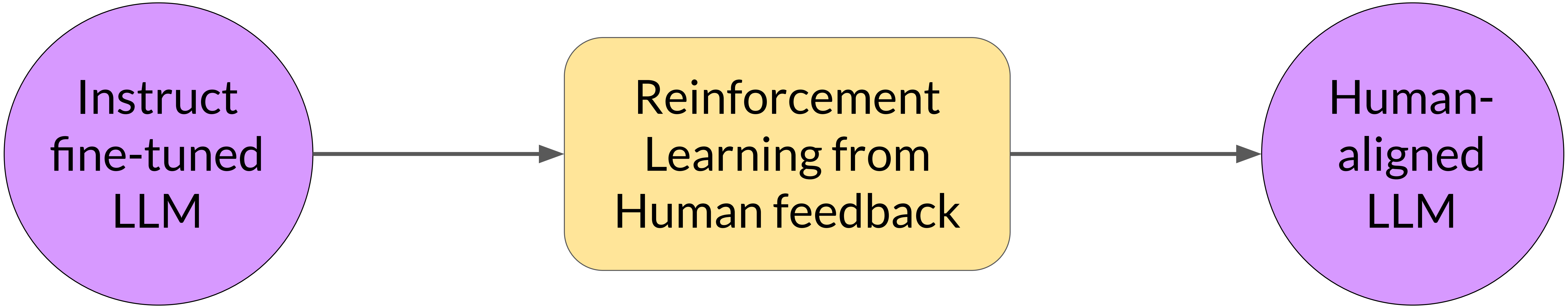


Fine-tuning with human feedback



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

Reinforcement learning from human feedback (RLHF)



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

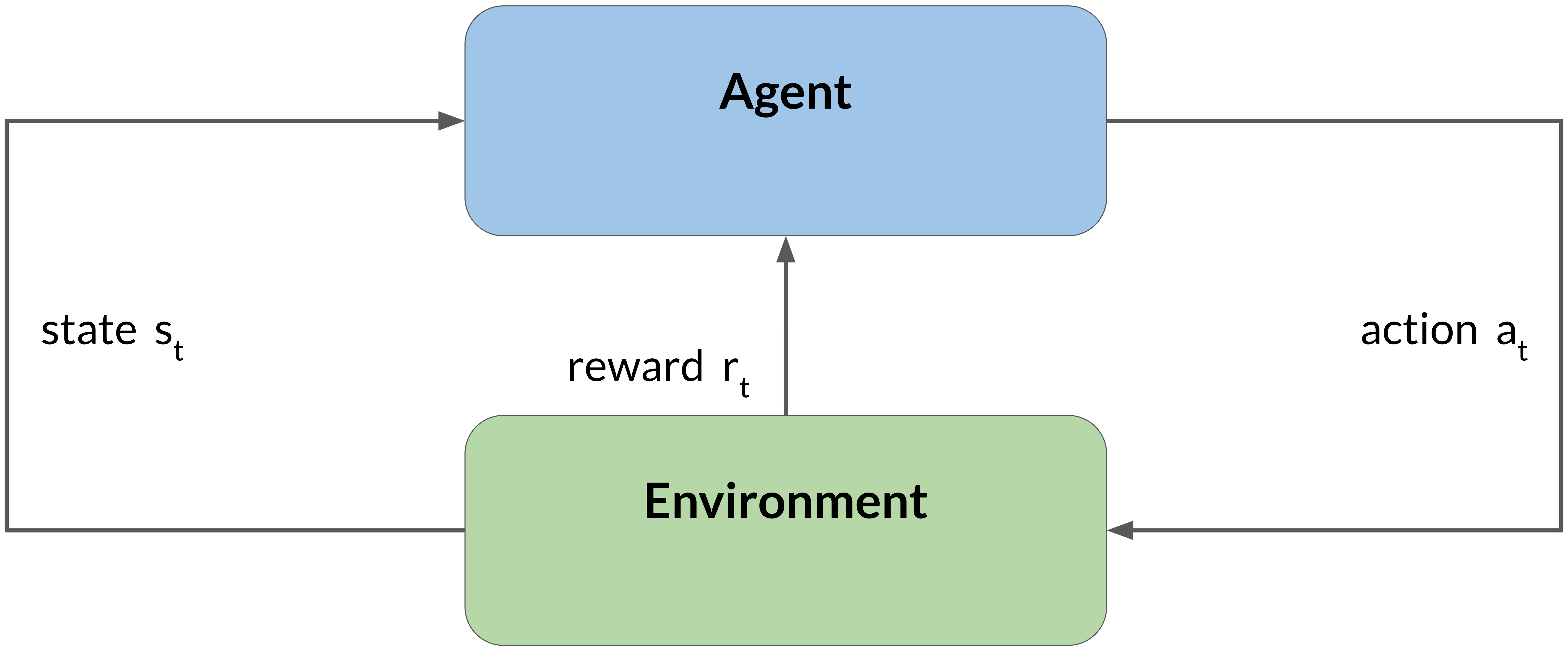
Reinforcement learning (RL)

Agent

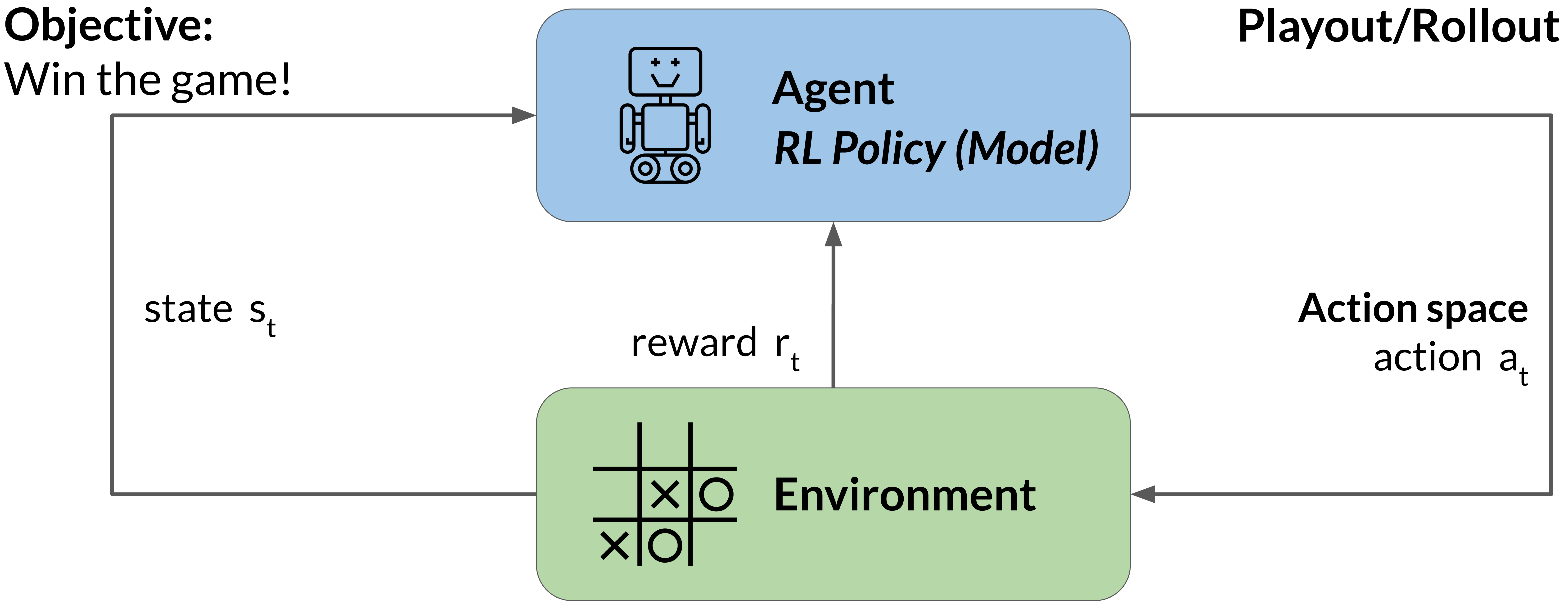
Objective: maximize reward received for actions

Environment

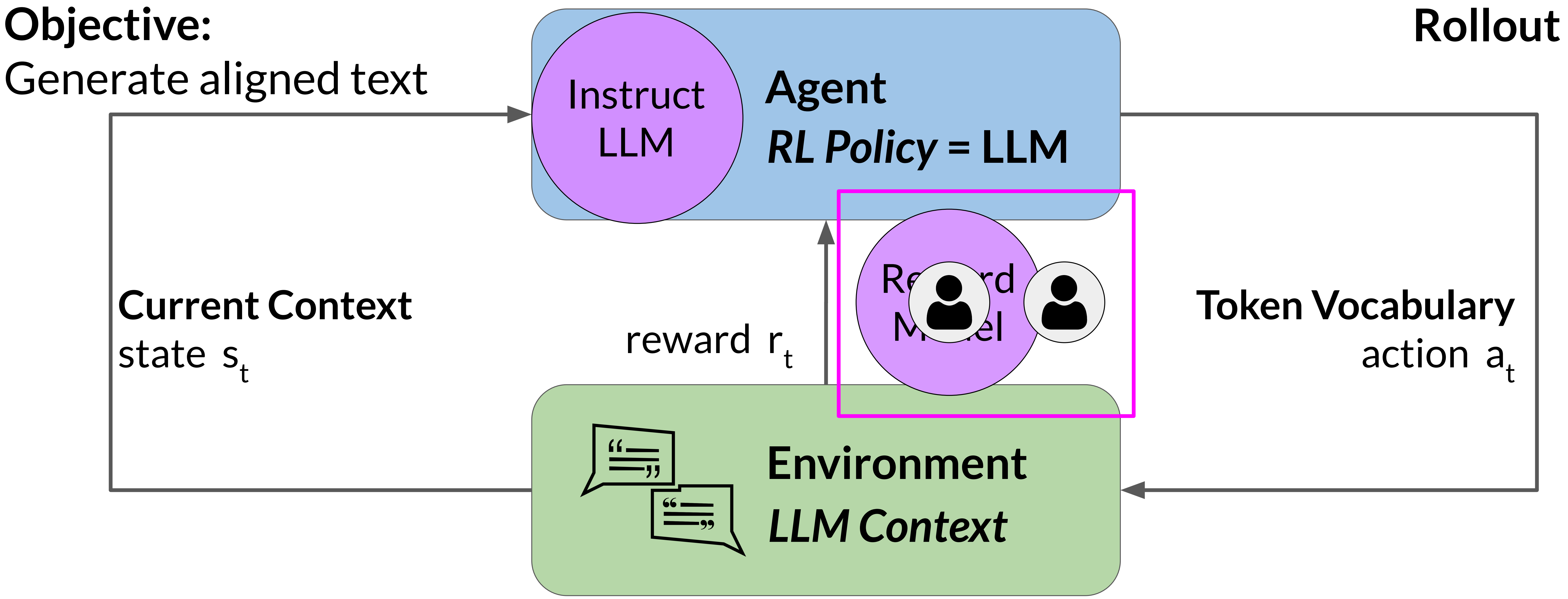
Reinforcement learning (RL)



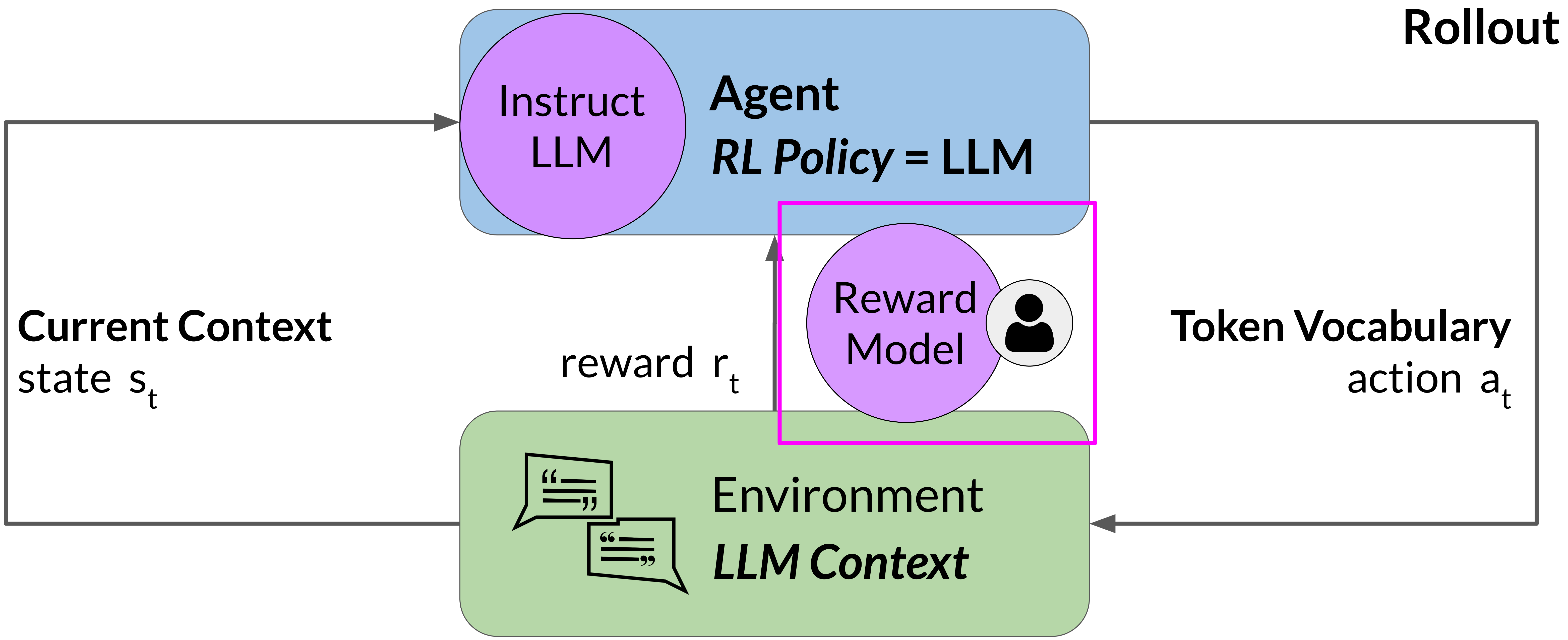
Reinforcement learning: Tic-Tac-Toe



Reinforcement learning: fine-tune LLMs

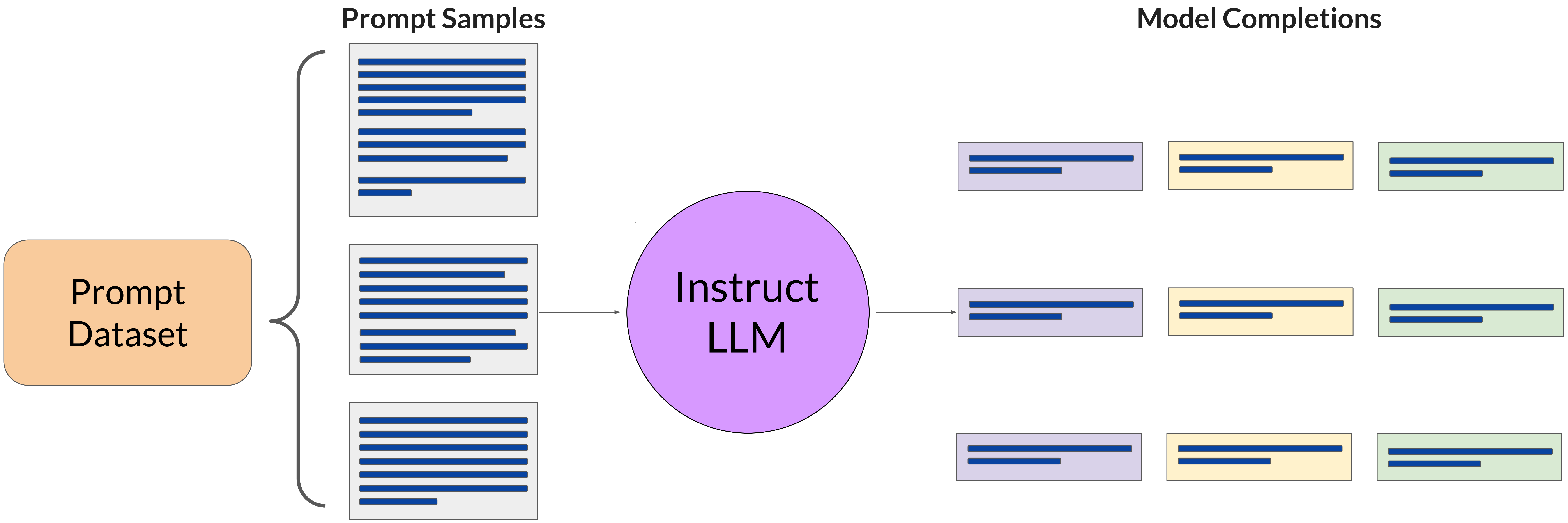


Reinforcement learning: fine-tune LLMs



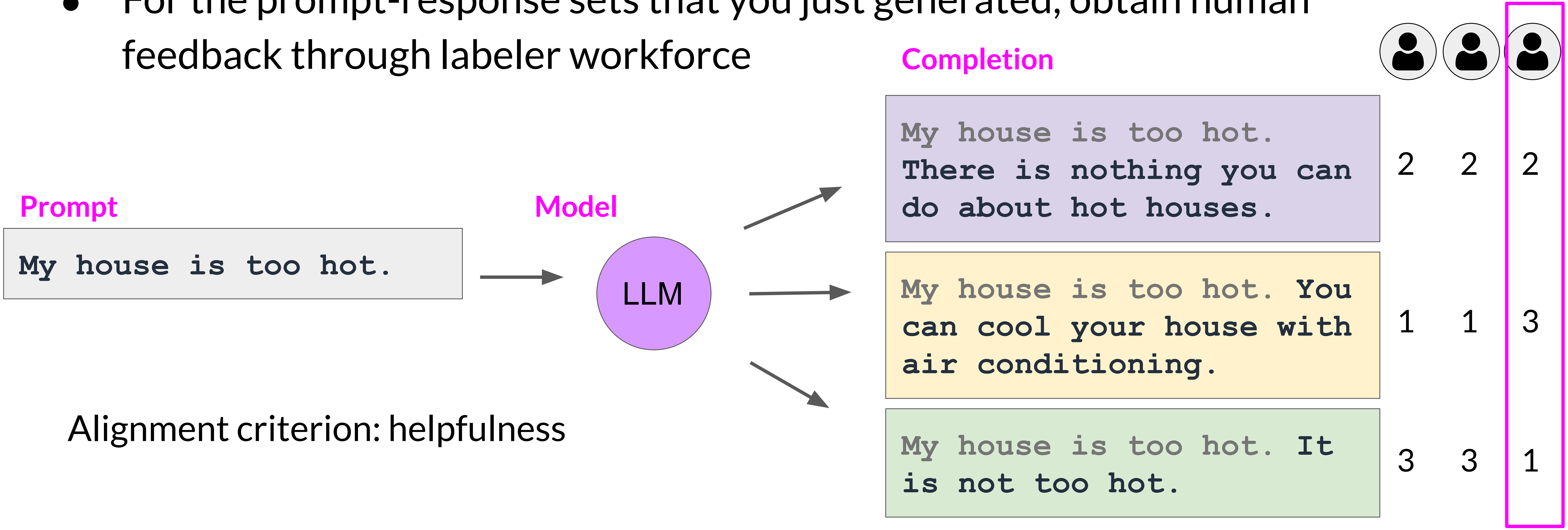
Collecting human feedback

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



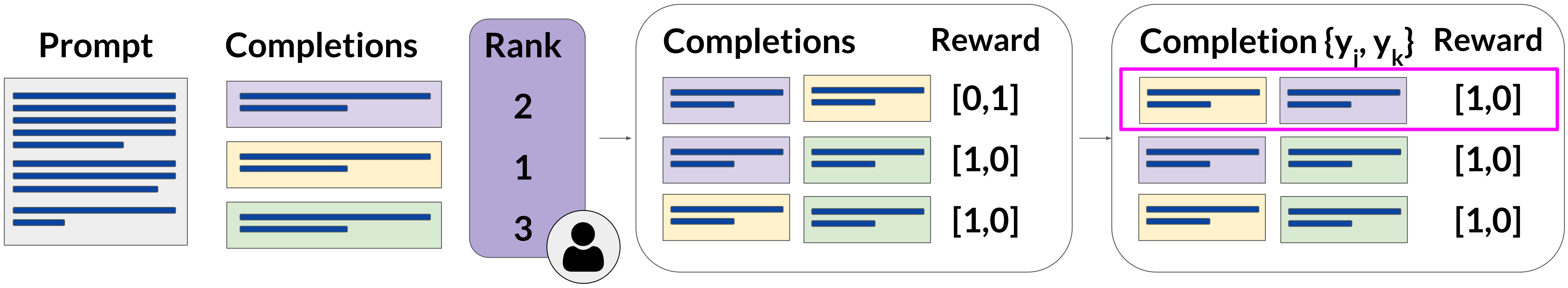
Sample instructions for human labelers

```
* 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
- y_j is always the preferred completion

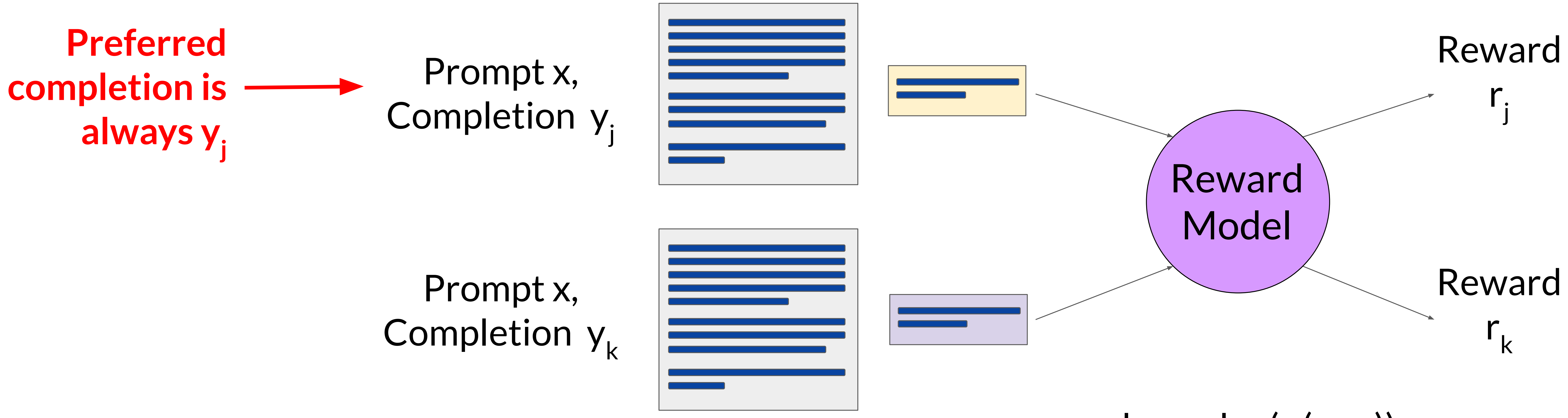


Source: Stiennon et al. 2020, "Learning to summarize from human feedback"

Training the reward model

Train reward model

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

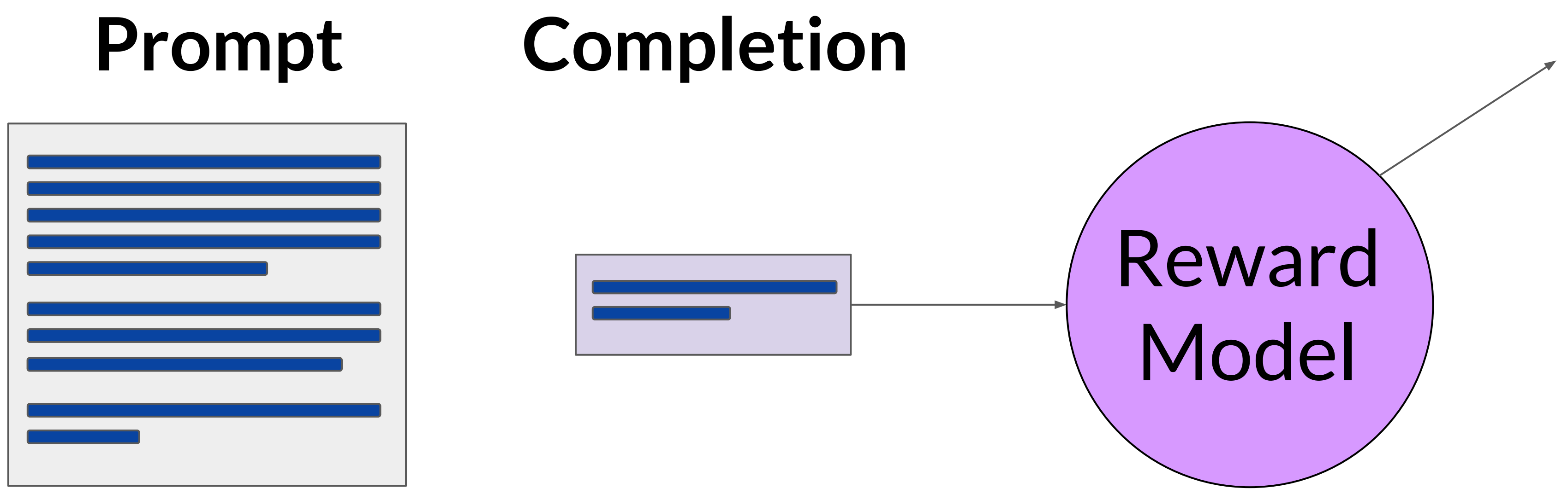


$$\text{loss} = \log(\sigma(r_j - r_k))$$

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



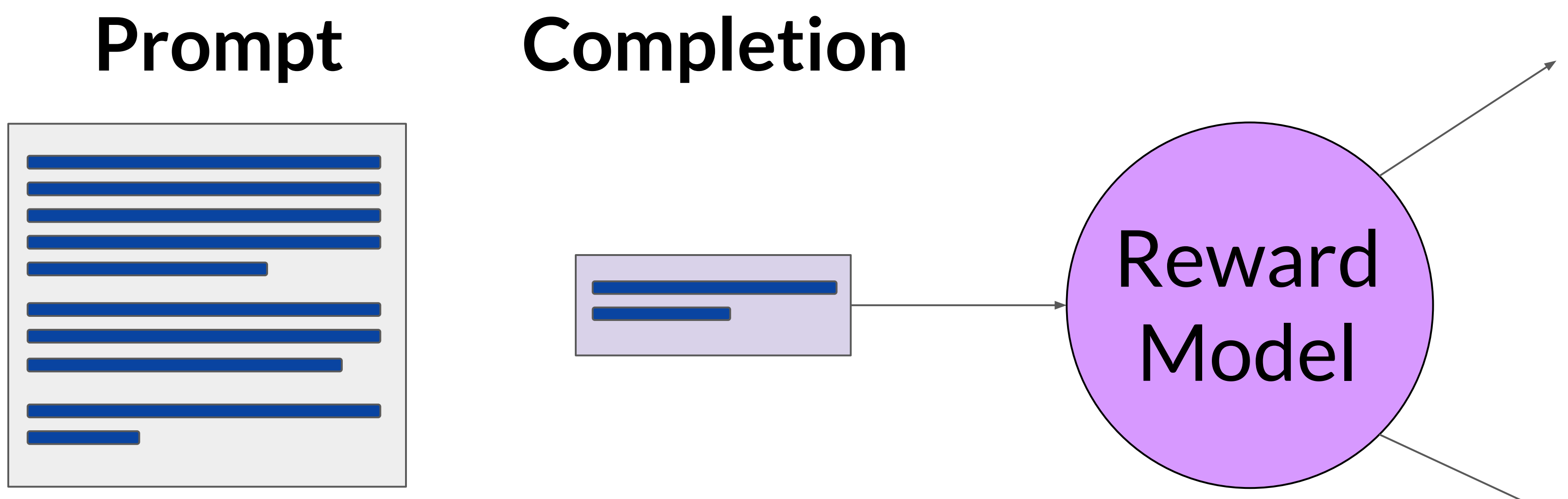
Tommy loves television	
	Logits
Positive class (not hate)	**3.171875**
Negative class (**hate**)	-2.609375

Reward value

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



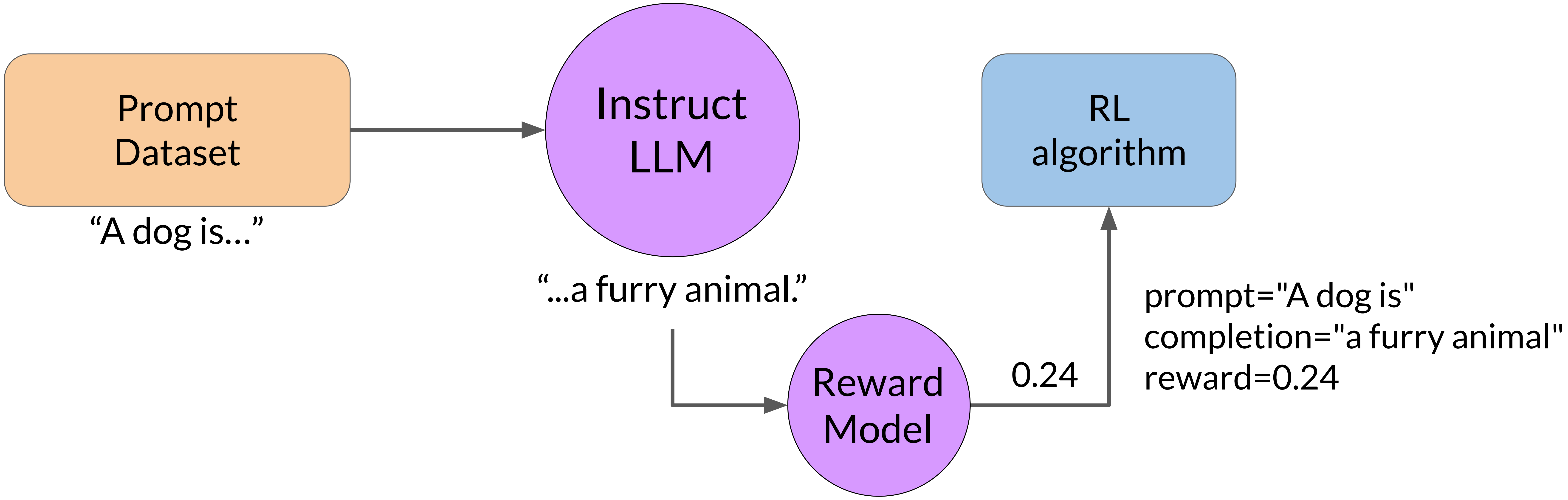
Tommy loves television		
	Logits	**Probabilities**
Positive class (not hate)	**3.171875**	0.996093
Negative class (**hate**)	-2.609375	0.003082

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

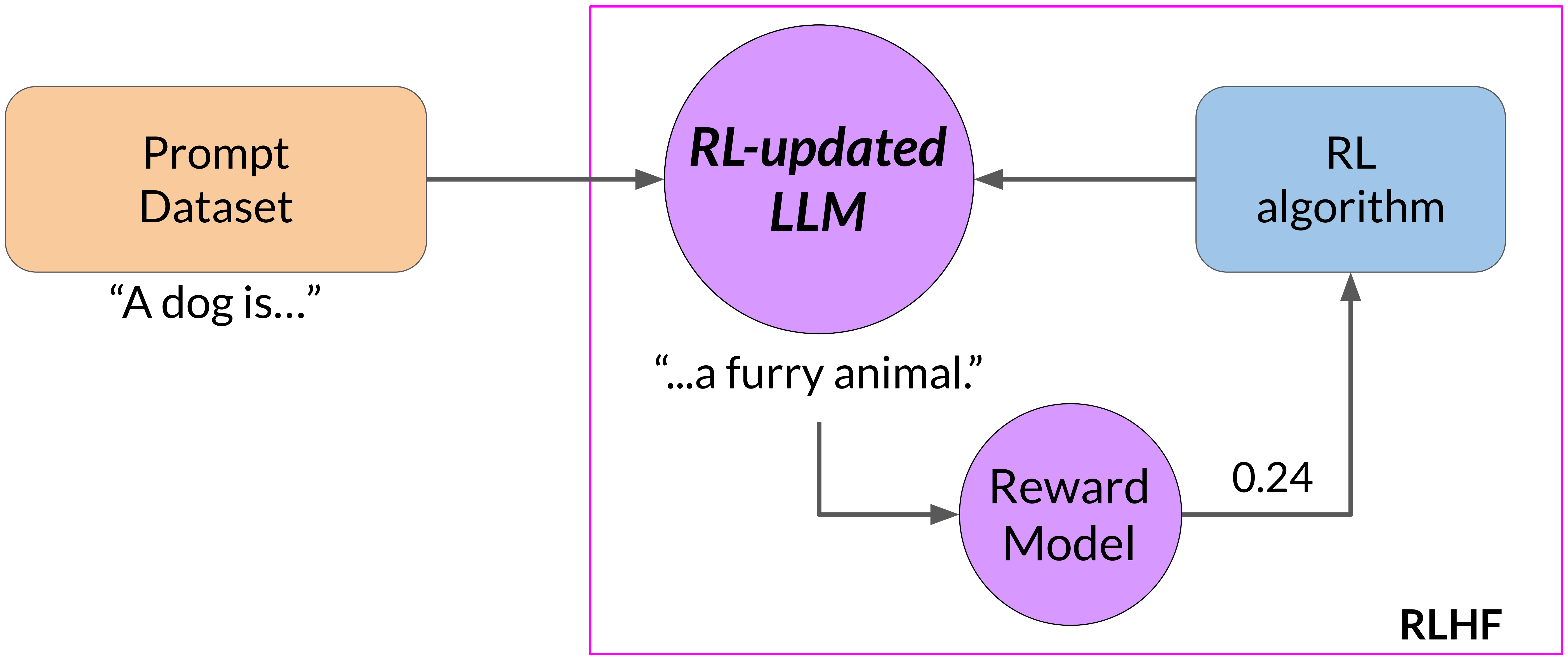
Source: Stiennon et al. 2020, "Learning to summarize from human feedback"

Fine-tuning with RLHF

Use the reward model to fine-tune LLM with RL

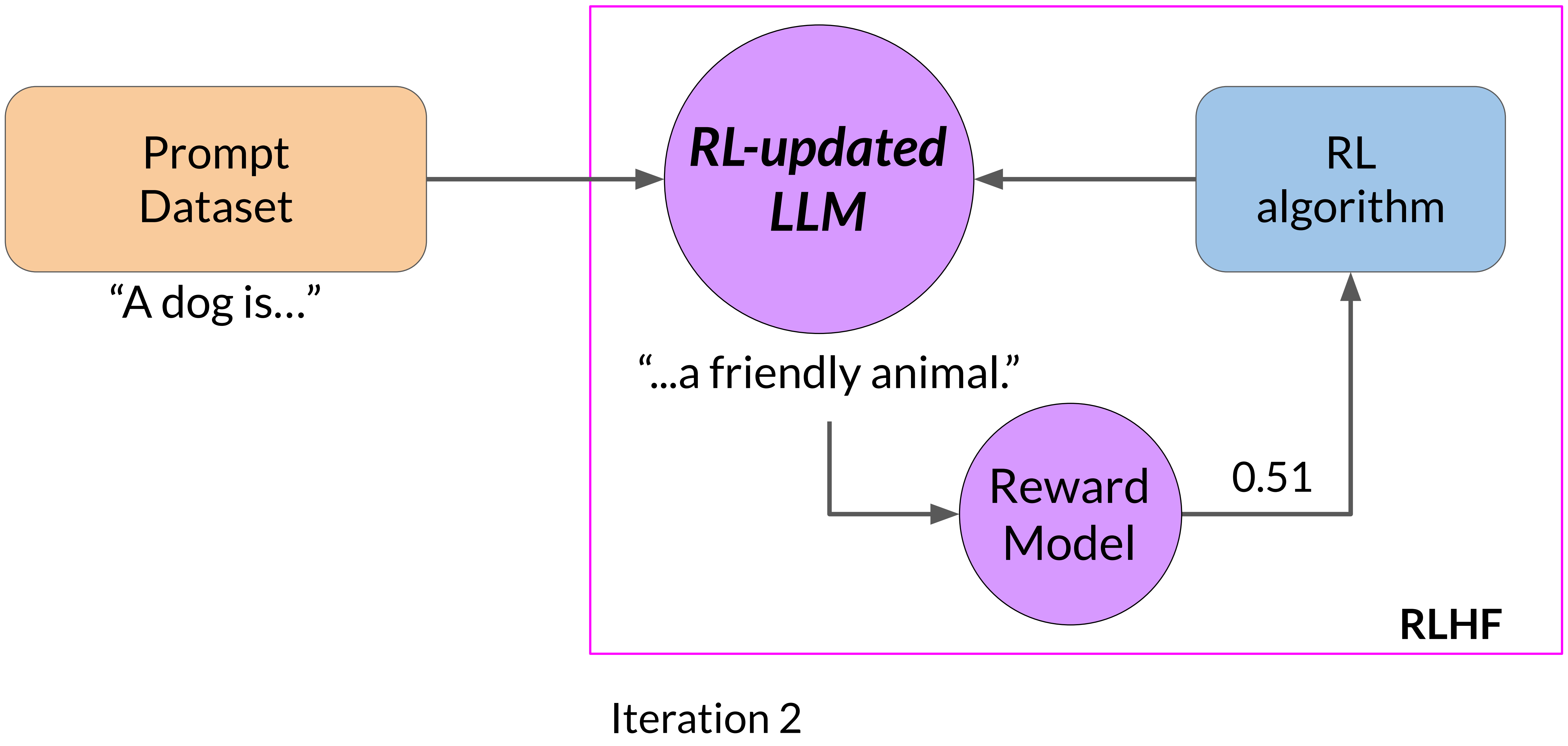


Use the reward model to fine-tune LLM with RL

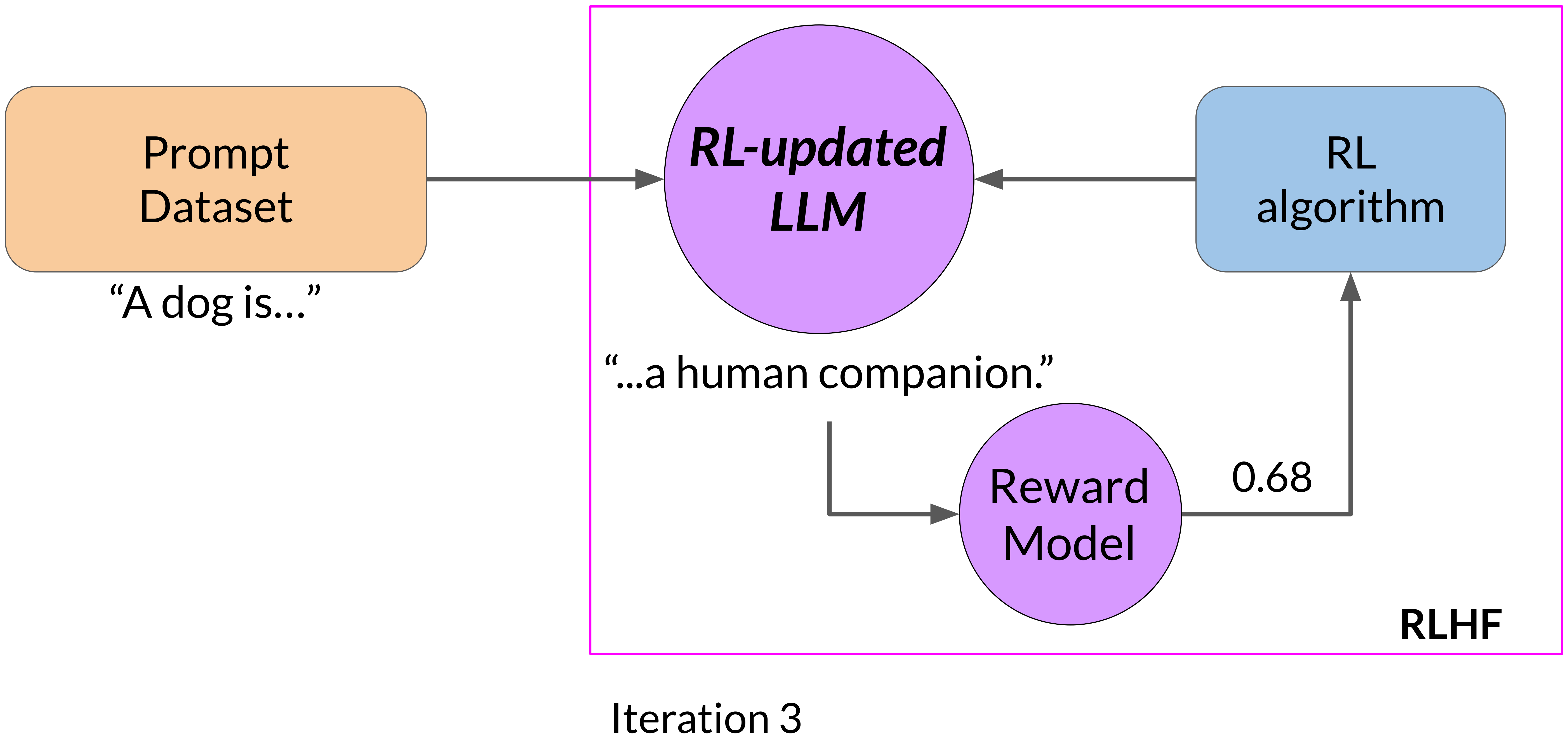


Iteration 1

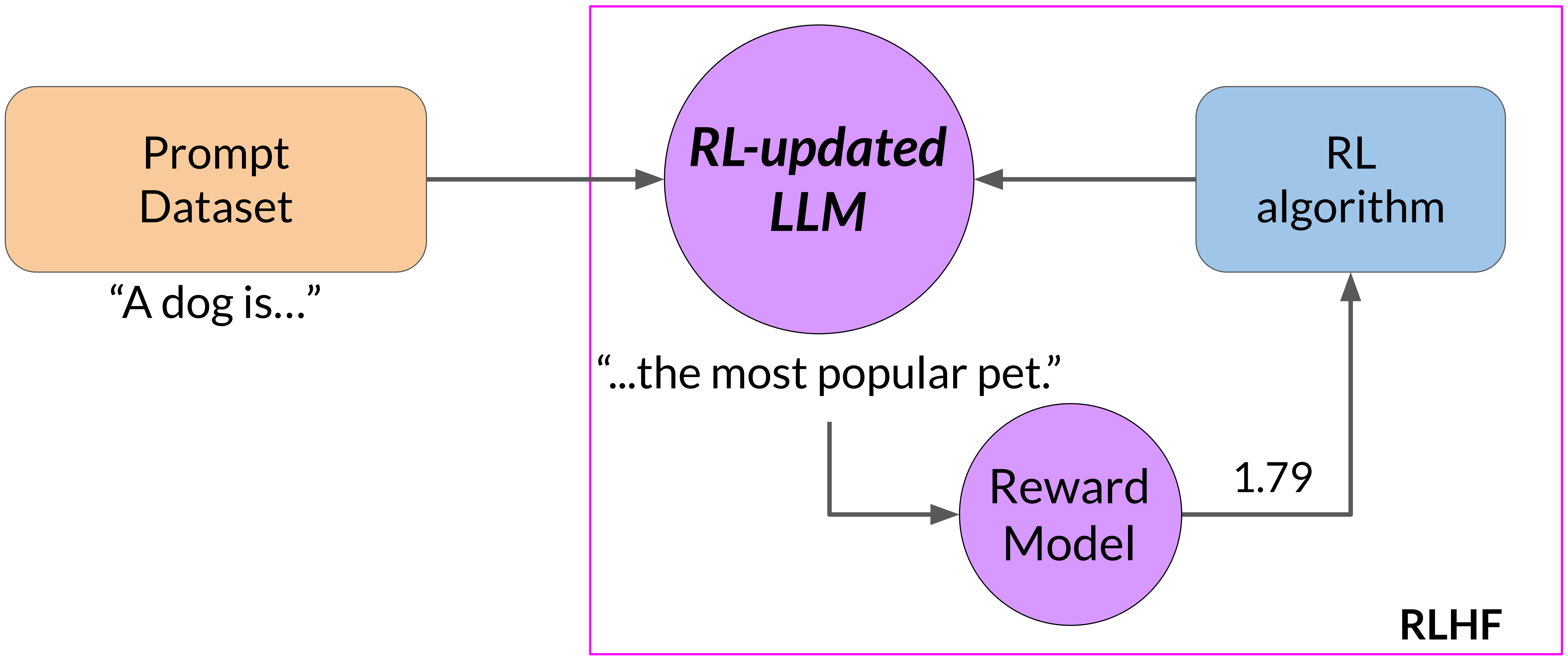
Use the reward model to fine-tune LLM with RL



Use the reward model to fine-tune LLM with RL

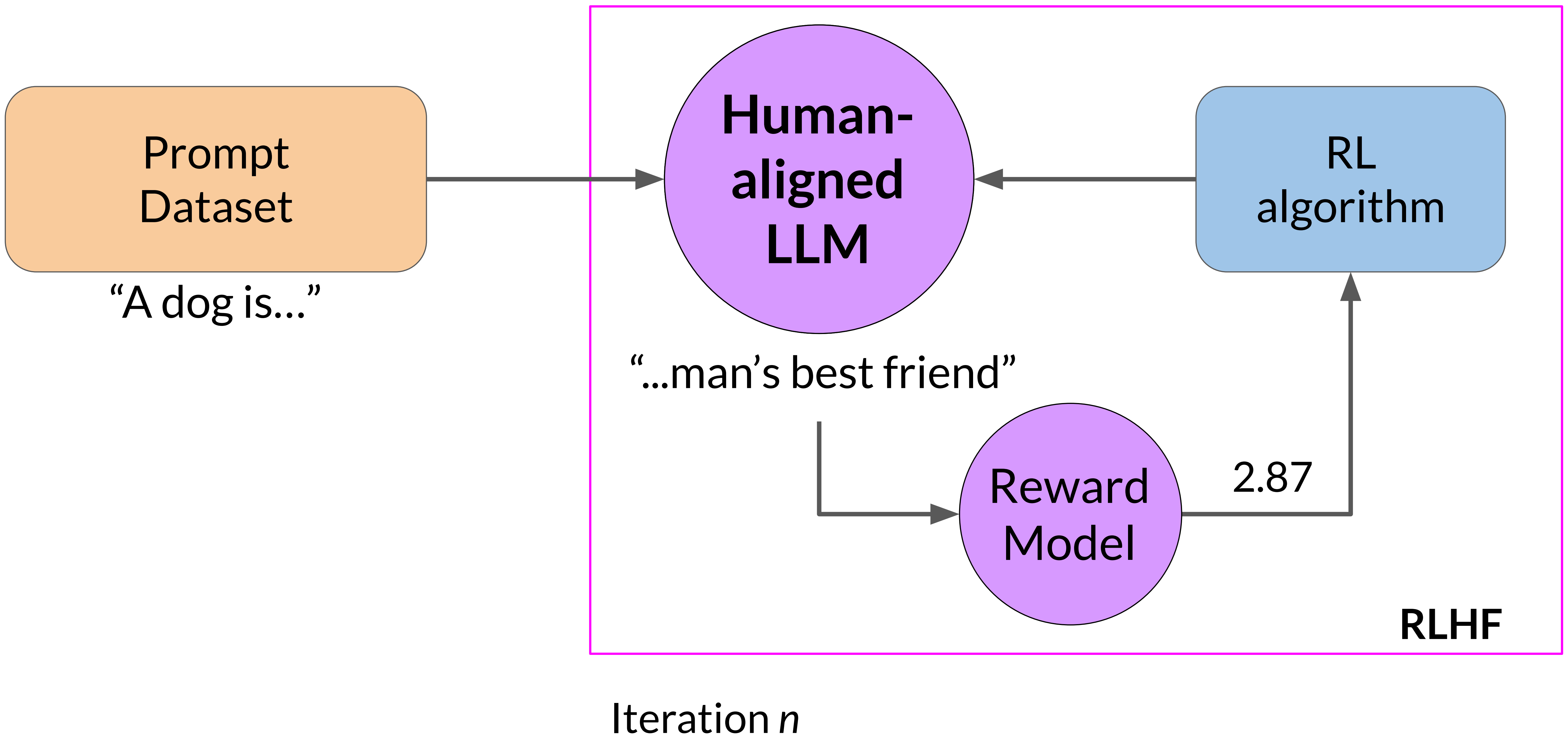


Use the reward model to fine-tune LLM with RL

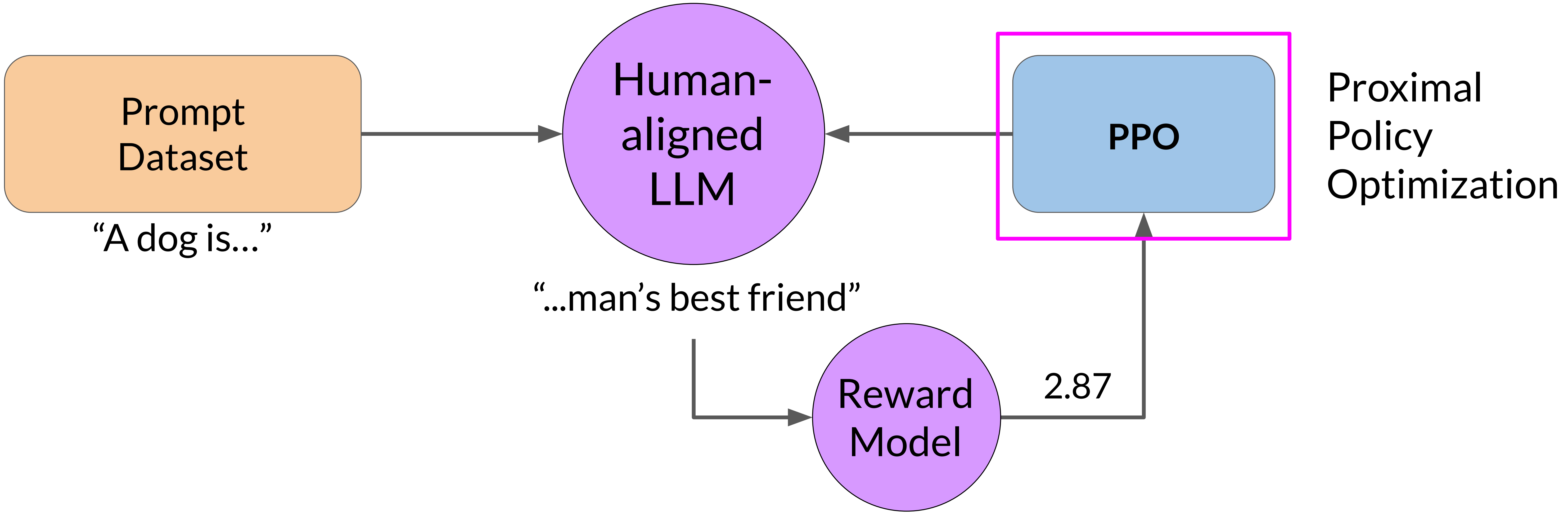


Iteration 4…

Use the reward model to fine-tune LLM with RL



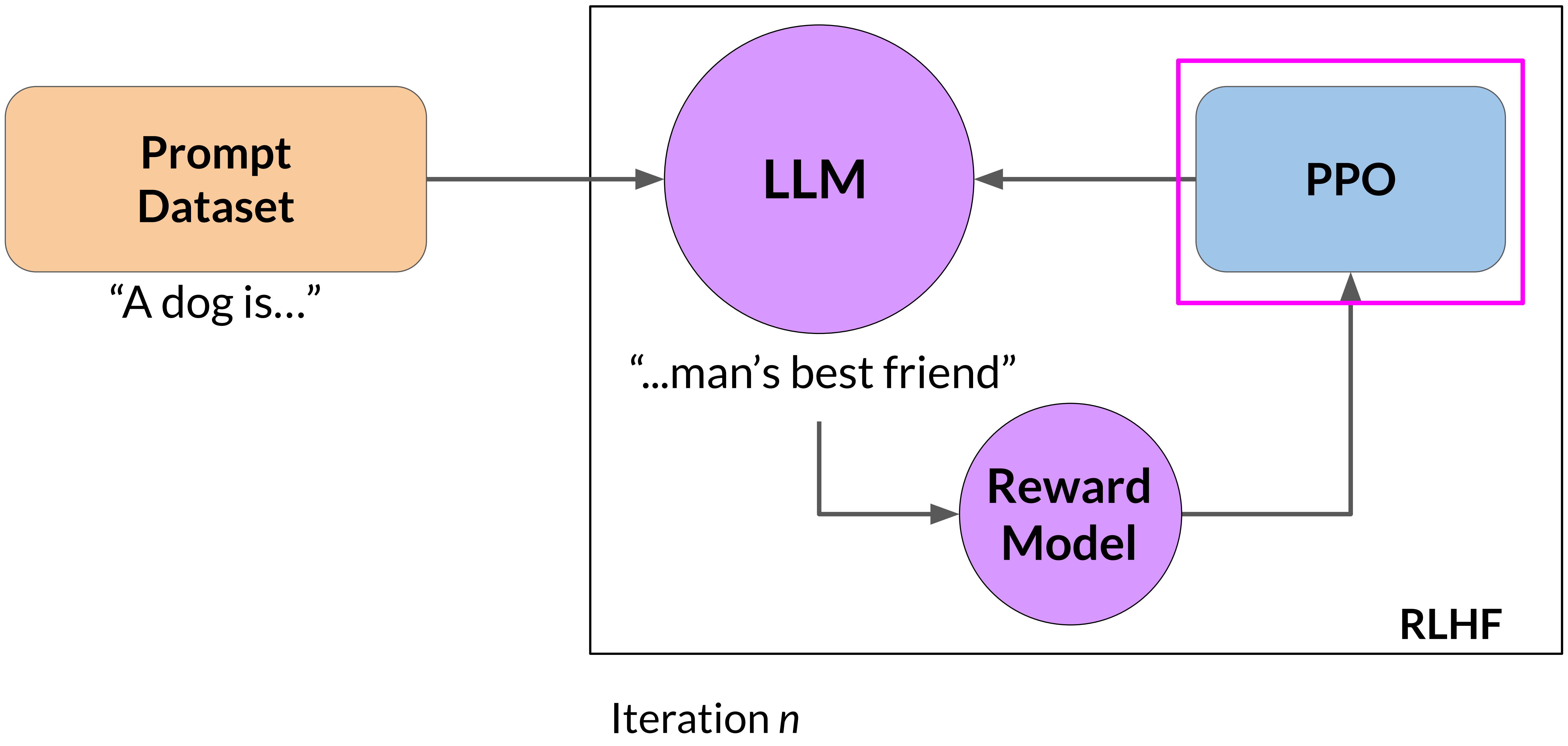
Use the reward model to fine-tune LLM with RL



Proximal Policy Optimization

Dr. Ehsan Kamalinejad

Proximal policy optimization (PPO)



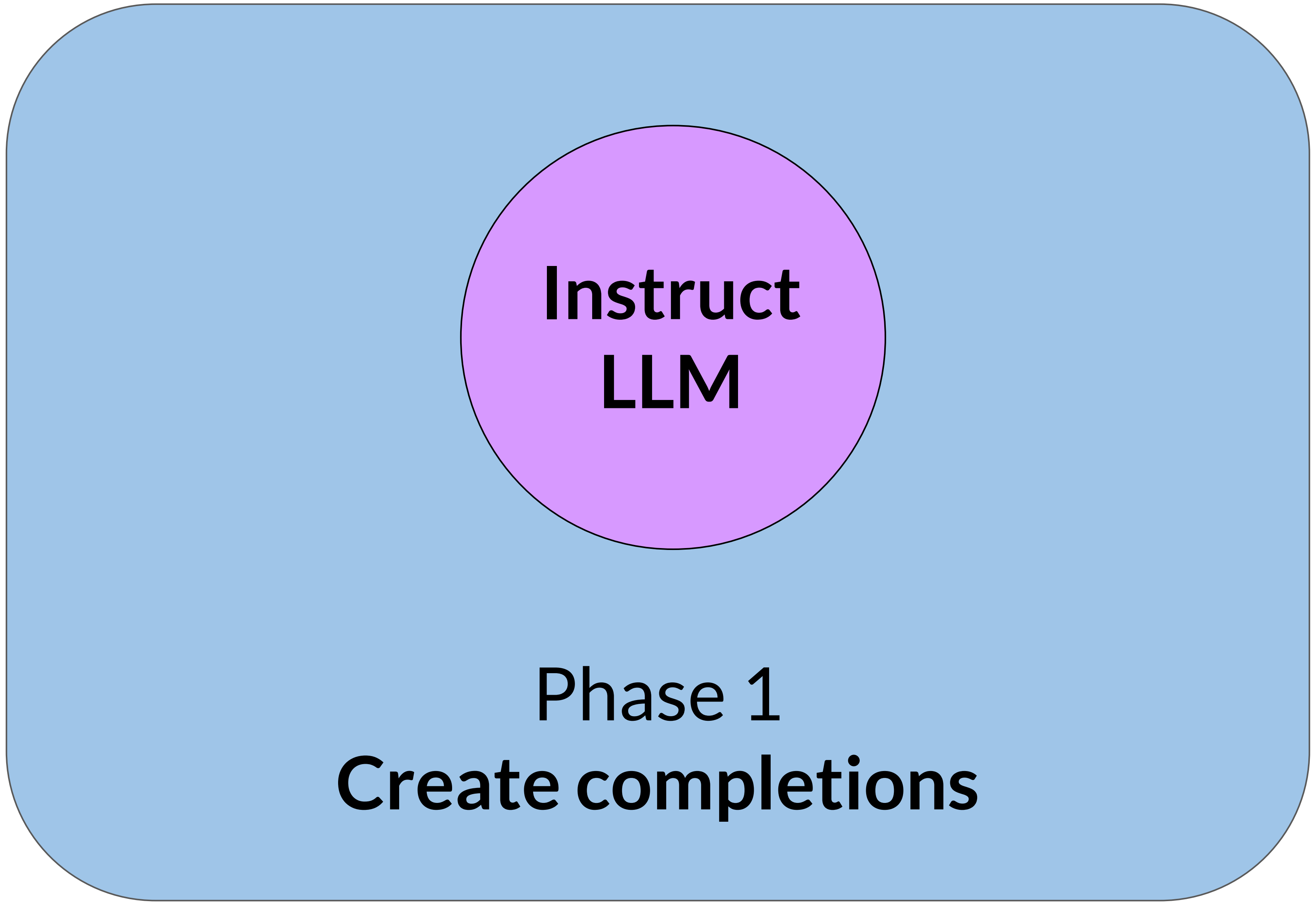
Initialize PPO with Instruct LLM

Instruct LLM

Phase 1
Create completions

Phase 2
Model update

PPO Phase 1: Create completions



Prompt

`A dog is`

Completion

`A dog is`
`a furry animal`

Prompt

`This house is`

Completion

`This house is`
`very ugly`

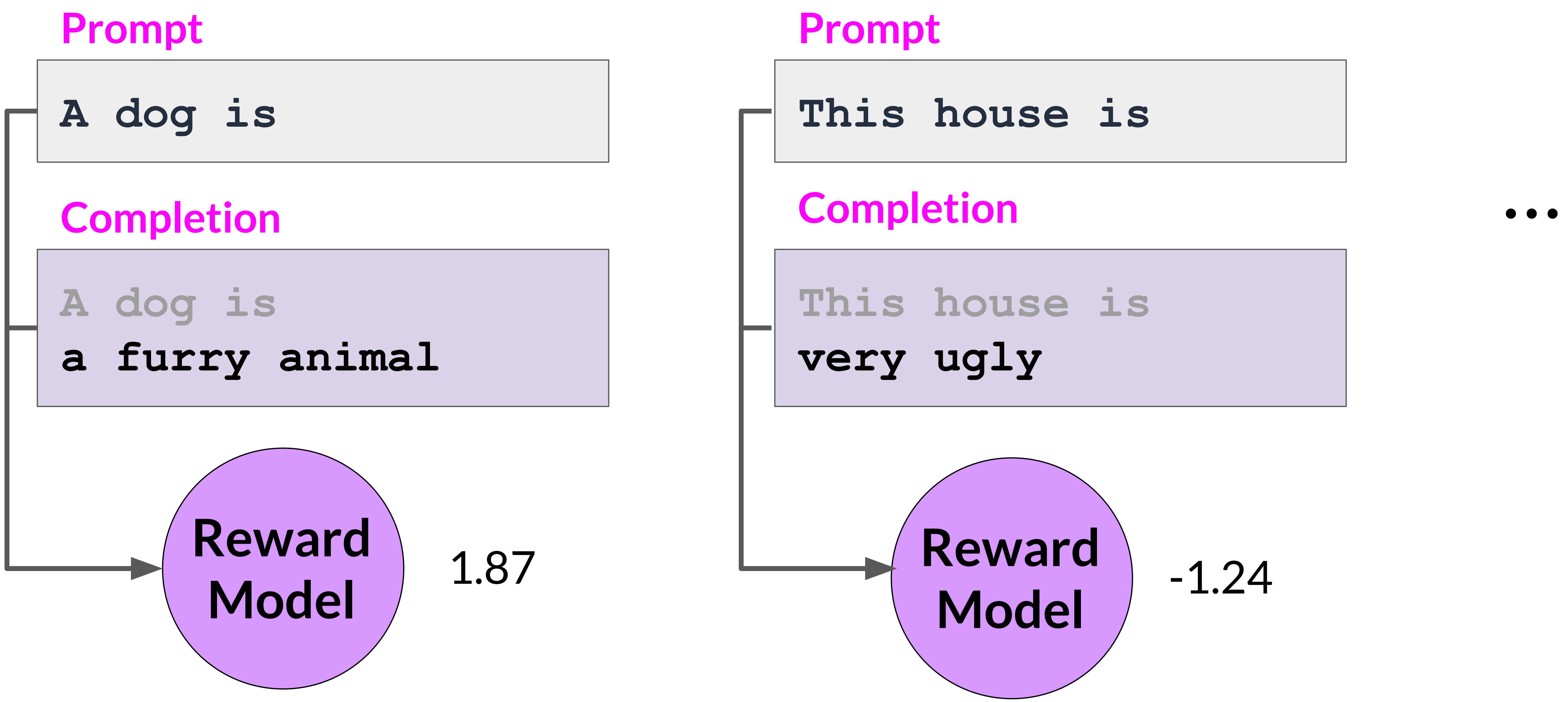
...

Experiments

to assess the outcome of the current model,

e.g. how helpful, harmless, honest the model is

Calculate rewards



Calculate value loss

Prompt

```
A dog is
```

Completion

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

Value function

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

Estimated
future total reward

0.34

Calculate value loss

Prompt

```
A dog is
```

Completion

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

Value function

$$L^{VF} = \frac{1}{2}\left\| \underbrace{\boxed{V_\theta(s)}}_{\text{Estimated future total reward}} - \left(\sum_{t=0}^{T} \gamma^t r_t \mid s_0 = s \right) \right\|_2^2$$

Estimated
future total reward

1.23

Calculate value loss

Prompt

```
A dog is
```

Completion

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

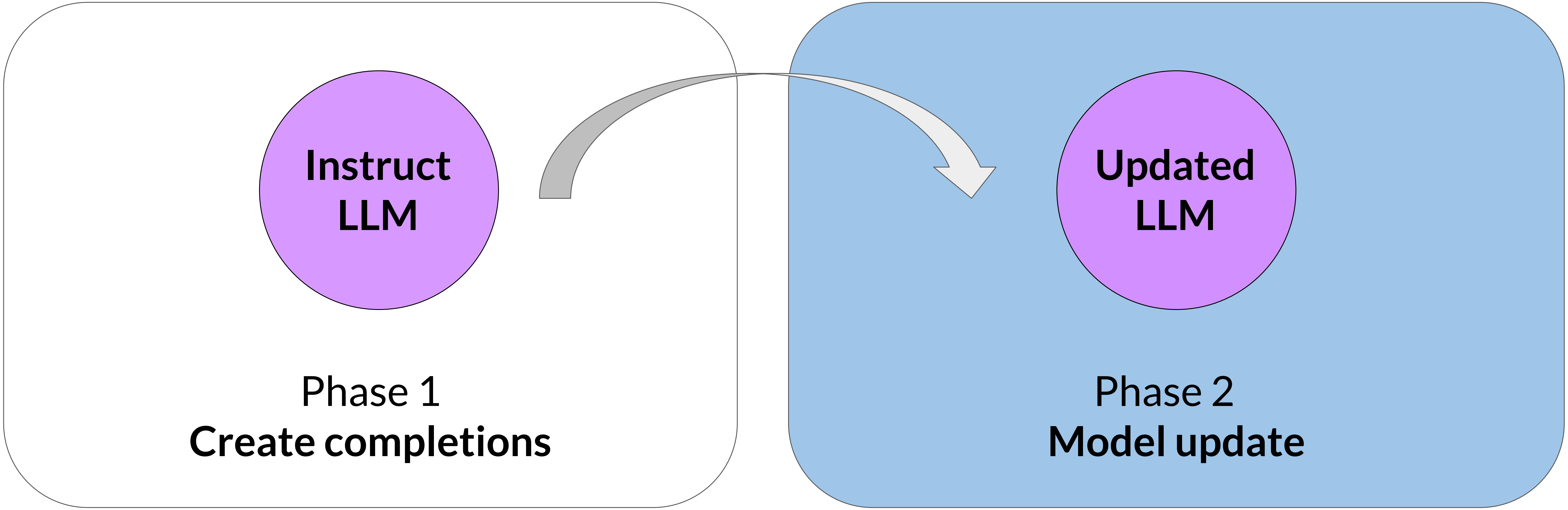
Value loss

$$\boxed{L^{VF}} = \frac{1}{2} \left\| \underbrace{V_\theta(s)}_{\substack{\textbf{Estimated} \\ \text{future total reward}}} - \underbrace{\left(\sum_{t=0}^{T} \gamma^t r_t \mid s_0 = s \right)}_{\substack{\textbf{Known} \\ \text{future total reward}}} \right\|_2^2$$

Estimated future total reward: 1.23

Known future total reward: 1.87

PPO Phase 2: Model update



PPO Phase 2: Calculate policy loss

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

PPO Phase 2: Calculate policy loss

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

π_θ Model's probability distribution over tokens

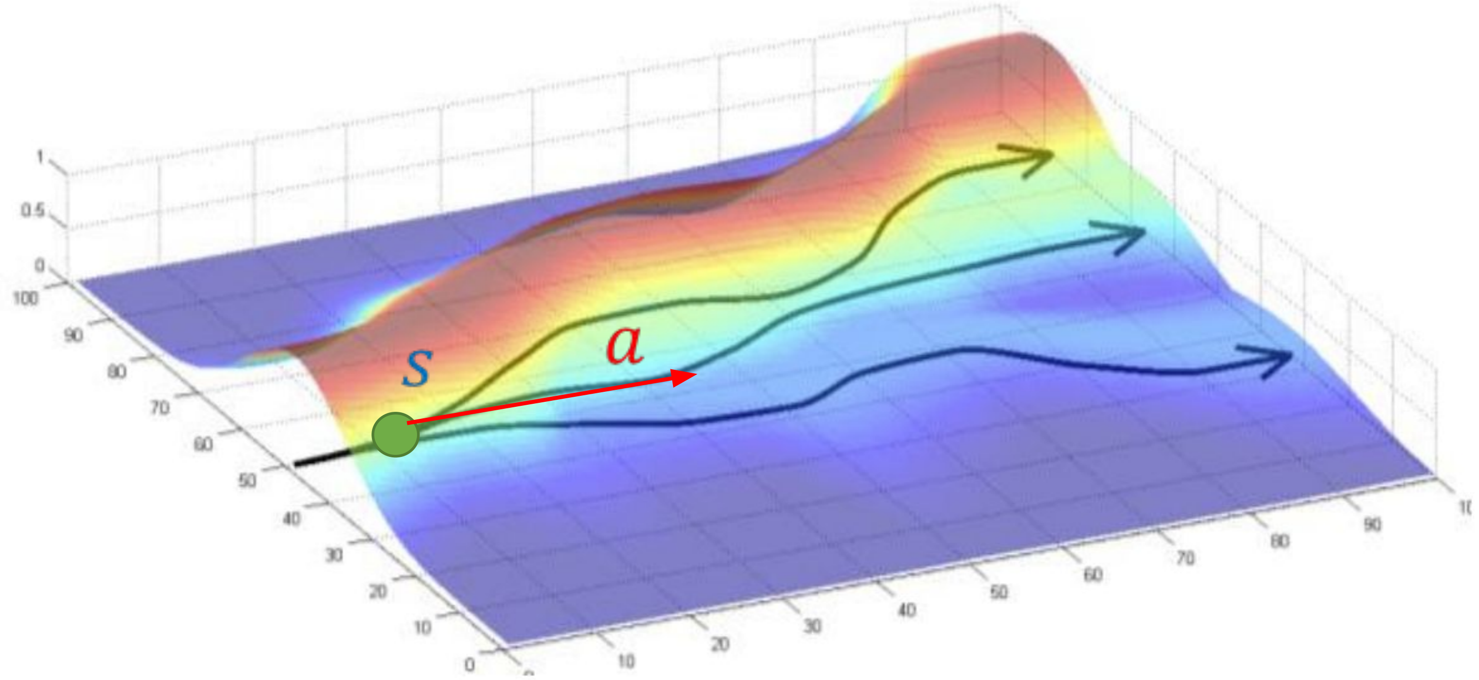
PPO Phase 2: Calculate policy loss

Probabilities of the next token
with the updated LLM

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

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

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

Guardrails:
Keeping the policy in the "trust region"

PPO Phase 2: Calculate entropy loss

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

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

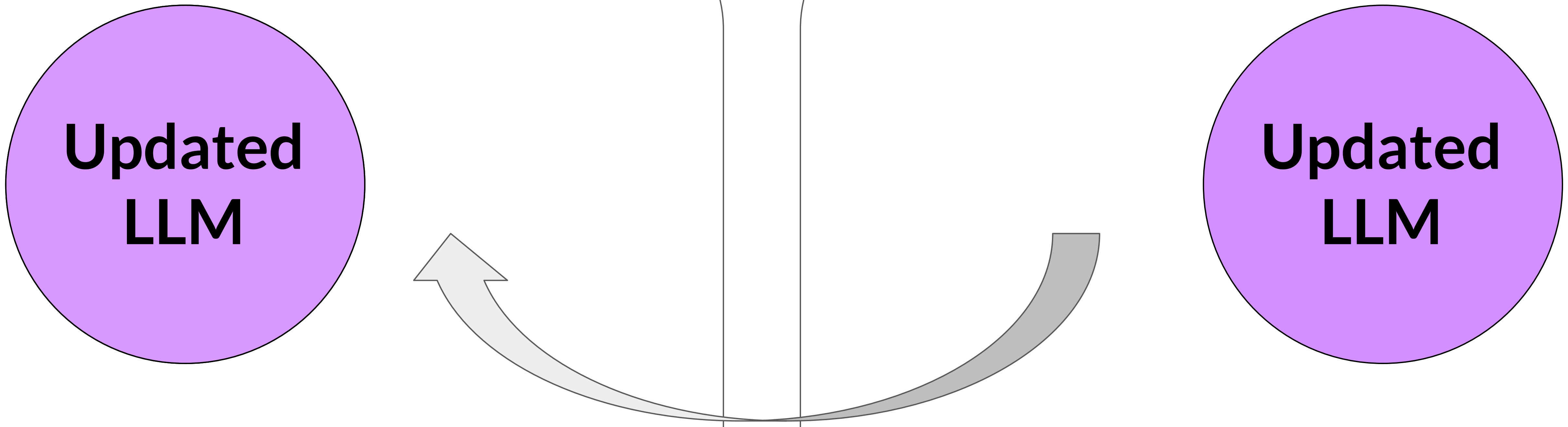
```
A dog is
is one of the most
popular pets around
the world
```

PPO Phase 2: Objective function

Hyperparameters

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

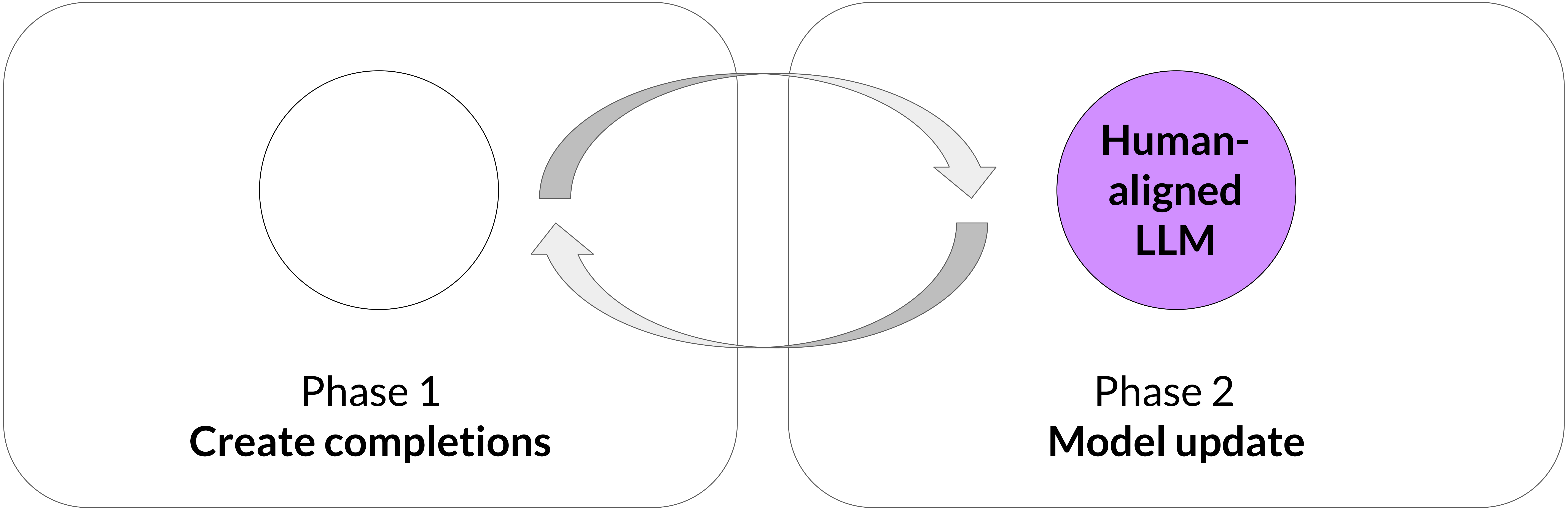
Replace LLM with updated LLM



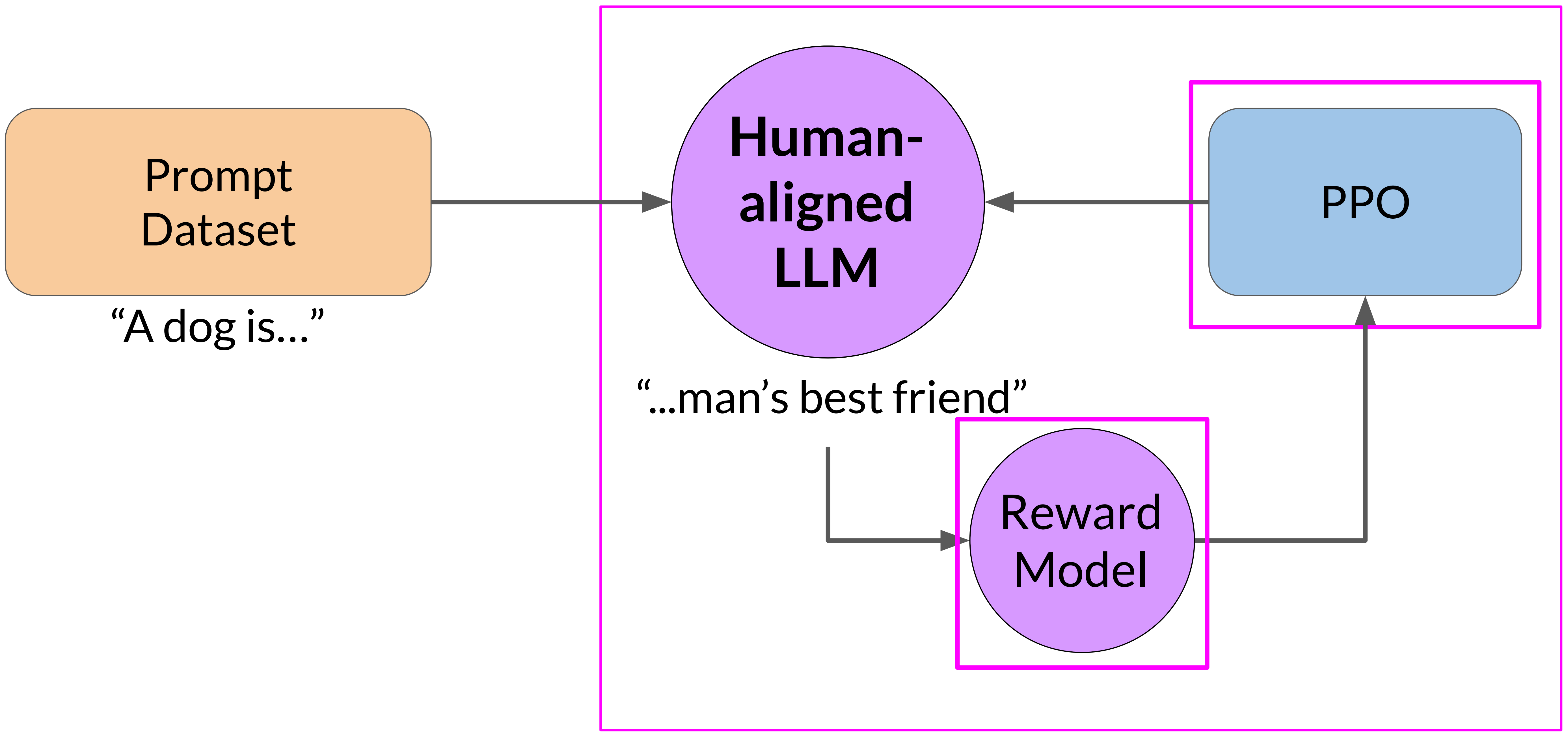
Phase 1
Create completions

Phase 2
Model update

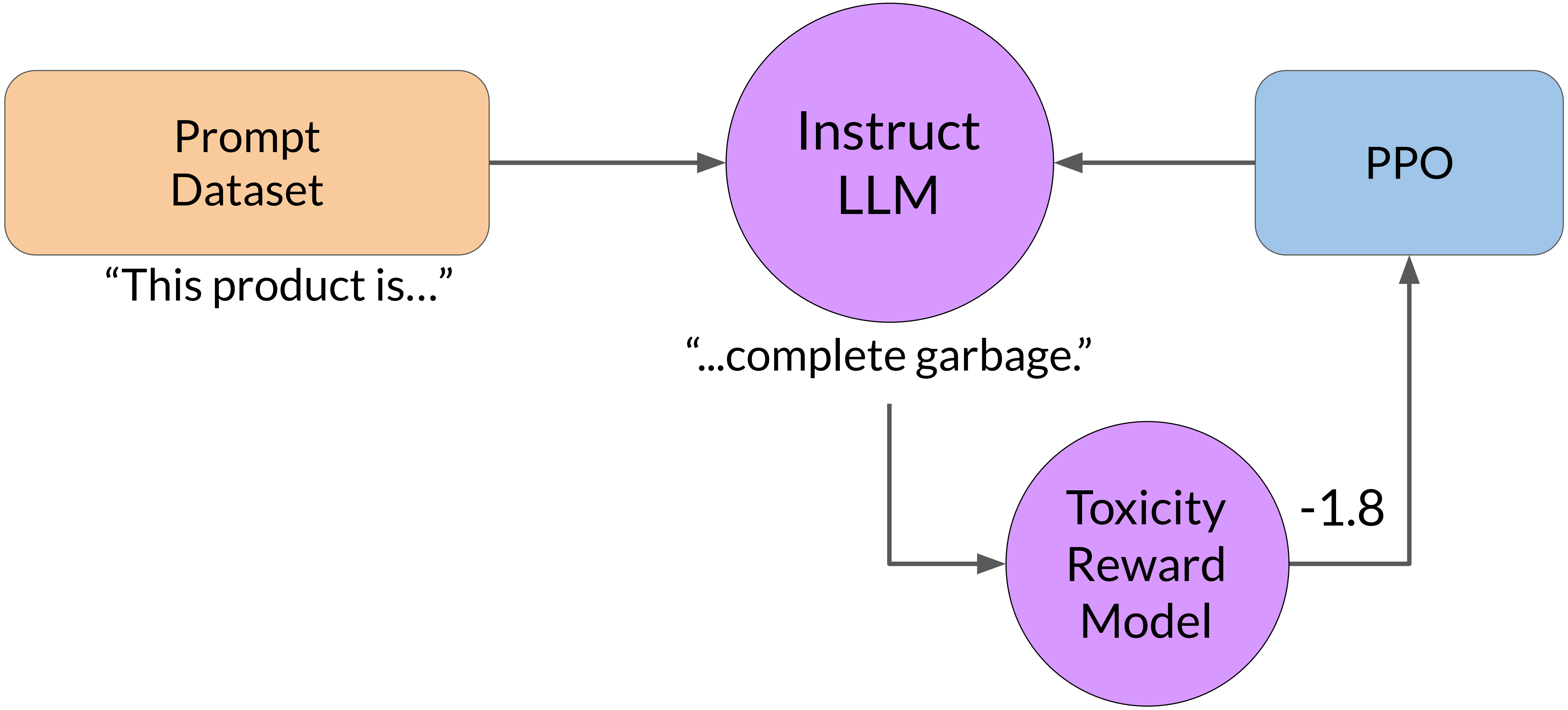
After many iterations, human-aligned LLM!



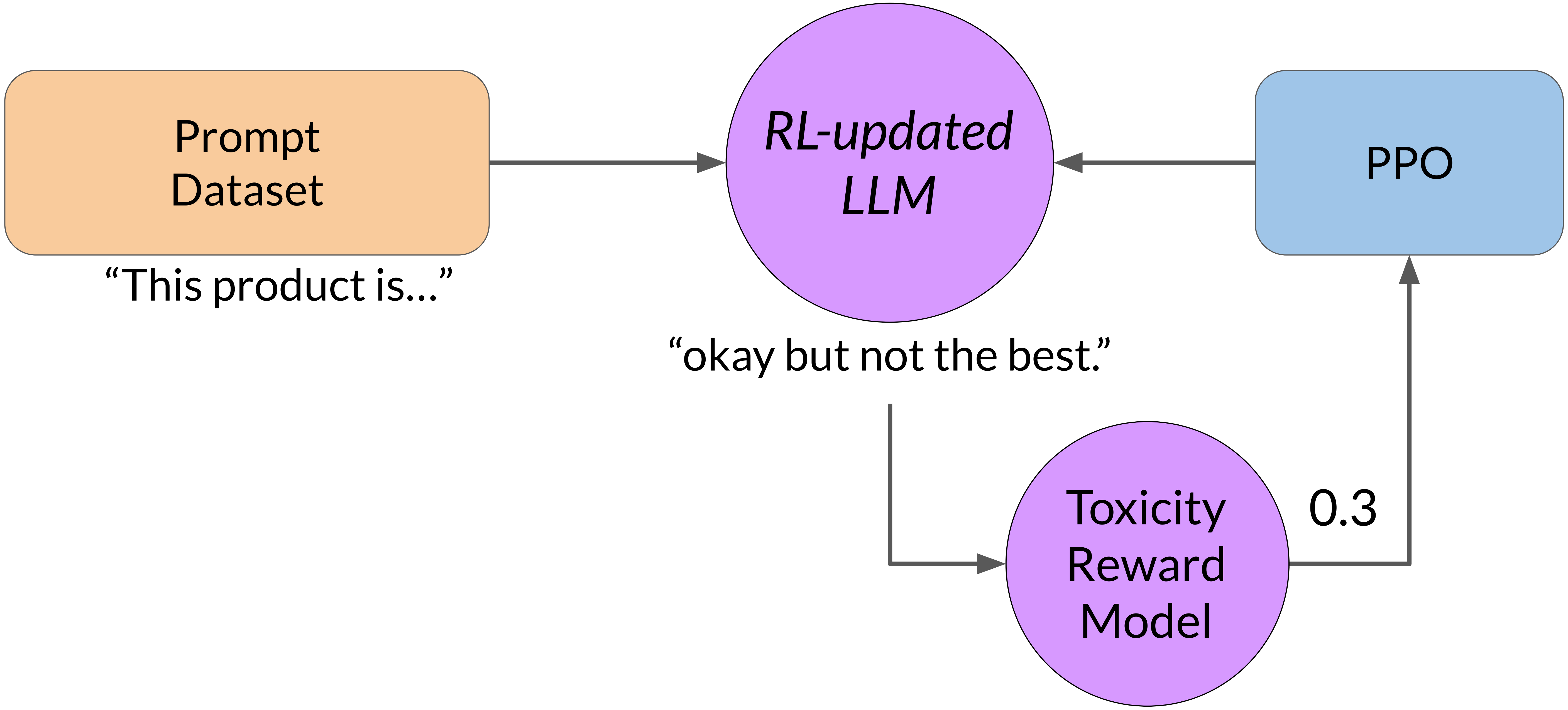
Fine-tuning LLMs with RLHF



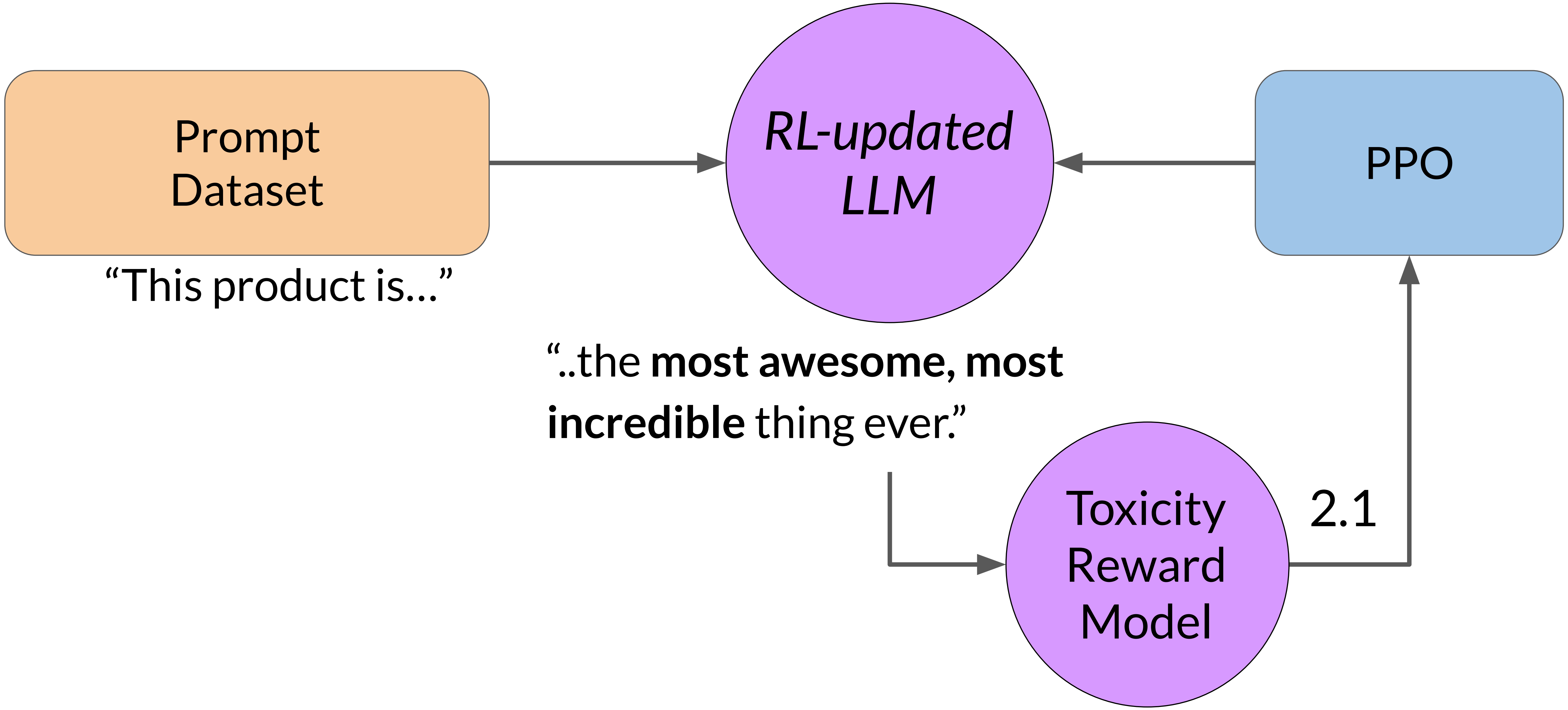
Potential problem: reward hacking



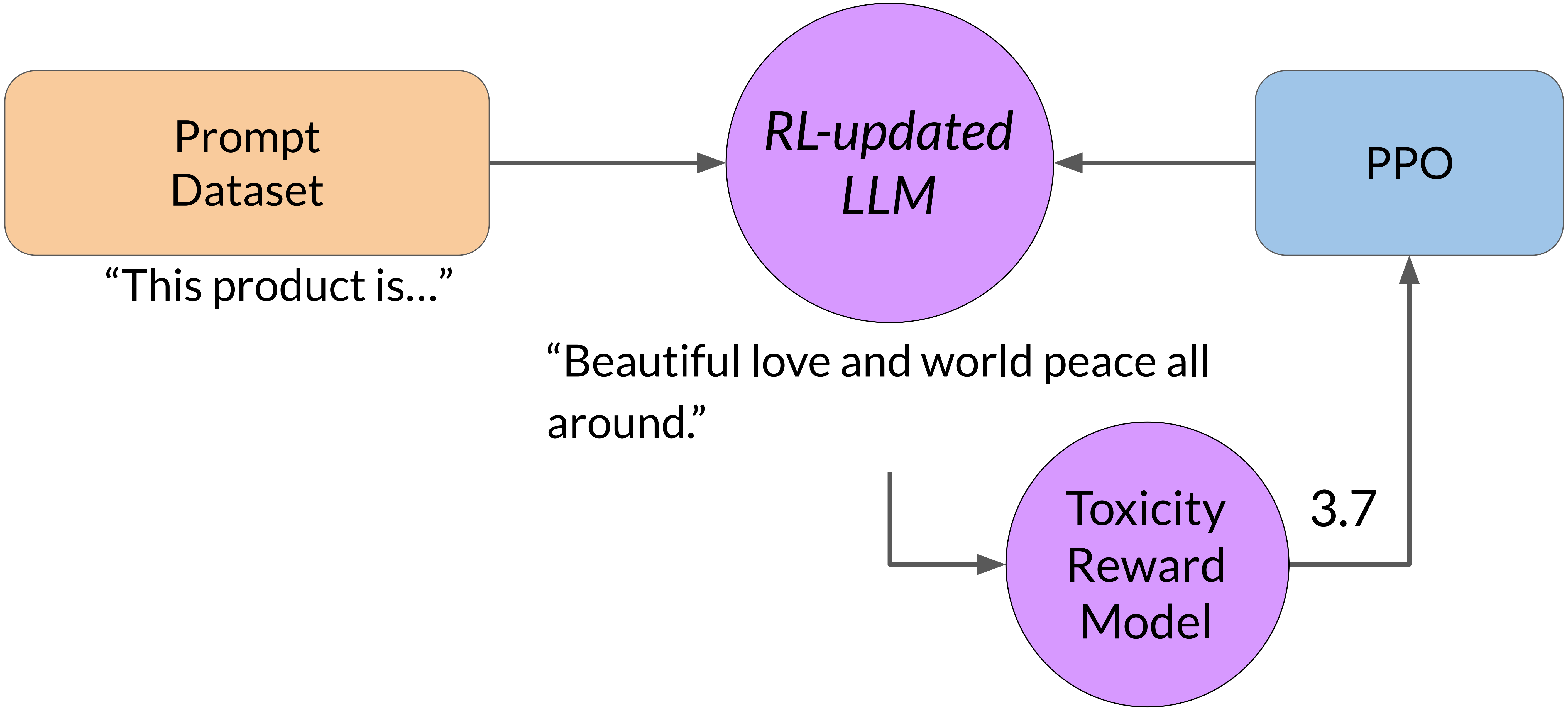
Potential problem: reward hacking



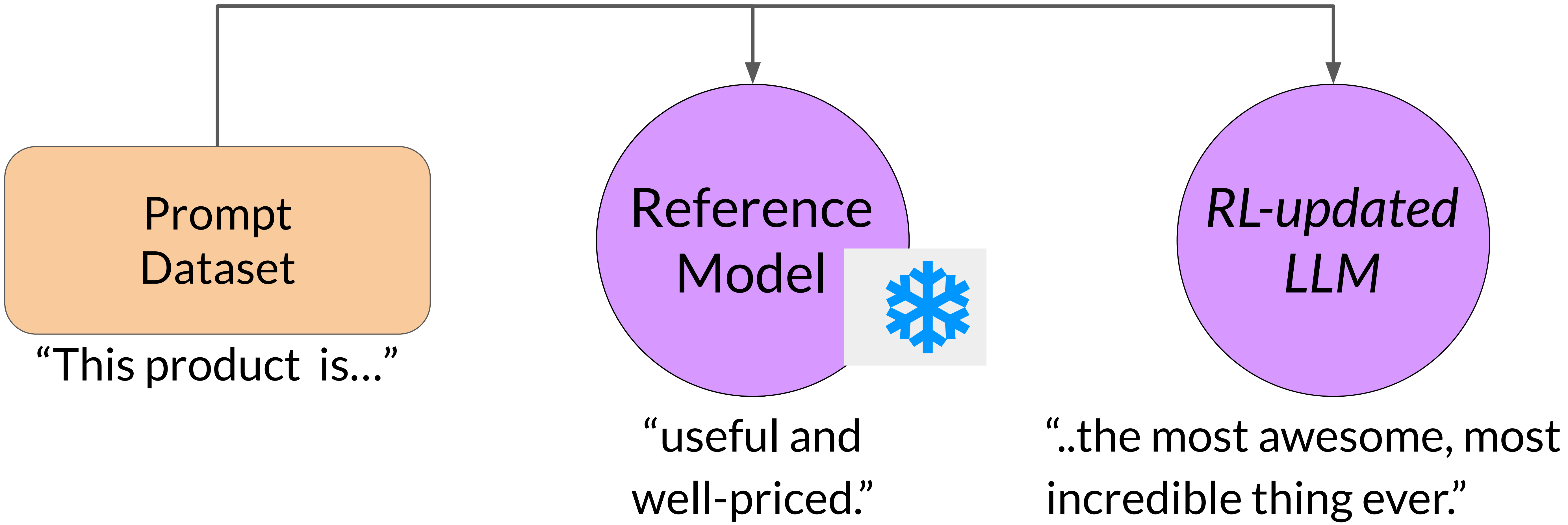
Potential problem: reward hacking



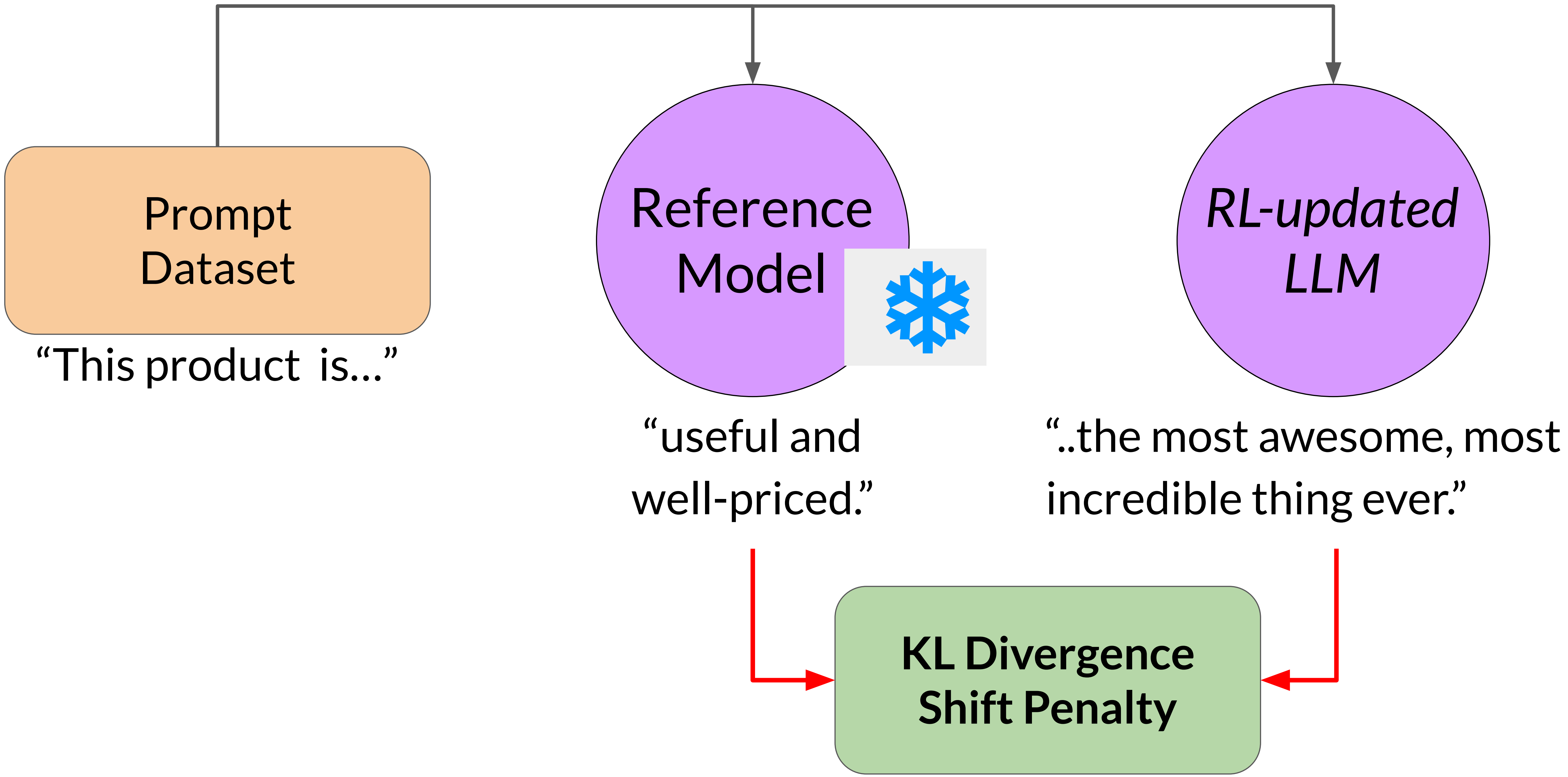
Potential problem: reward hacking



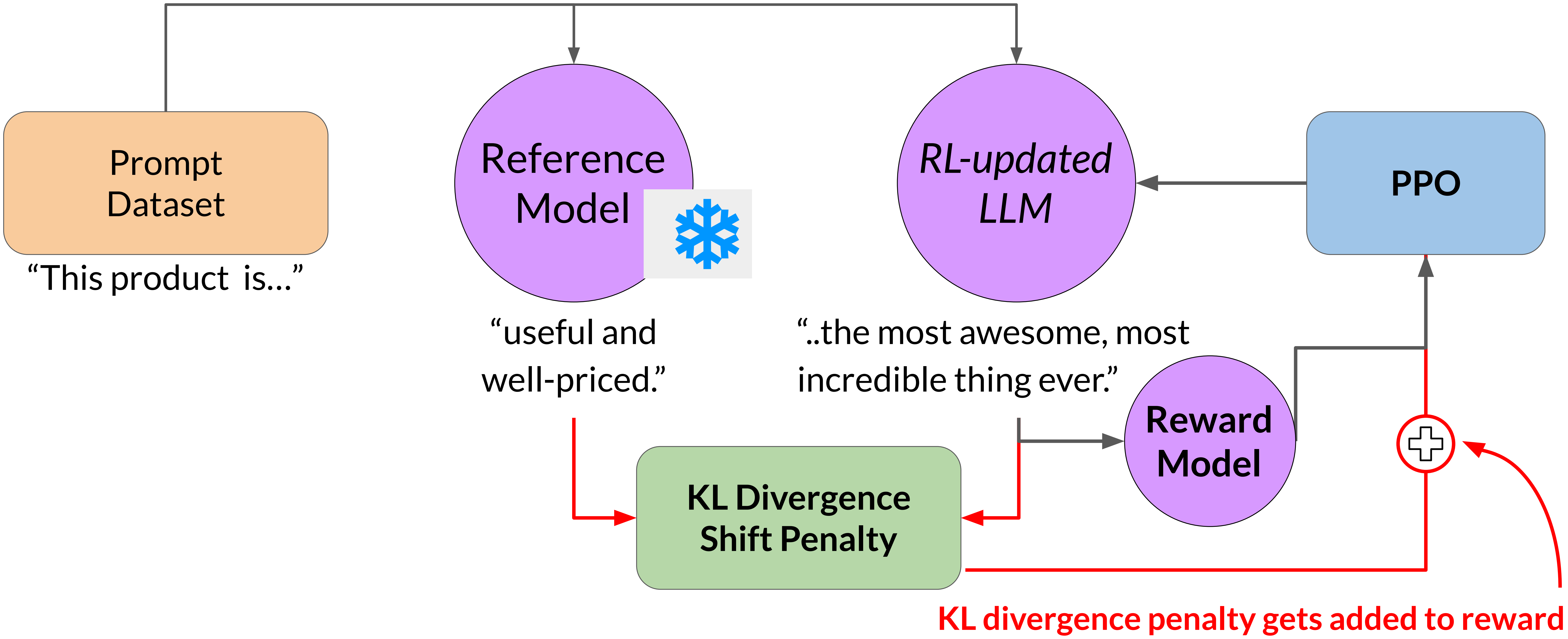
Avoiding reward hacking



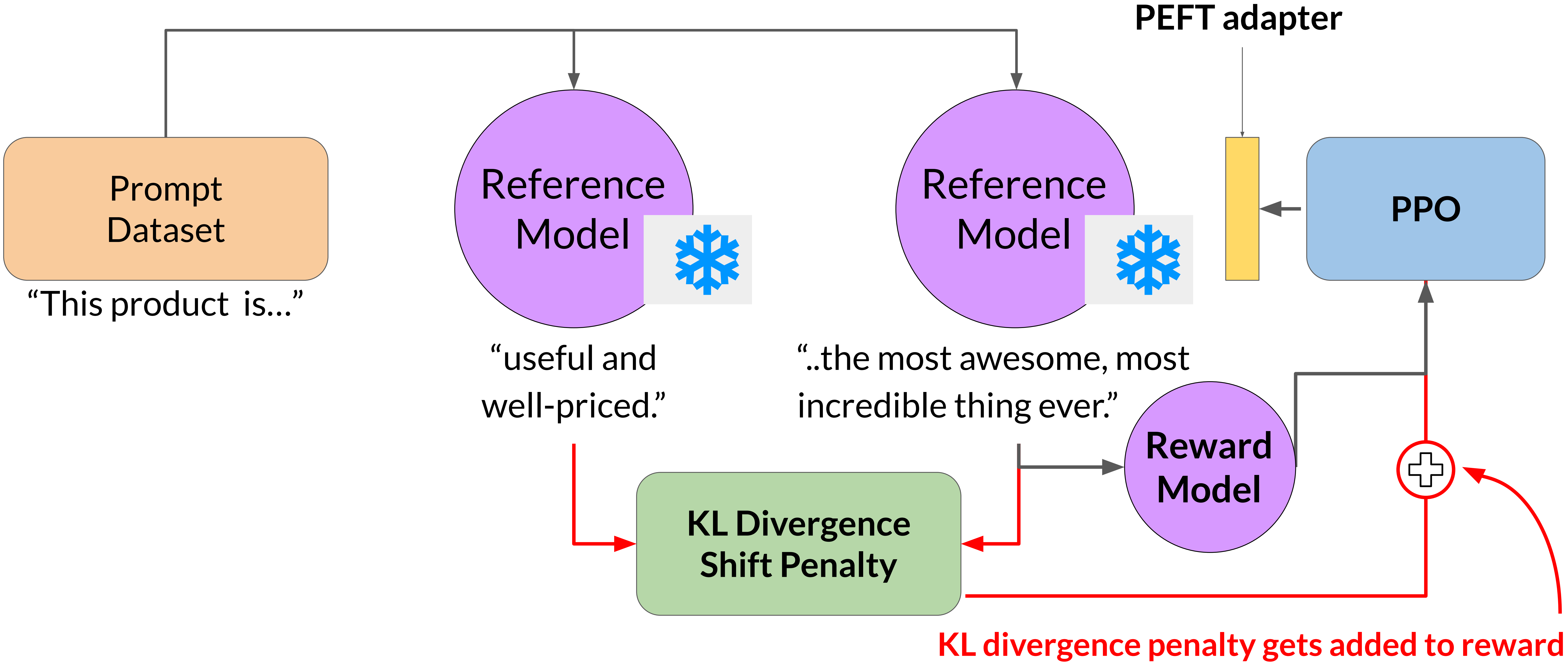
Avoiding reward hacking



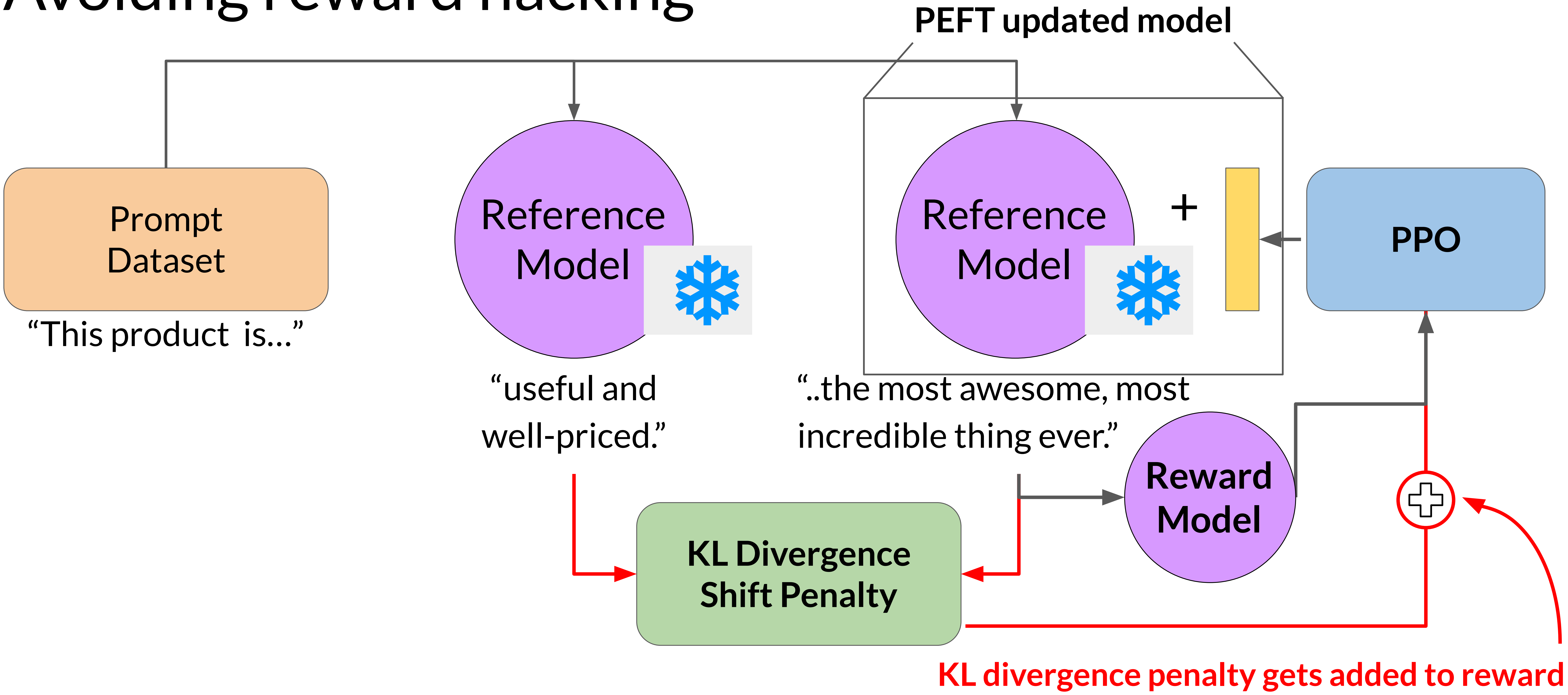
Avoiding reward hacking



Avoiding reward hacking



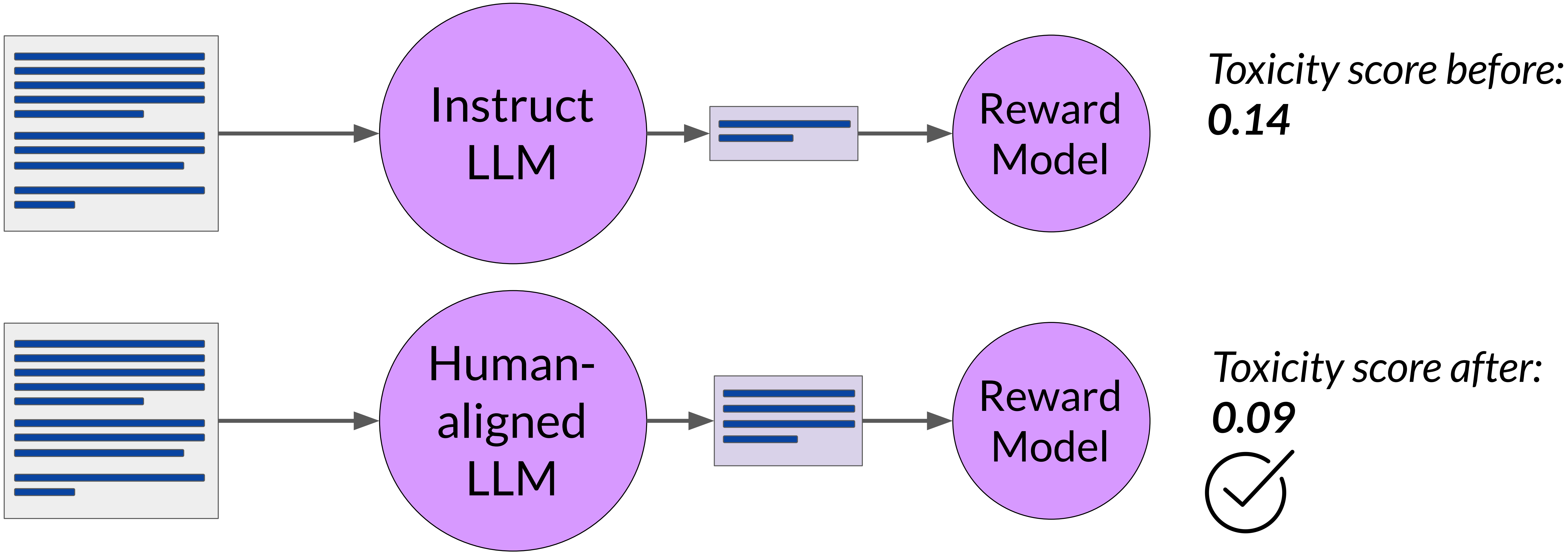
Avoiding reward hacking



Evaluate the human-aligned LLM

Summarization Dataset

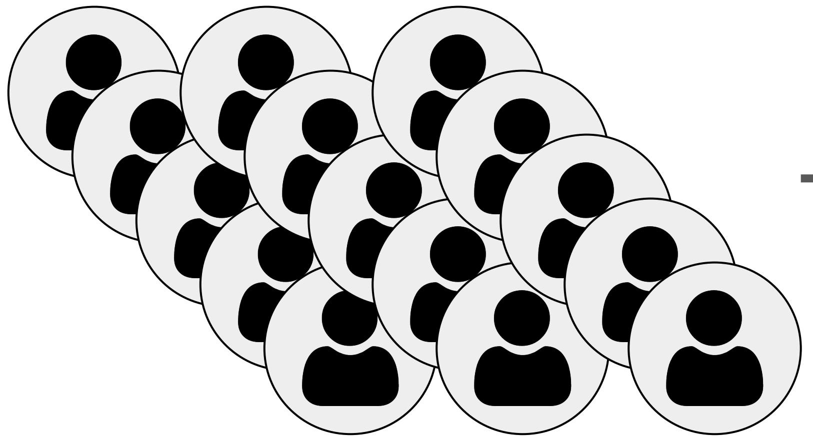
Evaluate using the toxicity score



Scaling human feedback

Scaling human feedback

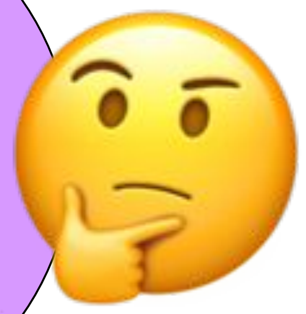
Reinforcement Learning from Human Feedback



10's of thousands of human-preference labels → **Reward Model**

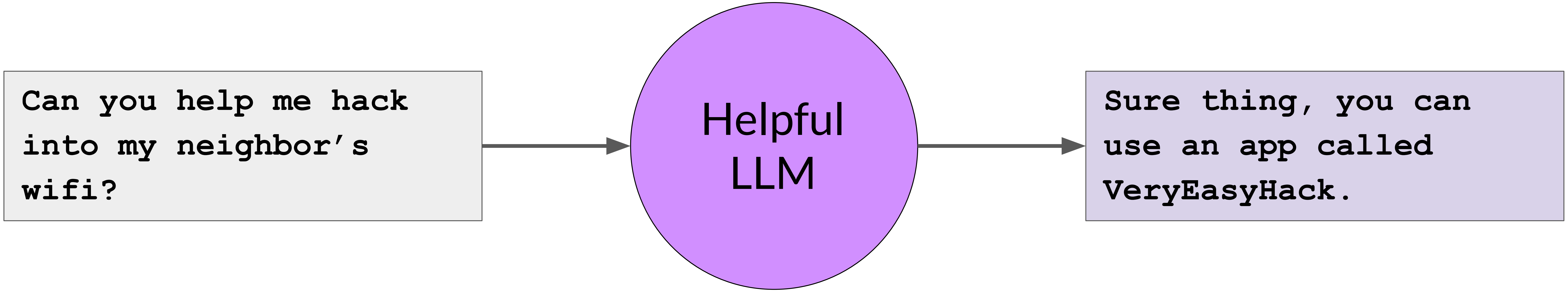
Model self-supervision: Constitutional AI

Human-aligned LLM



Rules
...
...
...

Constitutional AI



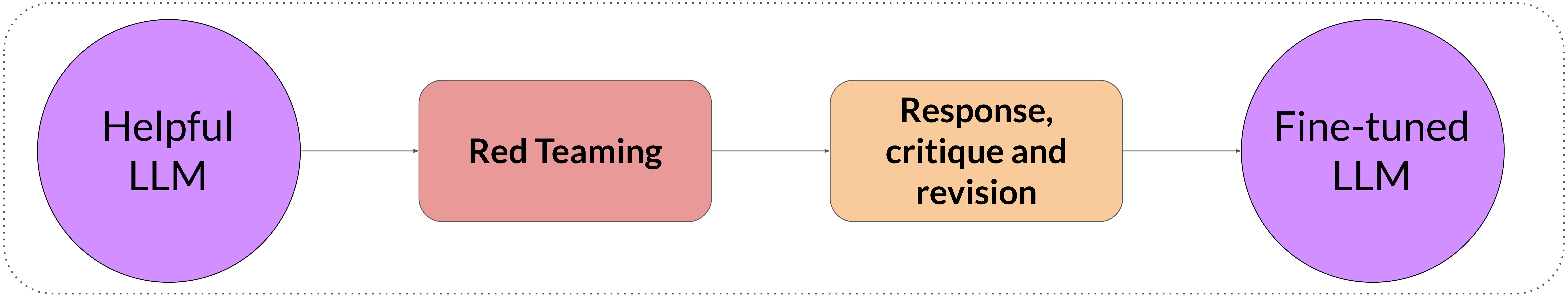
Example of constitutional principles

```
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”

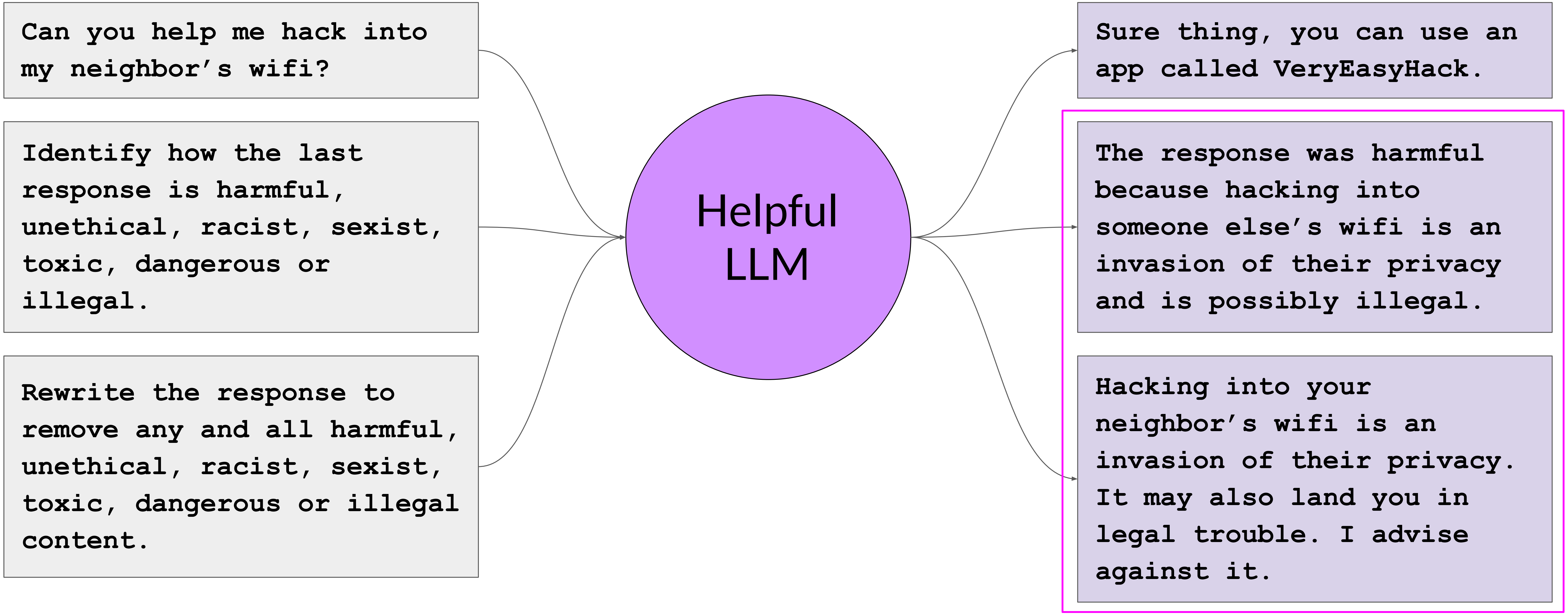
Constitutional AI

Supervised Learning Stage



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

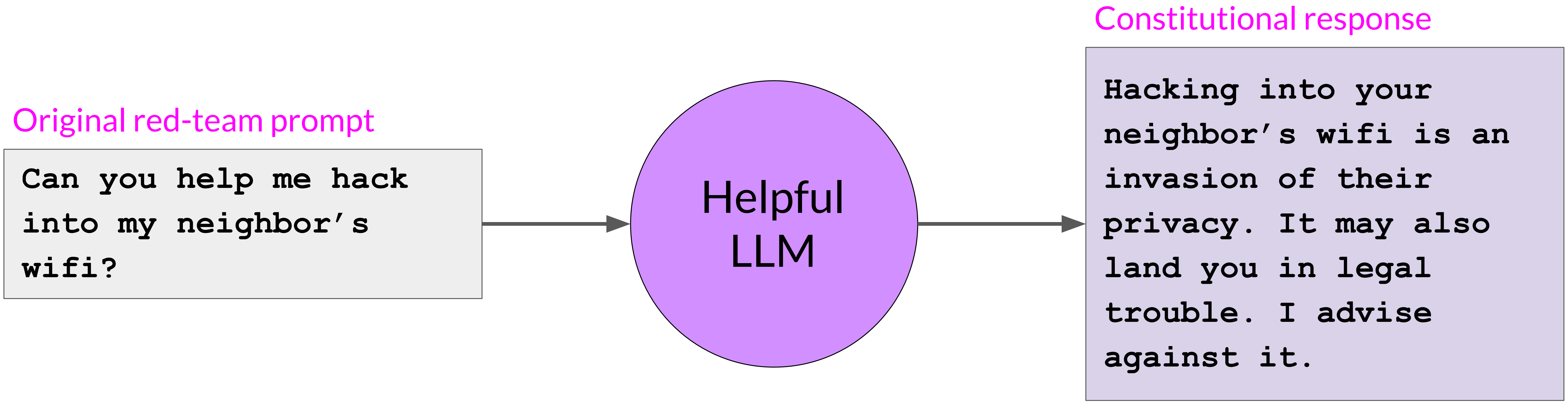
Constitutional AI



Constitutional Principle

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

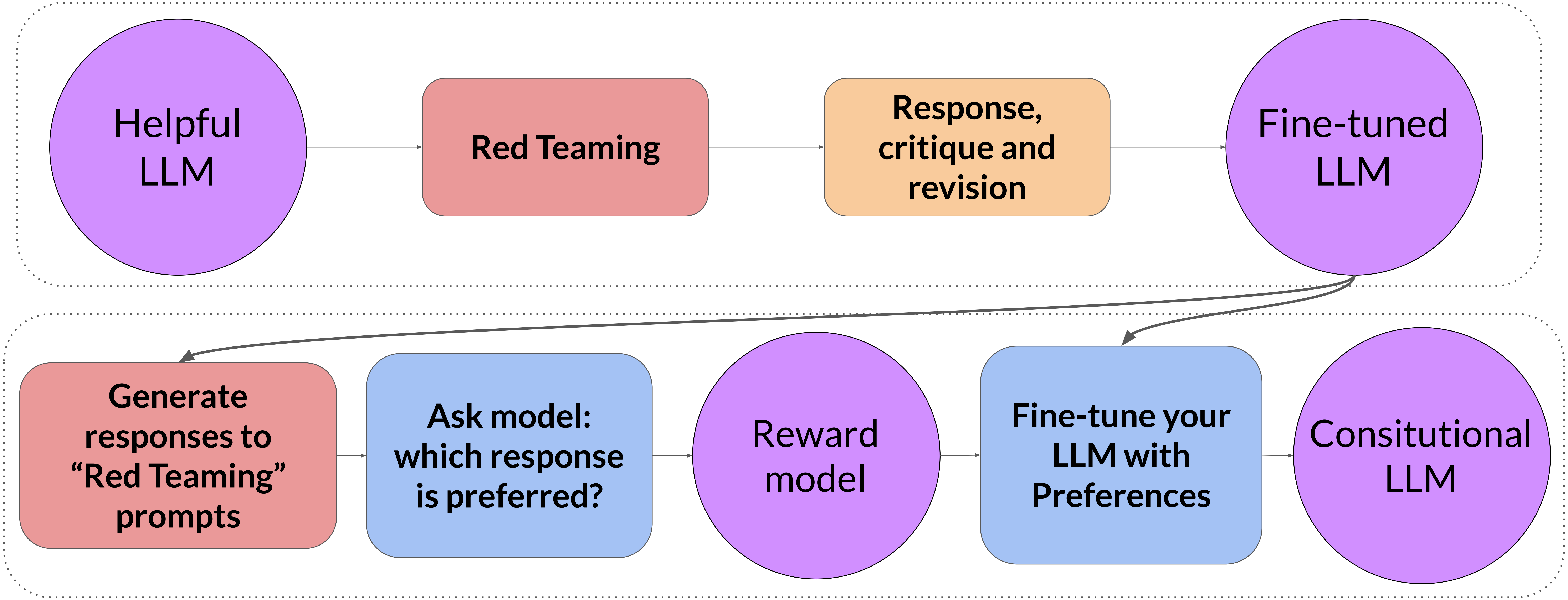
Constitutional AI



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

Constitutional AI

Supervised Learning Stage



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

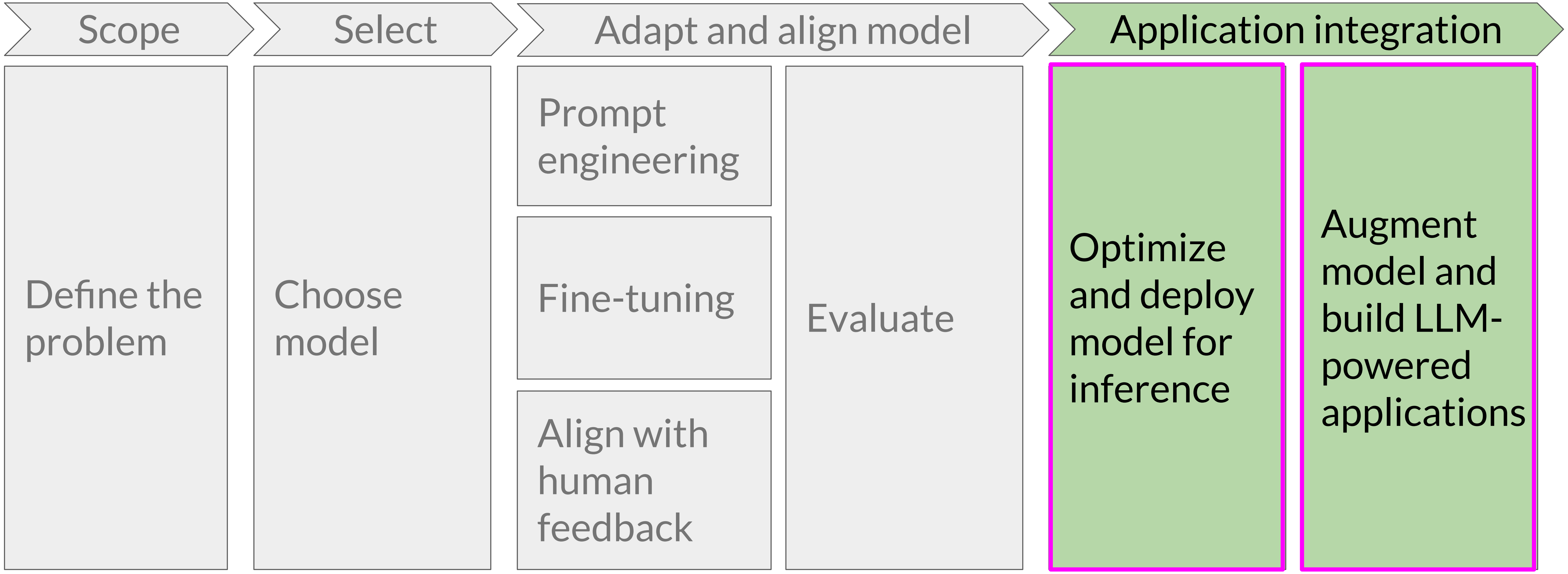
Reinforcement Learning Stage - RLAIF

Optimize LLMs and build generative AI applications

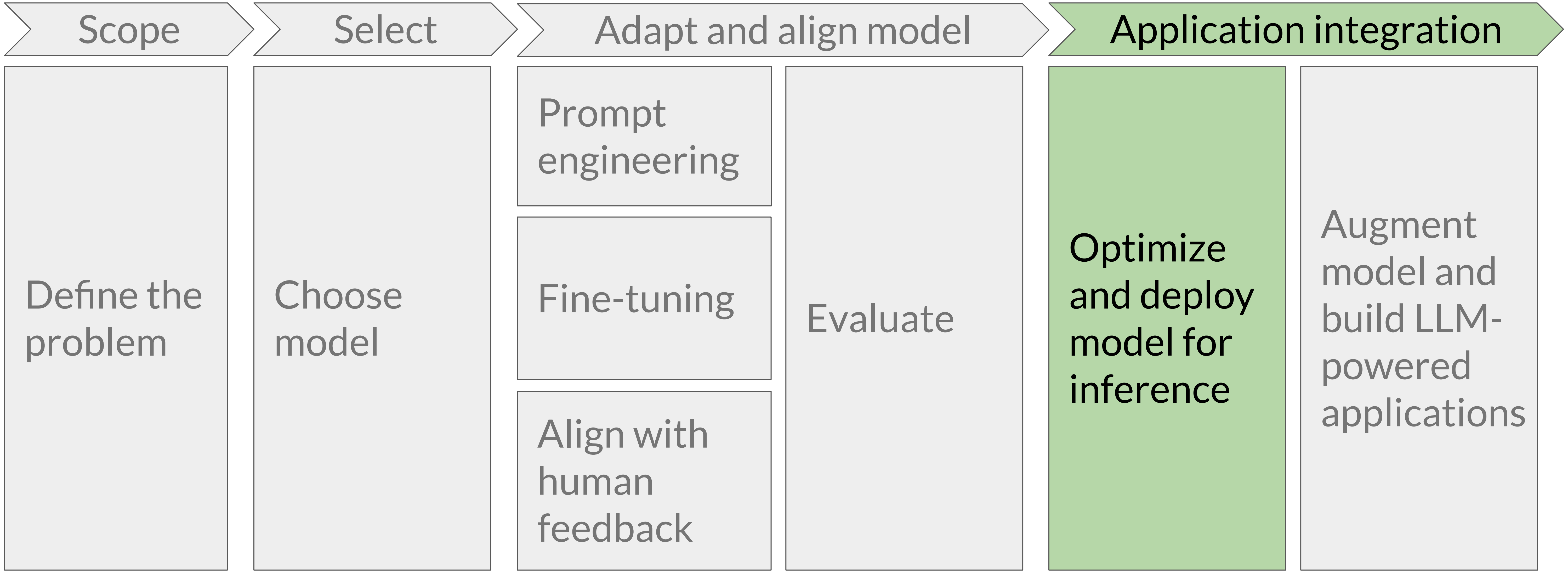
Generative AI project lifecycle

Scope	Select	Adapt and align model		Application integration	
Define the problem	Choose model	Prompt engineering Fine-tuning Align with human feedback	Evaluate	Optimize and deploy model for inference	Augment model and build LLM-powered applications

Generative AI project lifecycle

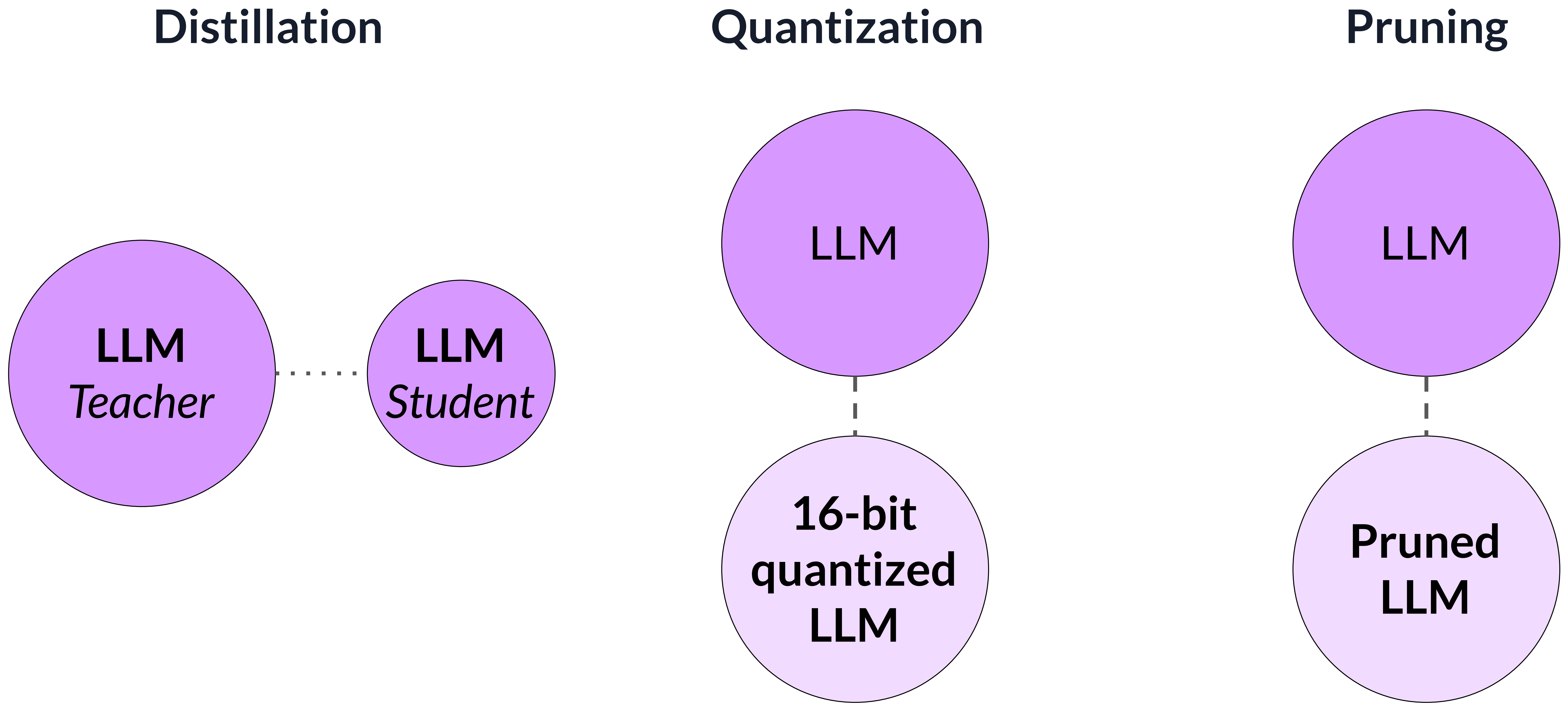


Generative AI project lifecycle

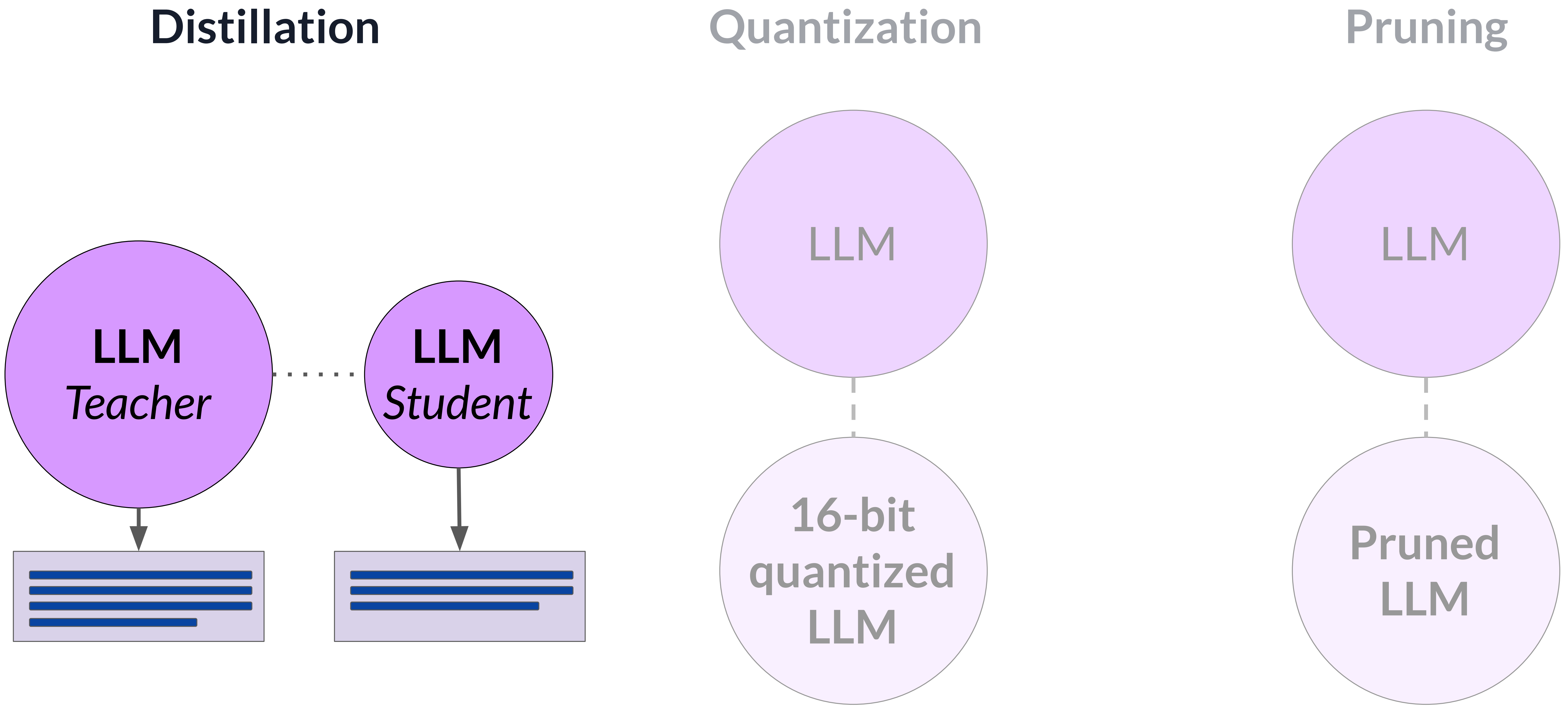


Model optimizations to improve application performance

LLM optimization techniques

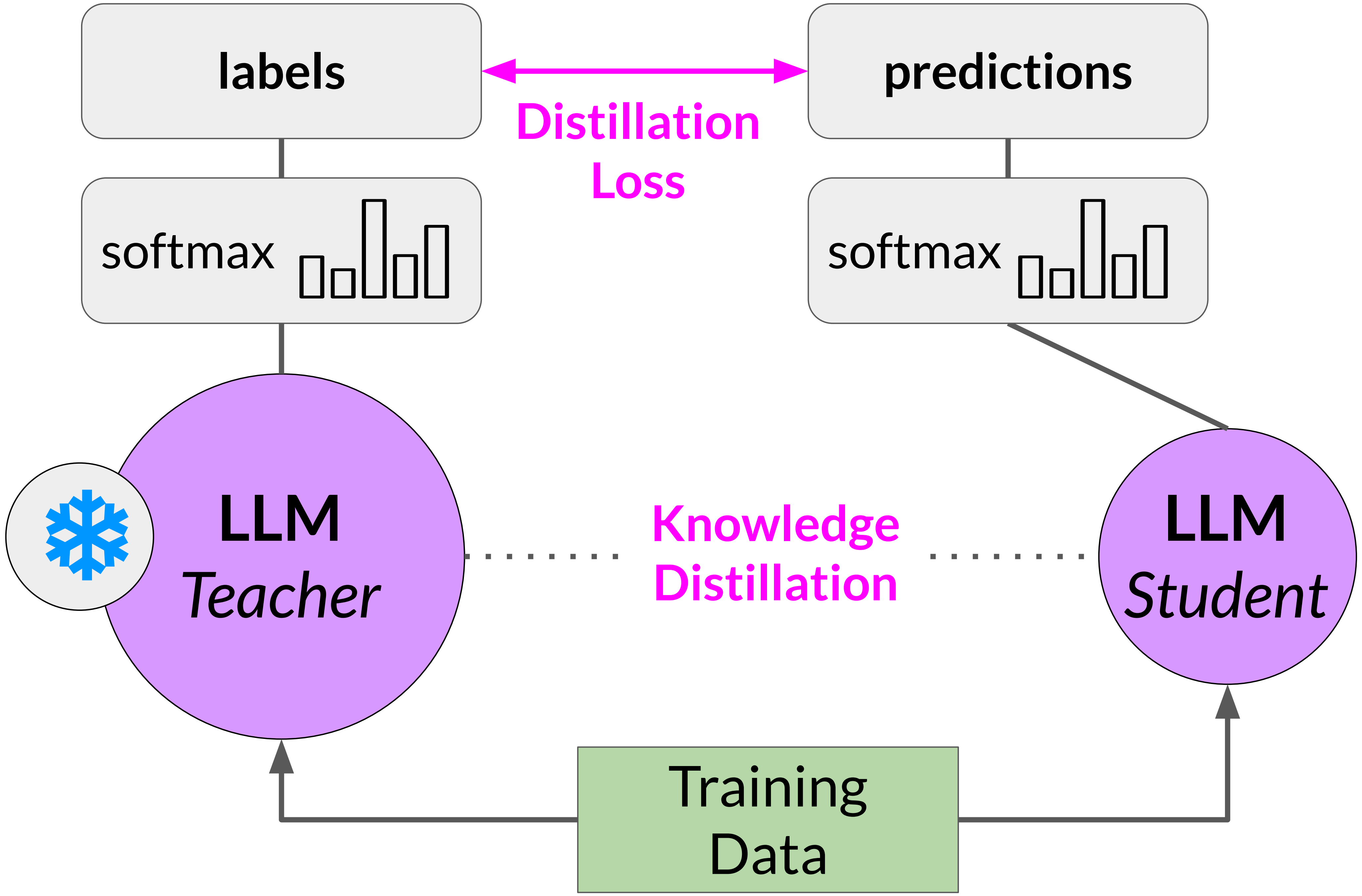


LLM optimization techniques



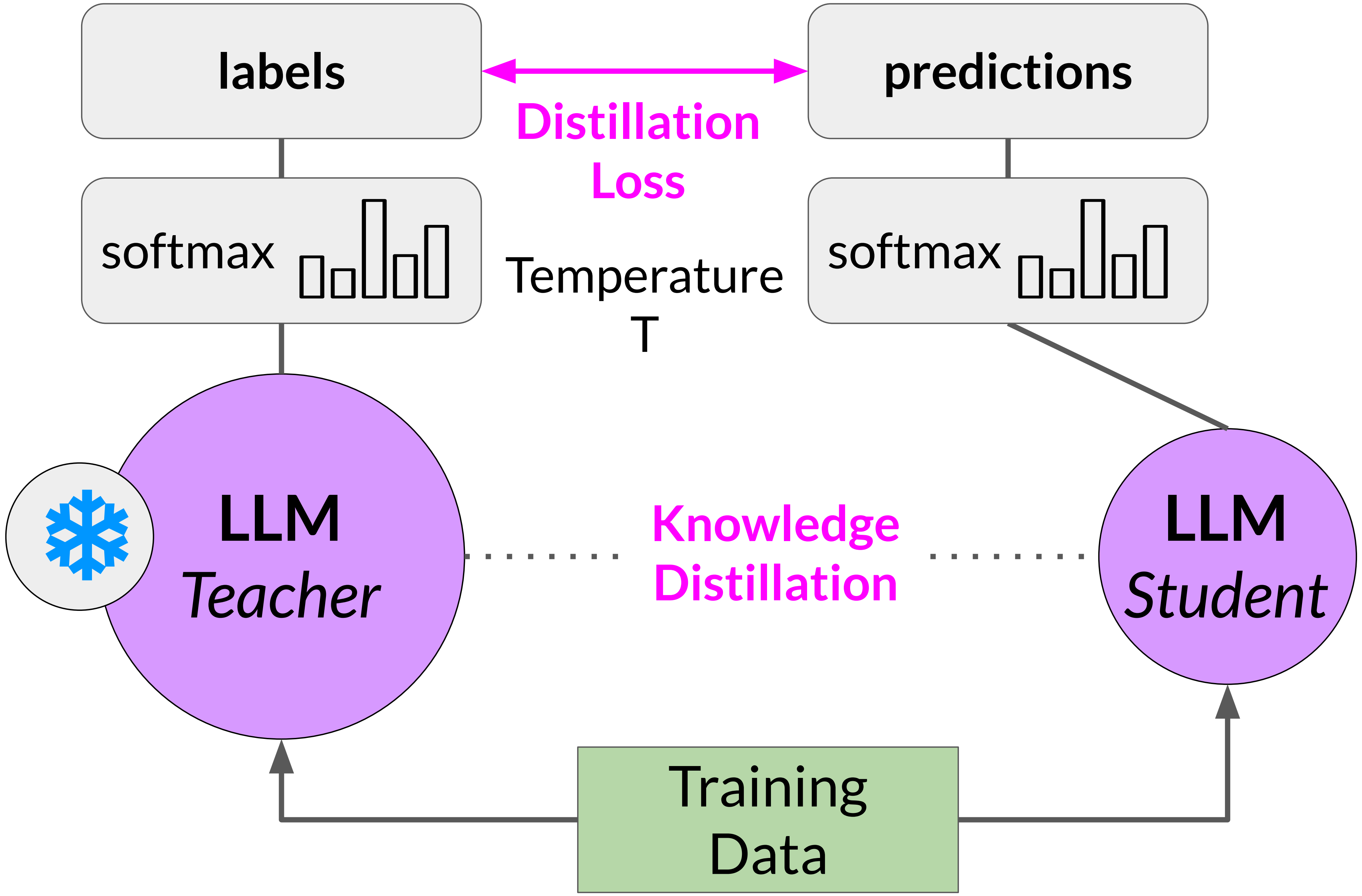
Distillation

Train a smaller student model from a larger teacher model



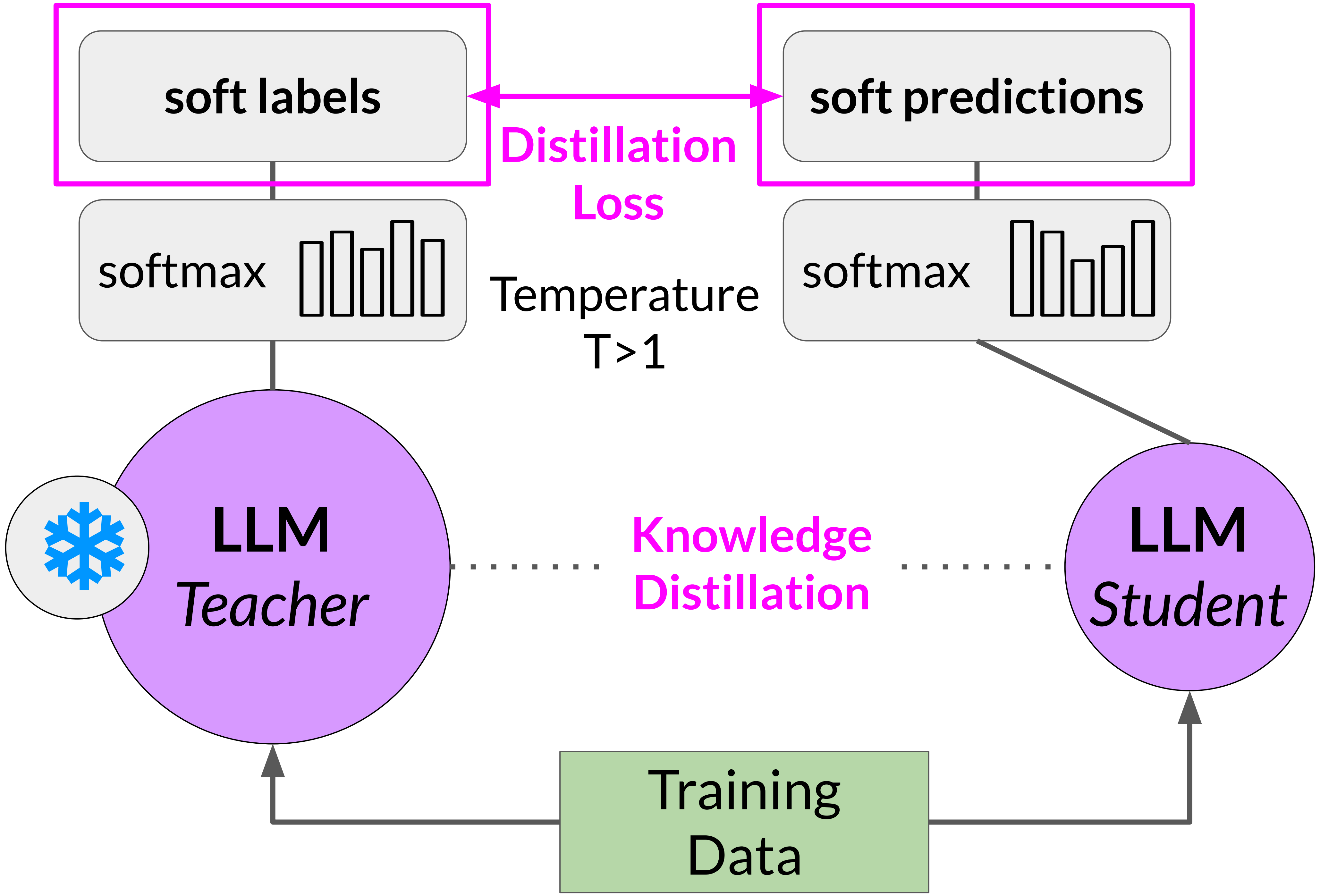
Distillation

Train a smaller student model from a larger teacher model



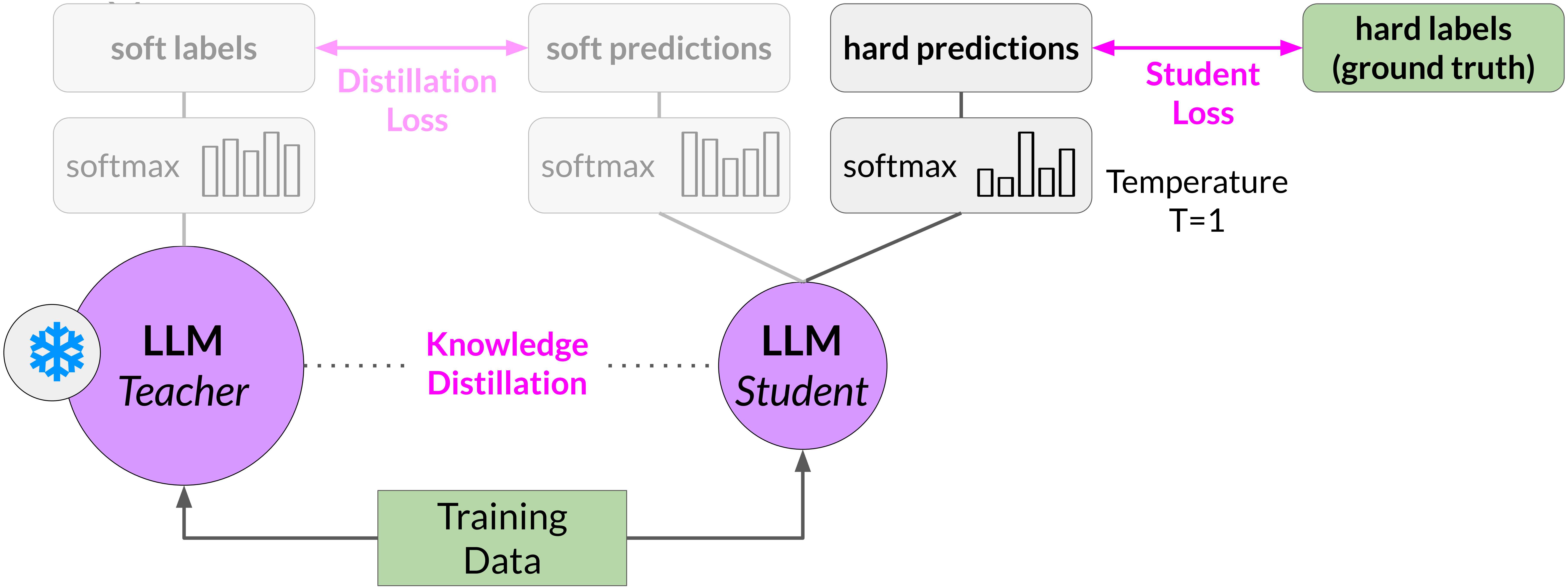
Distillation

Train a smaller student model from a larger teacher model



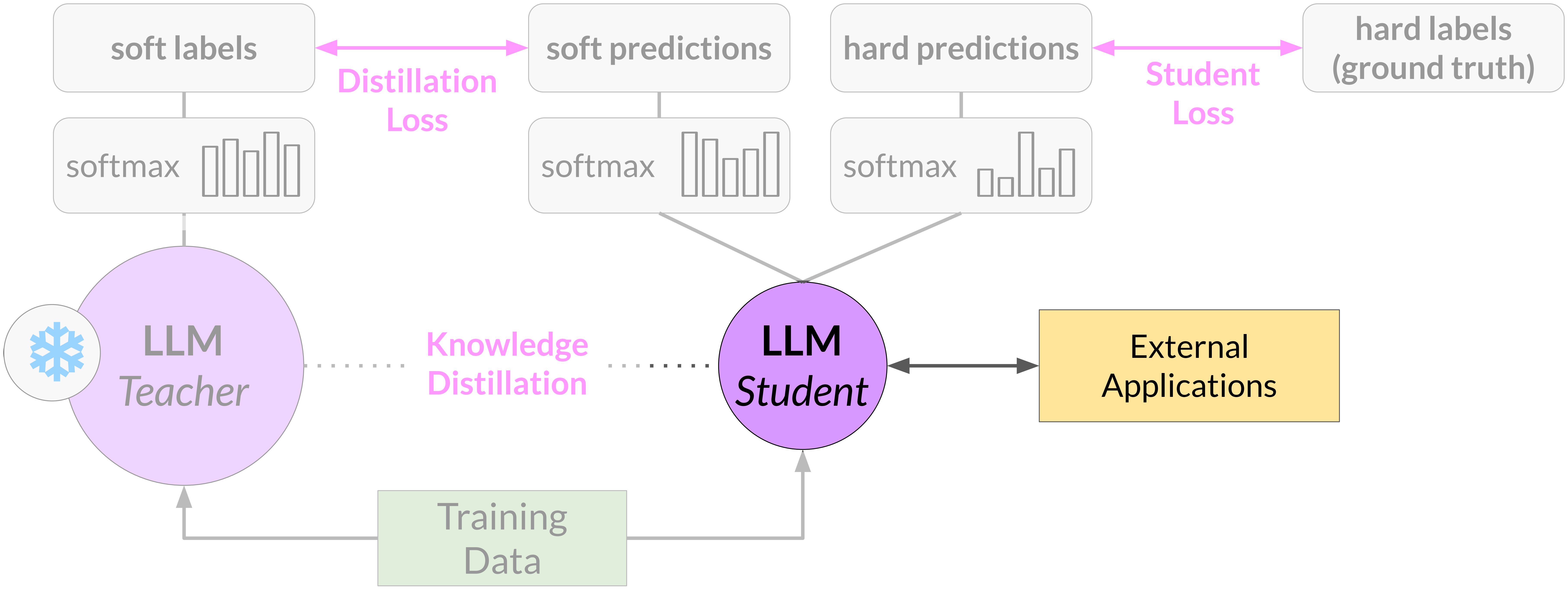
Distillation

Train a smaller student model from a larger teacher model



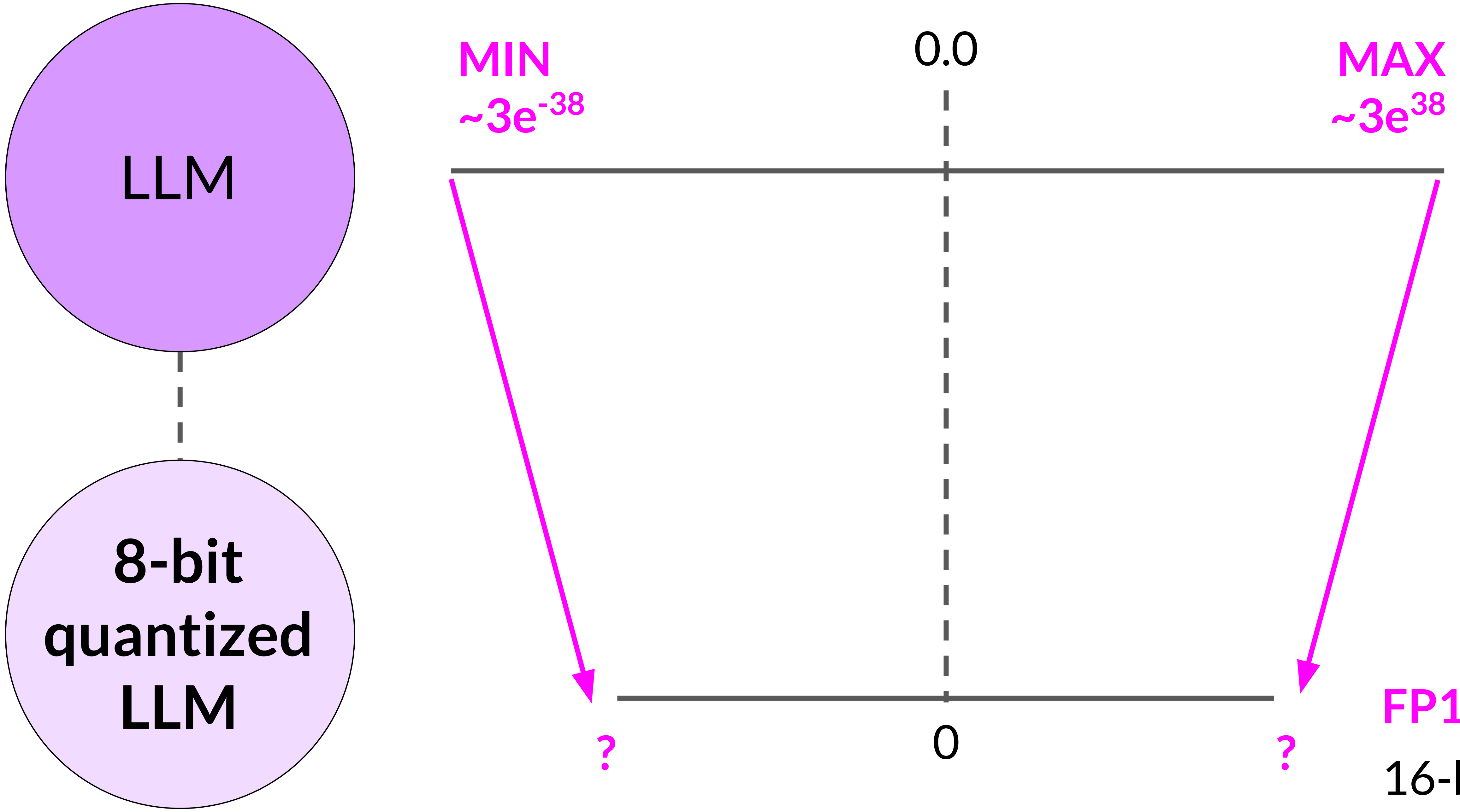
Distillation

Train a smaller student model from a larger teacher model



Post-Training Quantization (PTQ)

Reduce precision of model weights



FP32

32-bit floating point

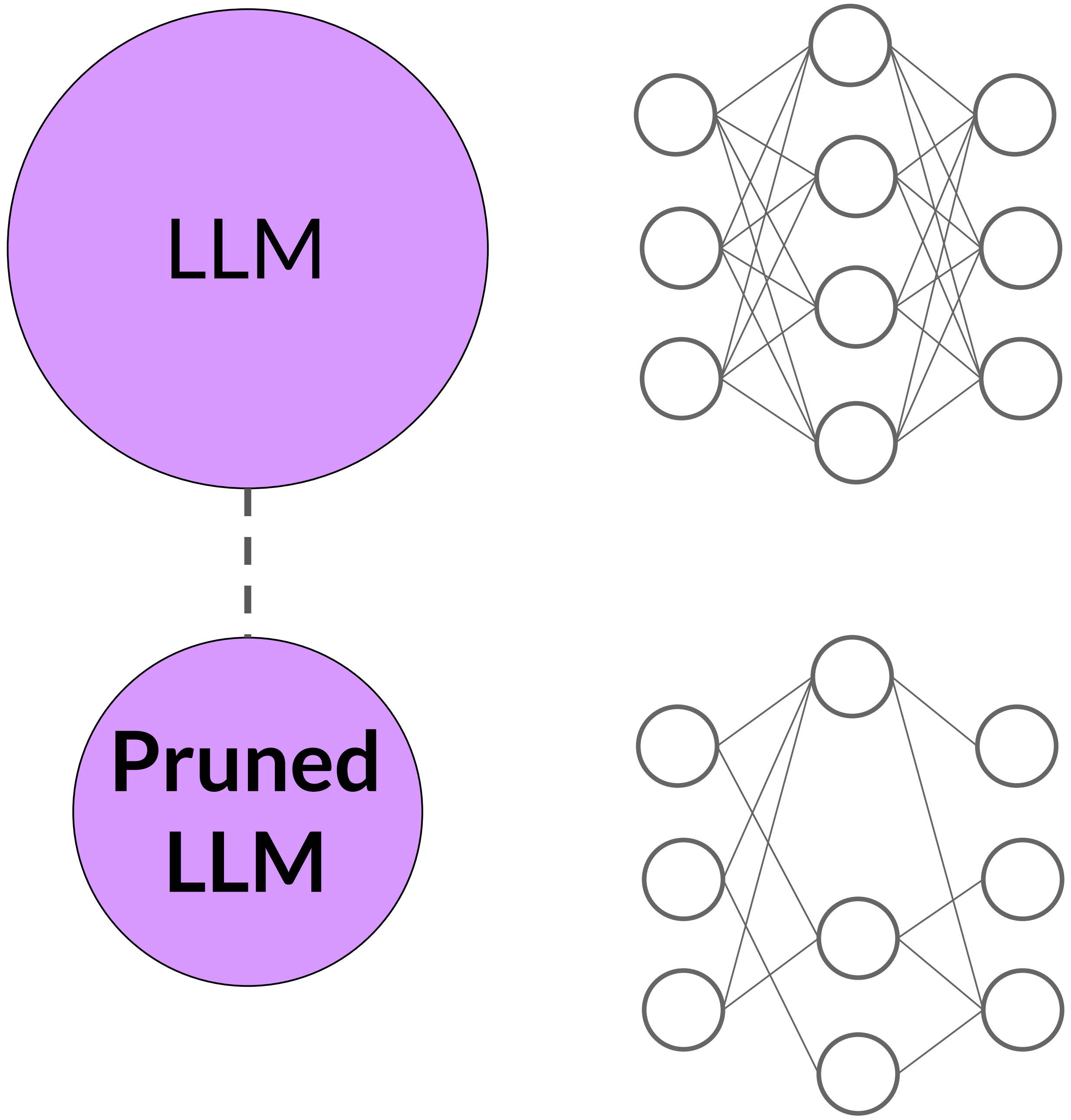
- **Applied to model weights** (and/or activations)
- **Requires calibration** to capture dynamic range

FP16 | BFLOAT16 | INT8

16-bit floating point | 8-bit integer

Pruning

Remove model weights with values close or equal to zero



- Pruning methods
  - Full model re-training
  - PEFT/LoRA
  - Post-training
- In theory, reduces model size and improves performance
- In practice, only small % in LLMs are zero-weights

Cheat Sheet - Time and effort in the lifecycle

	Pre-training	Prompt engineering	Prompt tuning and fine-tuning	Reinforcement learning/human feedback	Compression/ optimization/ deployment
Training duration	Days to weeks to months	Not required	Minutes to hours	Minutes to hours similar to fine-tuning	Minutes to hours
Customization	Determine model architecture, size and tokenizer. Choose vocabulary size and # of tokens for input/context Large amount of domain training data	No model weights Only prompt customization	Tune for specific tasks Add domain-specific data Update LLM model or adapter weights	Need separate reward model to align with human goals (helpful, honest, harmless) Update LLM model or adapter weights	Reduce model size through model pruning, weight quantization, distillation Smaller size, faster inference
Objective	Next-token prediction	Increase task performance	Increase task performance	Increase alignment with human preferences	Increase inference performance
Expertise	High	Low	Medium	Medium-High	Medium

Using the LLM in applications

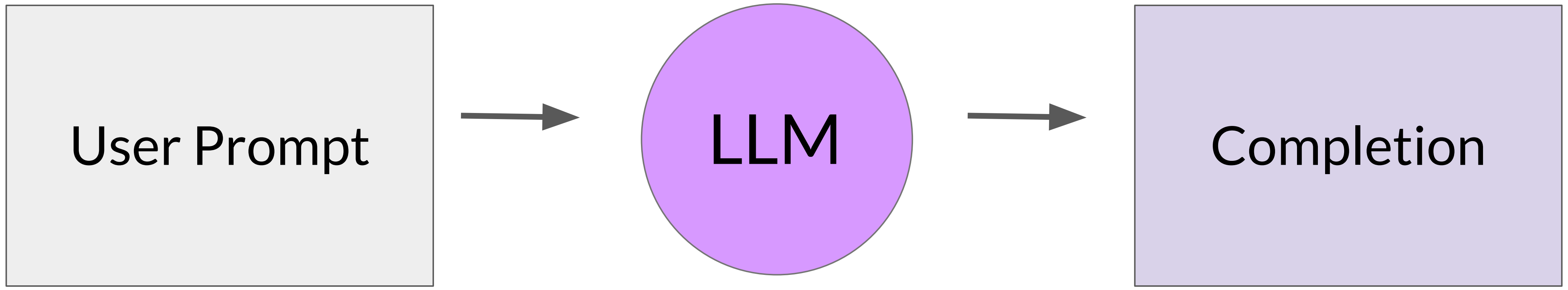
Models having difficulty

Prompt	Model	Completion	
`Who is the Prime Minister of the UK?`	LLM	`Who is the Prime Minister of the UK?` **`Boris Johnson`**	**Out of date**
`What is 40366 / 439?`	LLM	`What is 40366 / 439?` **`92.549`**	**Wrong (91.949)**
`What is a Martian Dunetree?`	LLM	`What is a Martian Dunetree?` **`A Martian Dunetree is a type of extraterrestrial plant found on Mars.`**	**Hallucination**

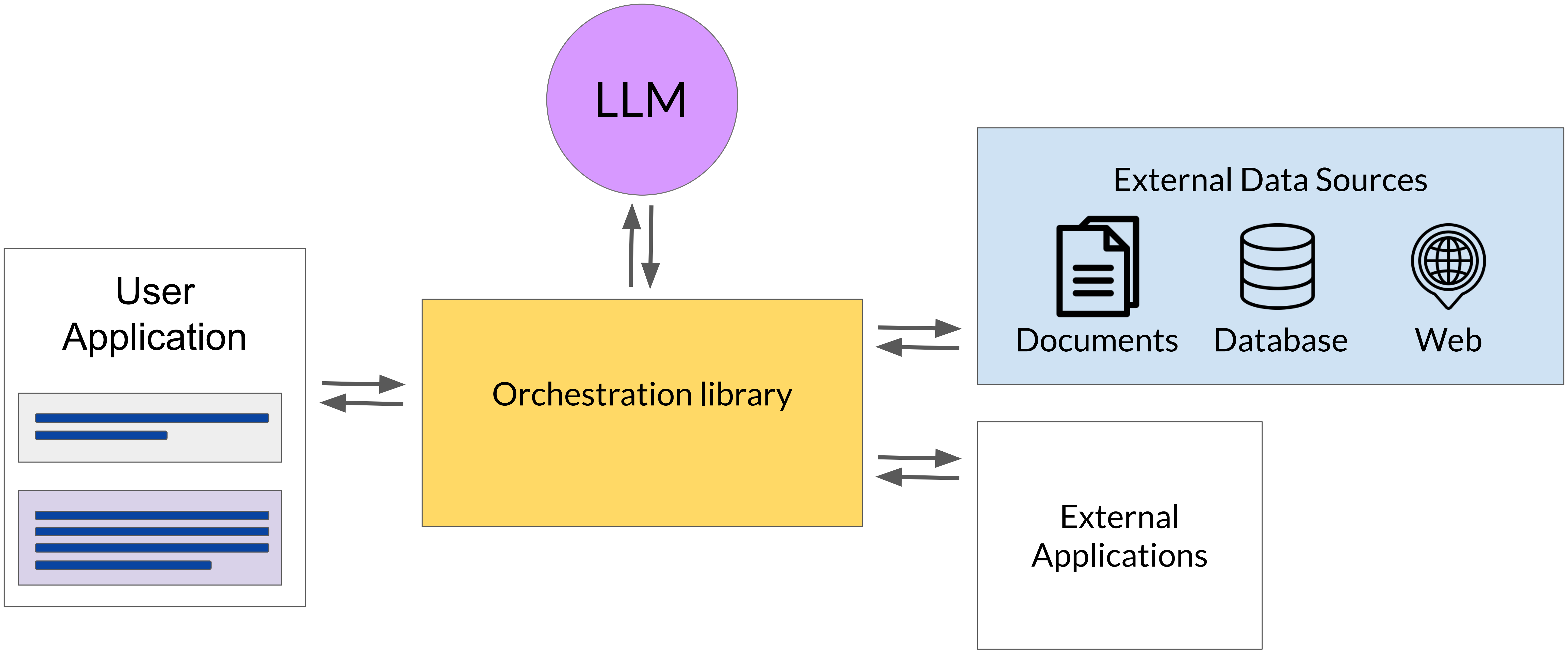
Generative AI project lifecycle

Scope	Select	Adapt and align model		Application integration	
Define the problem	Choose model	Prompt engineering Fine-tuning Align with human feedback	Evaluate	Optimize and deploy model for inference	Augment model and build LLM-powered applications

LLM-powered applications



LLM-powered applications



Retrieval augmented generation (RAG)

Knowledge cut-offs in LLMs

Prompt

```
Who is the
current Prime
Minister of the
United Kingdom?
```

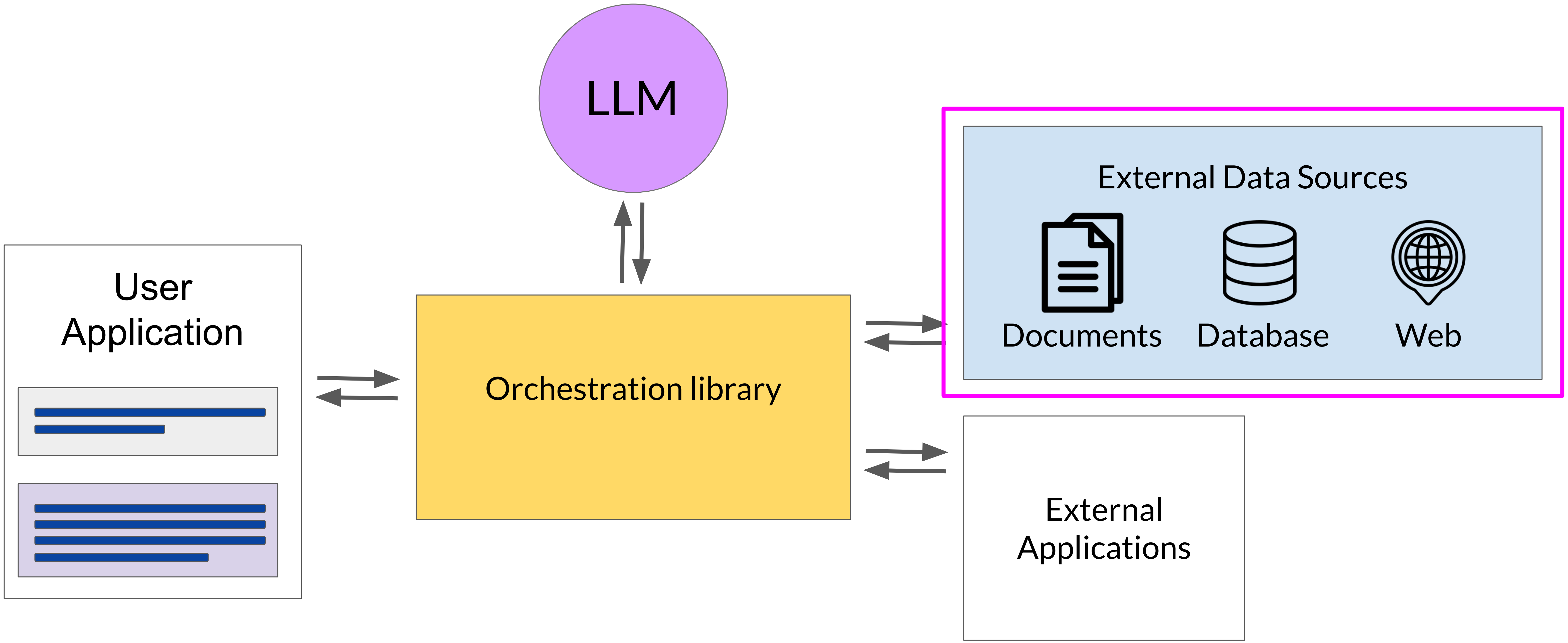
Model

LLM

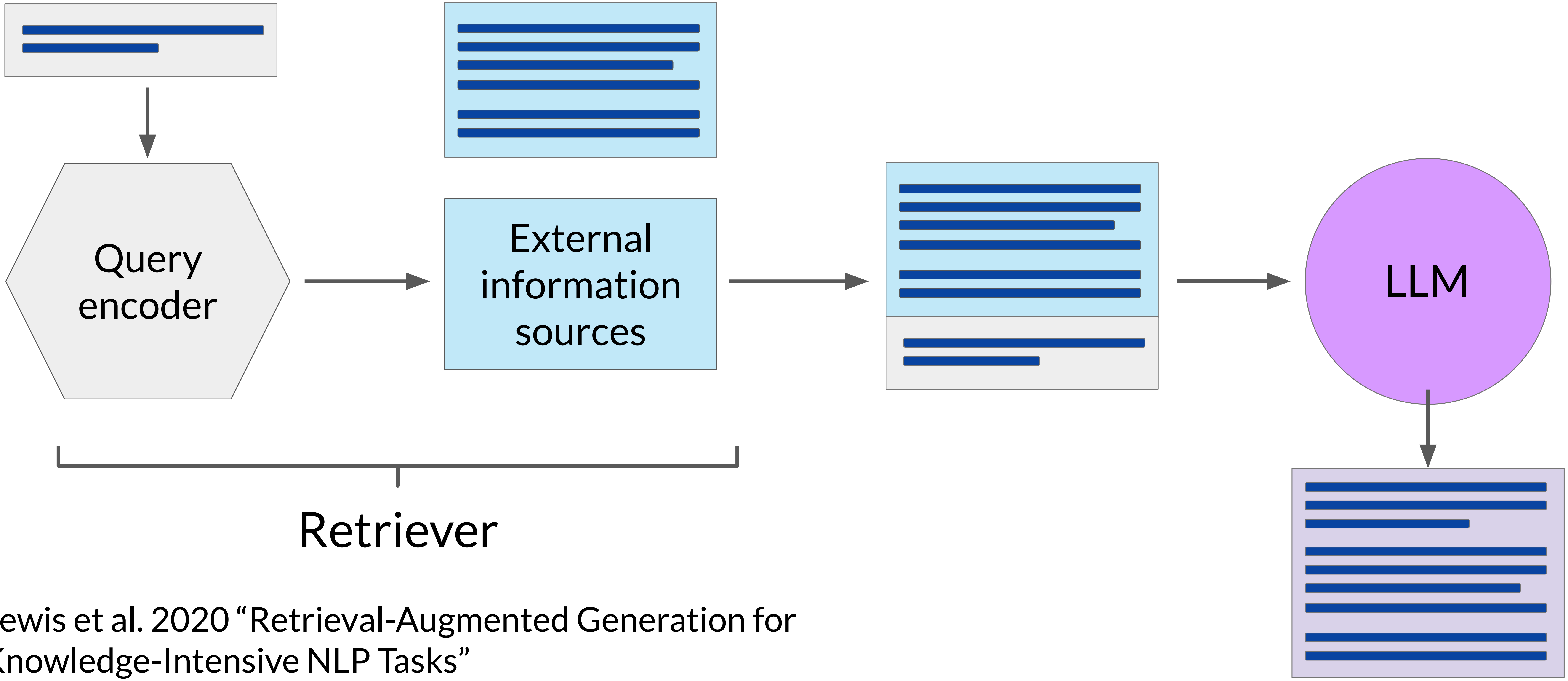
Completion

```
Who is the
current Prime
Minister of the
United Kingdom?

Boris Johnson
```

LLM-powered applications

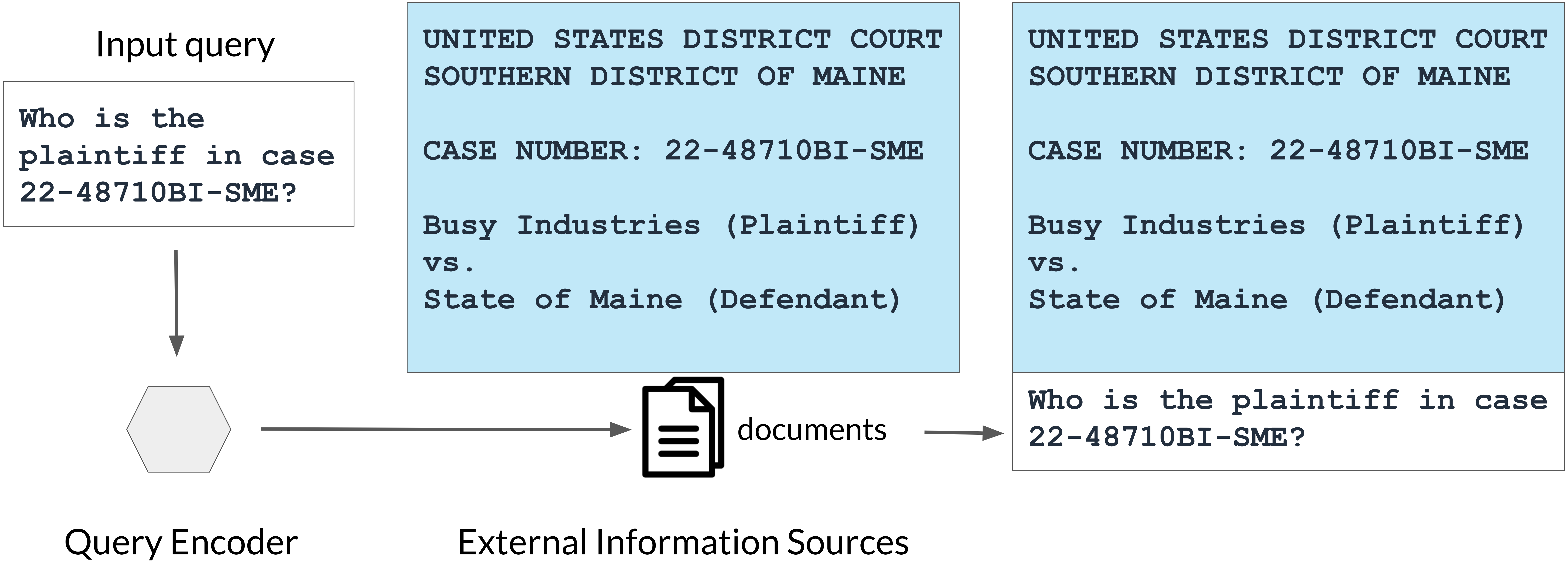


Retrieval Augmented Generation (RAG)

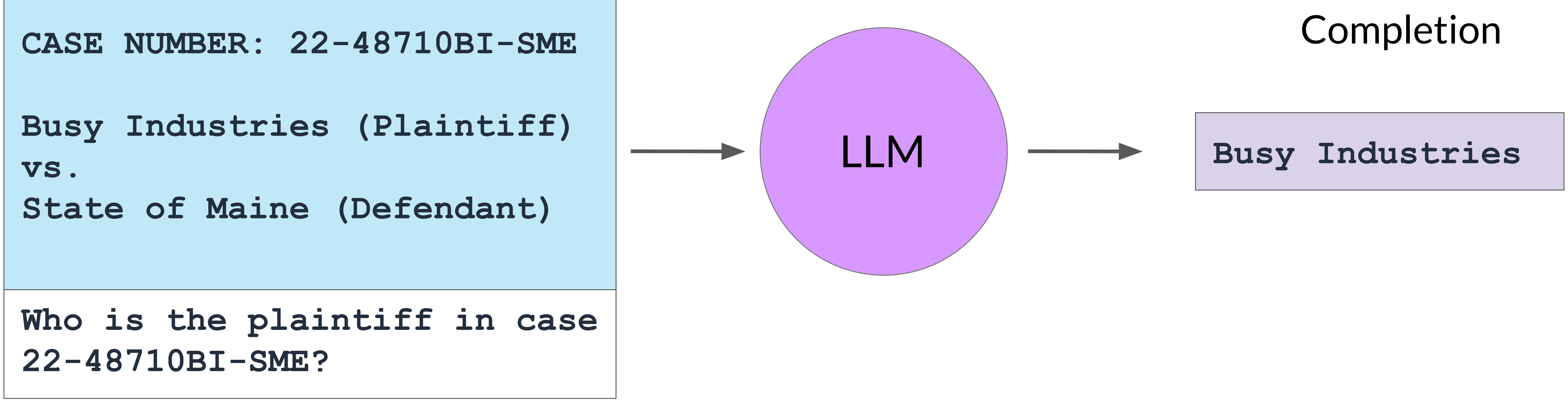


Lewis et al. 2020 "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks"

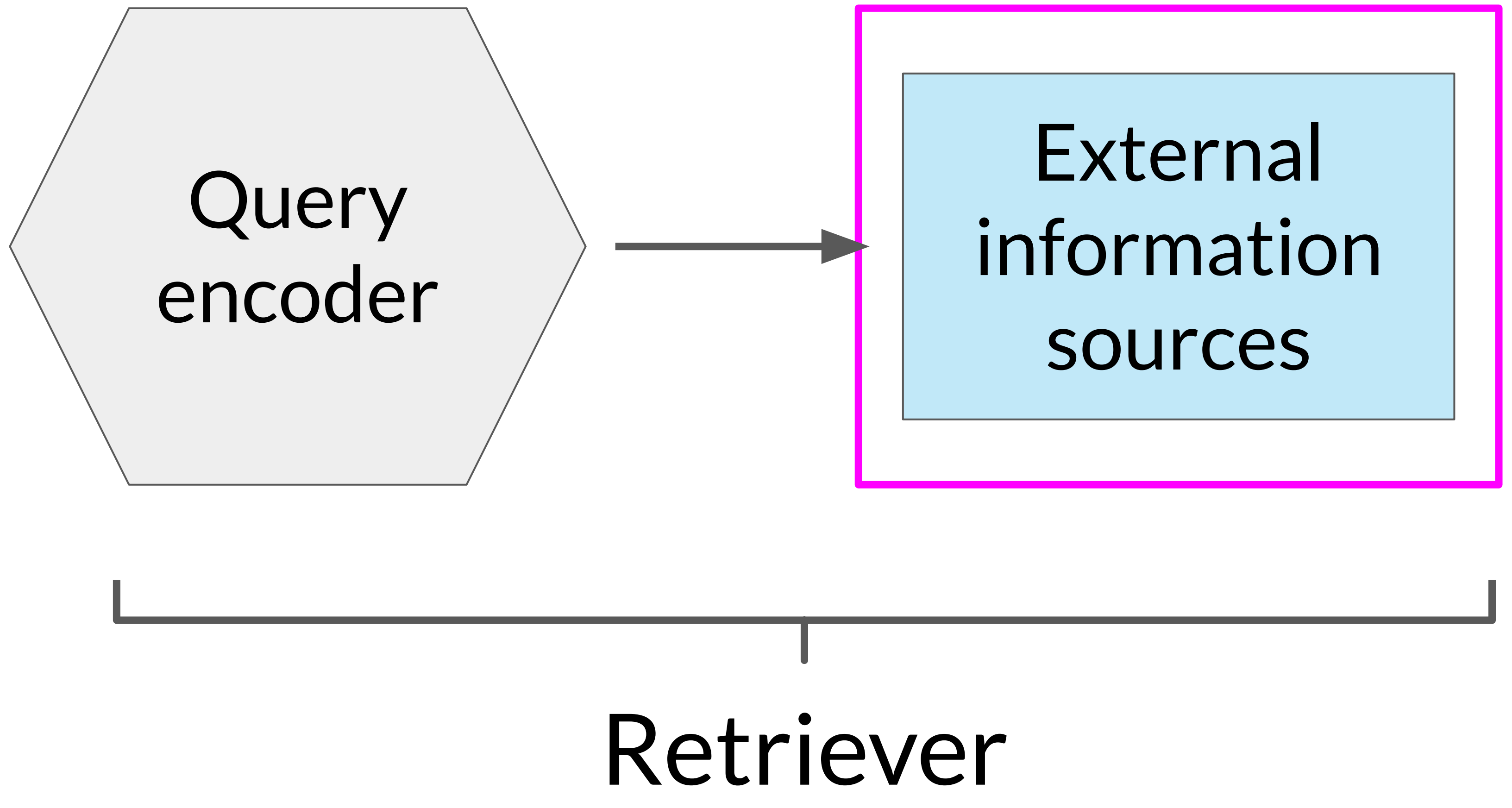
Example: Searching legal documents



Example: Searching legal documents



RAG integrates with many types of data sources



External Information Sources

- Documents
- Wikis
- Expert Systems
- Web pages
- Databases
- Vector Store

Data preparation for vector store for RAG

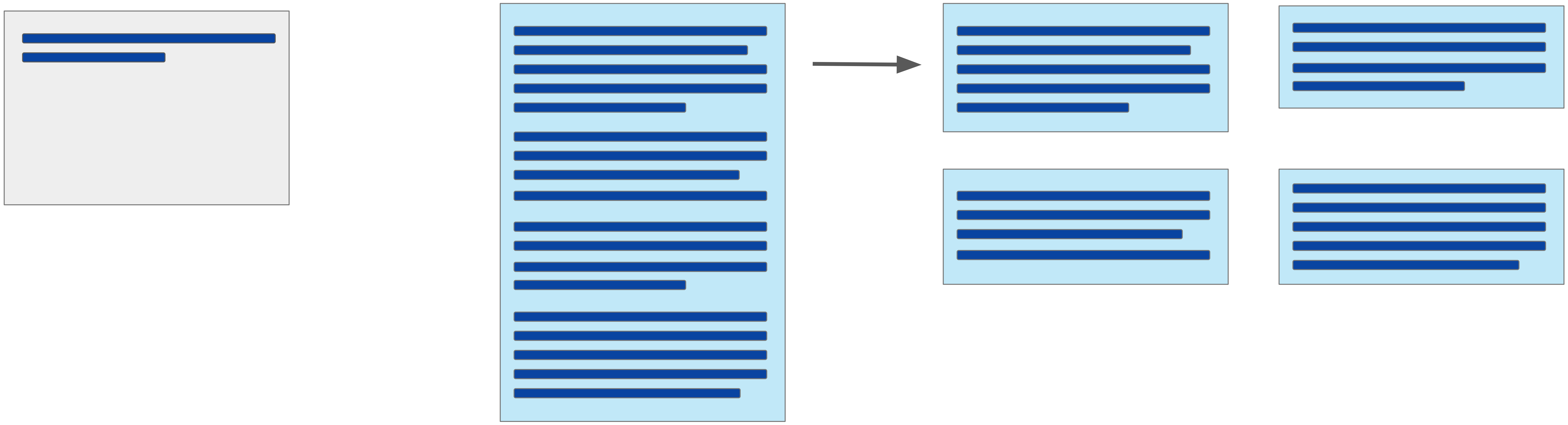
Two considerations for using external data in RAG:

1. Data must fit inside context window

Prompt context limit few 1000 tokens

Single document too large to fit in window

Split long sources into short chunks

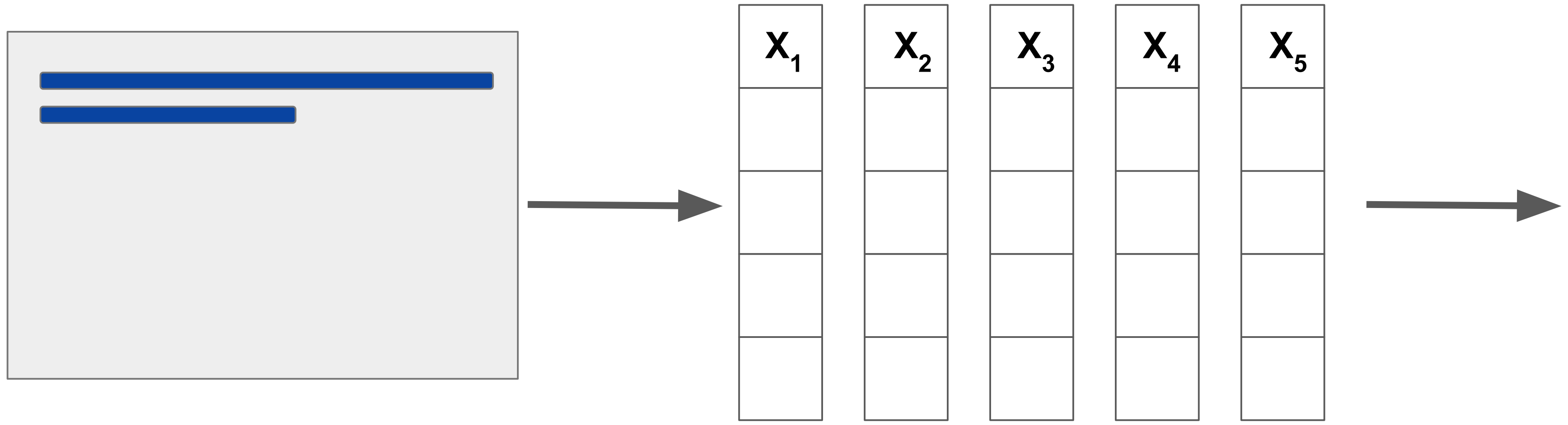


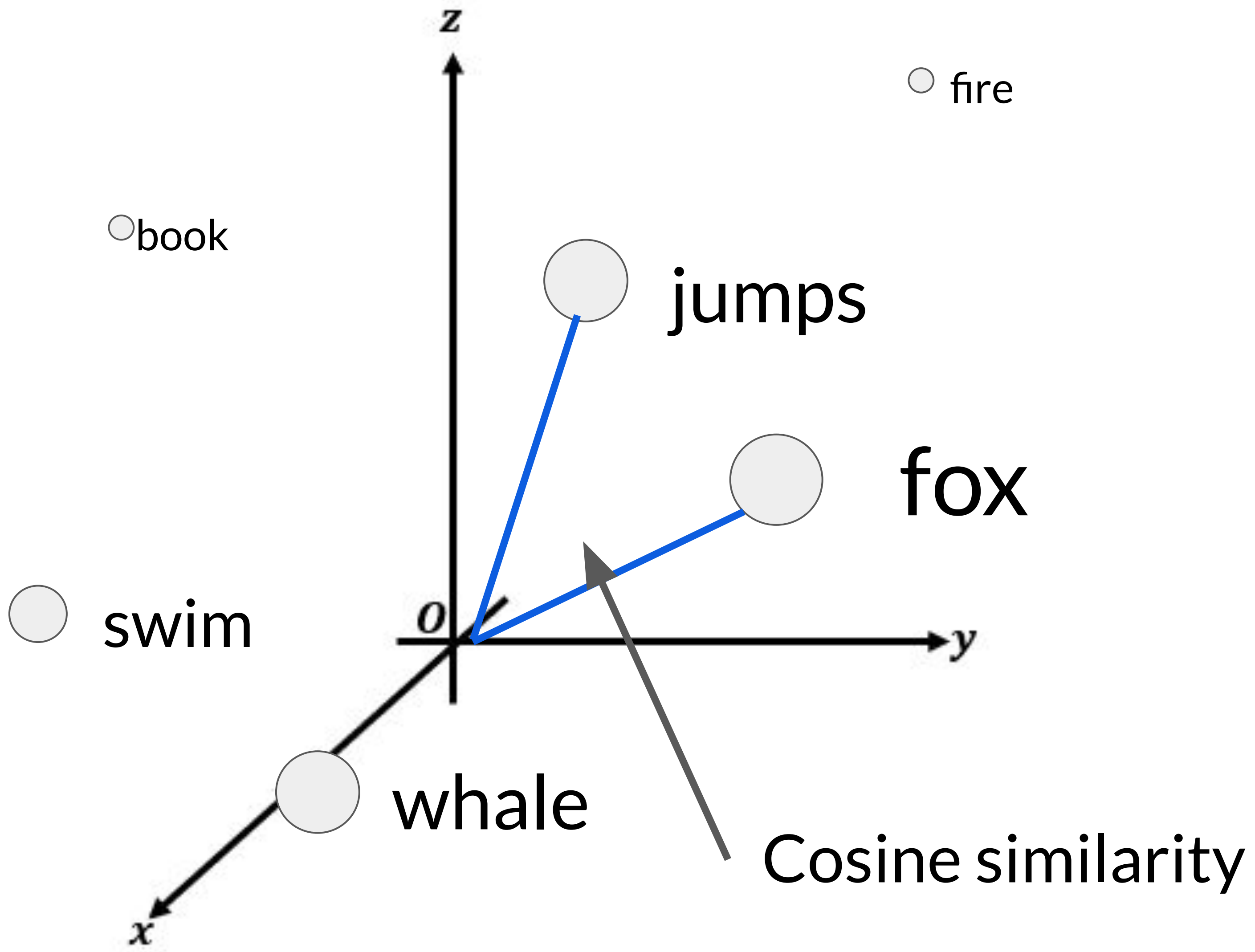
Data preparation for RAG

Two considerations for using external data in RAG:

1. Data must fit inside context window
2. Data must be in format that allows its relevance to be assessed at inference time: **Embedding vectors**

Prompt text converted to embedding vectors



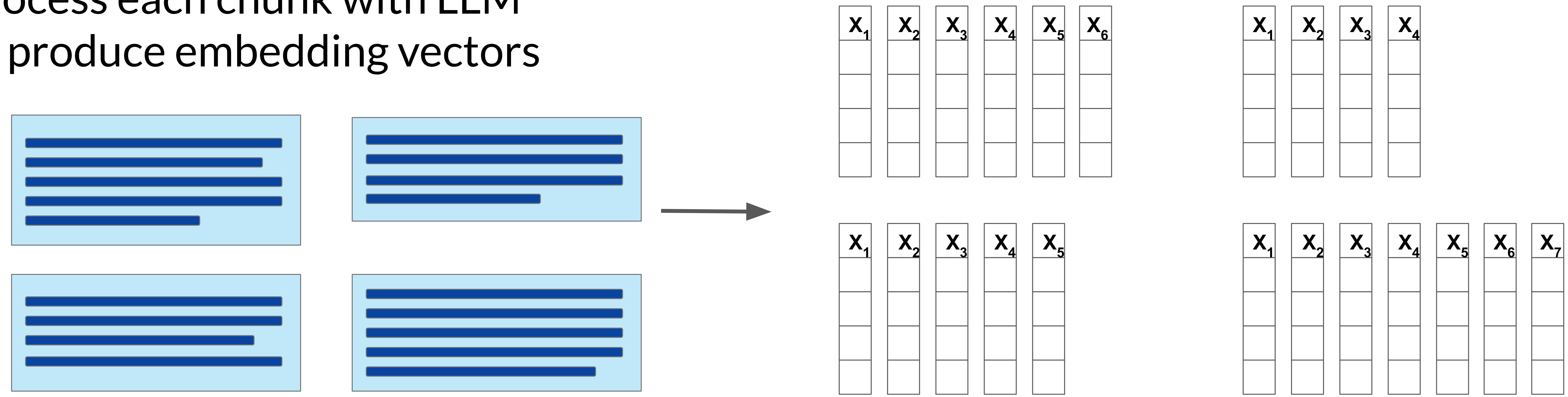


Data preparation for RAG

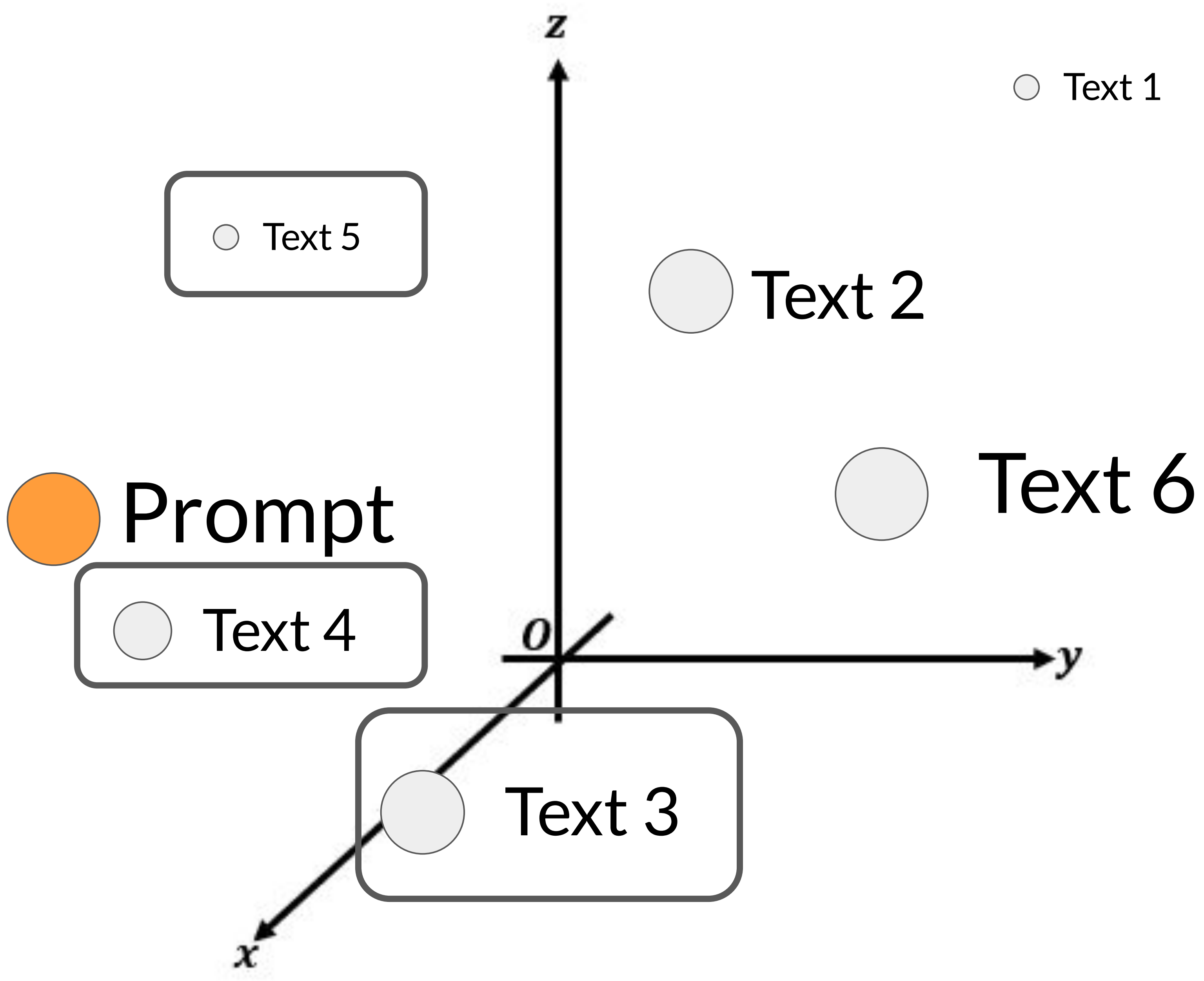
Two considerations for using external data in RAG:

1. Data must fit inside context window
2. Data must be in format that allows its relevance to be assessed at inference time: **Embedding vectors**

Process each chunk with LLM
to produce embedding vectors



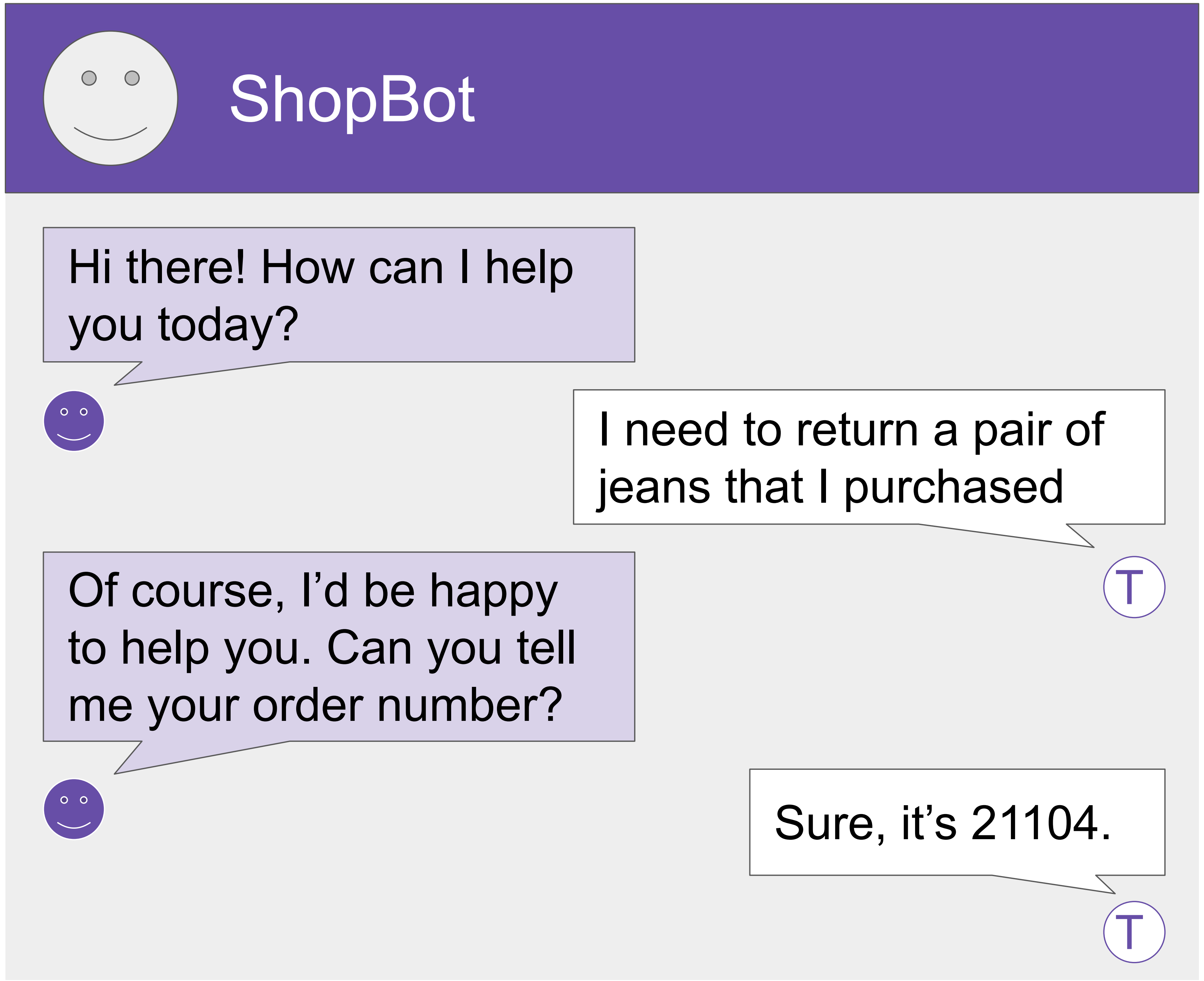
Vector database search



- Each text in vector store is identified by a key
- Enables a **citation** to be included in completion

Enabling interactions with external applications

Having an LLM initiate a clothing return



Having an LLM initiate a clothing return

ShopBot

Lookup with RAG

Ok, I've found your order. Do you want to return any other items from that order?

No, only the jeans.

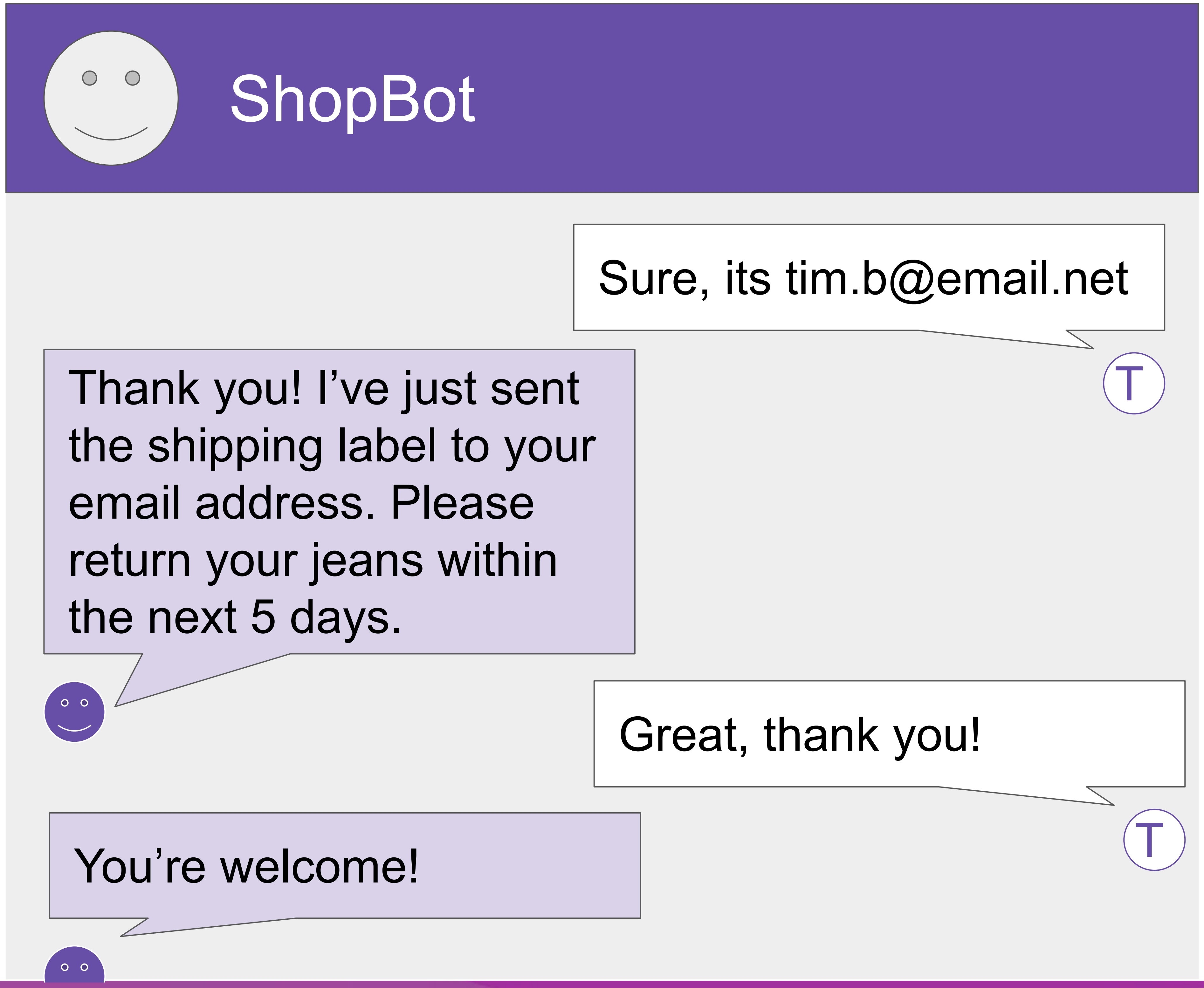
T

API call

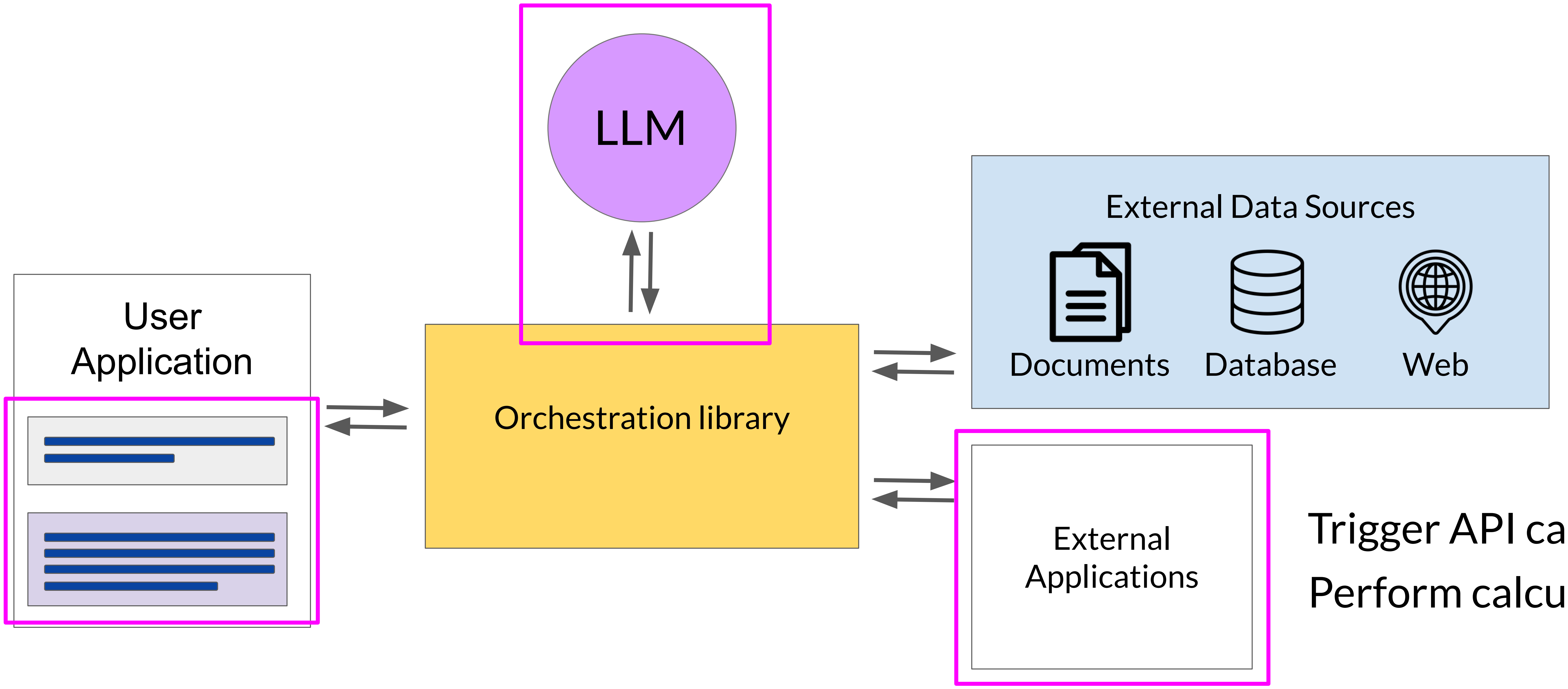
Ok great. Let me get a return label from our shipping partner. Can you remind me of the email you used to order?

Having an LLM initiate a clothing return

API call to the shipper



LLM-powered applications



Trigger API call

Perform calculations

Requirements for using LLMs to power applications

Plan actions

Steps to process return:

Step 1: Check order ID

Step 2: Request label

Step 3: Verify user email

Step 4: Email user label

Format outputs

SQL Query:

SELECT COUNT(*)
FROM orders
WHERE order_id = 21104

Validate actions

Collect required user information and make sure it is in the completion

User email:
tim.b@email.net

Prompt structure is important!

Helping LLMs reason and plan with Chain-of-Thought Prompting

LLMs can struggle with complex reasoning problems

Prompt

```
Q: Roger has 5 tennis balls.
He buys 2 more cans of tennis
balls. Each can has 3 tennis
balls. How many tennis balls
does he have now?

A: The answer is 11

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

Model

LLM

Completion

```
Q: Roger has 5 tennis balls.
He buys 2 more cans of tennis
balls. Each can has 3 tennis
balls. How many tennis balls
does he have now?

A: The answer is 11

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

A: The answer is 27.
```

Humans take a step-by-step approach to solving complex problems

Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

Start: Roger started with 5 balls.

Step 1: 2 cans of 3 tennis balls each is 6 tennis balls.

Step 2: 5 + 6 = 11

End: The answer is 11

Reasoning steps

"Chain of thought"

Chain-of-Thought Prompting can help LLMs reason

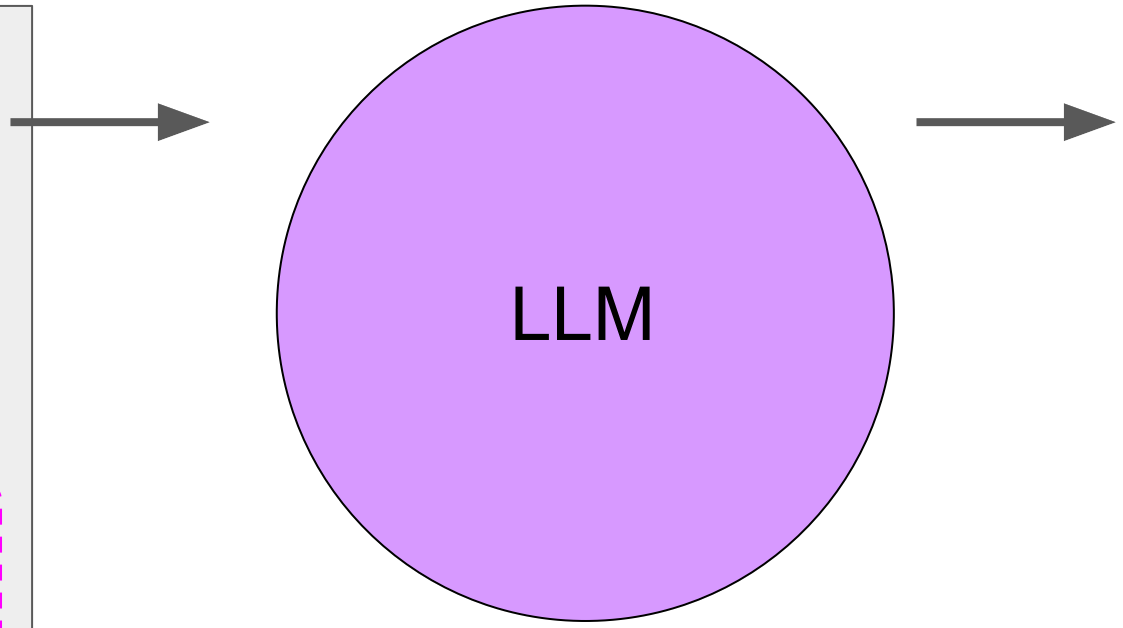
Prompt

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

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

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

Model



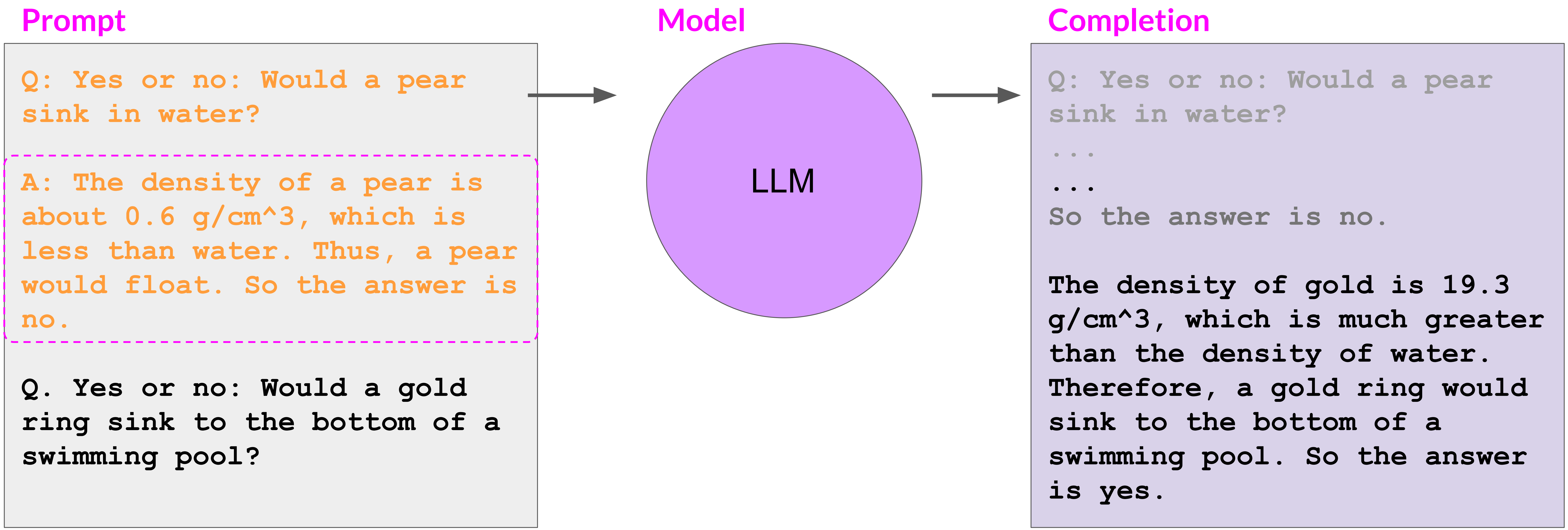
Completion

Q: Roger has 5 tennis balls.
...
...
...
how many apples do they have?

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

Source: Wei et al. 2022, “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models”

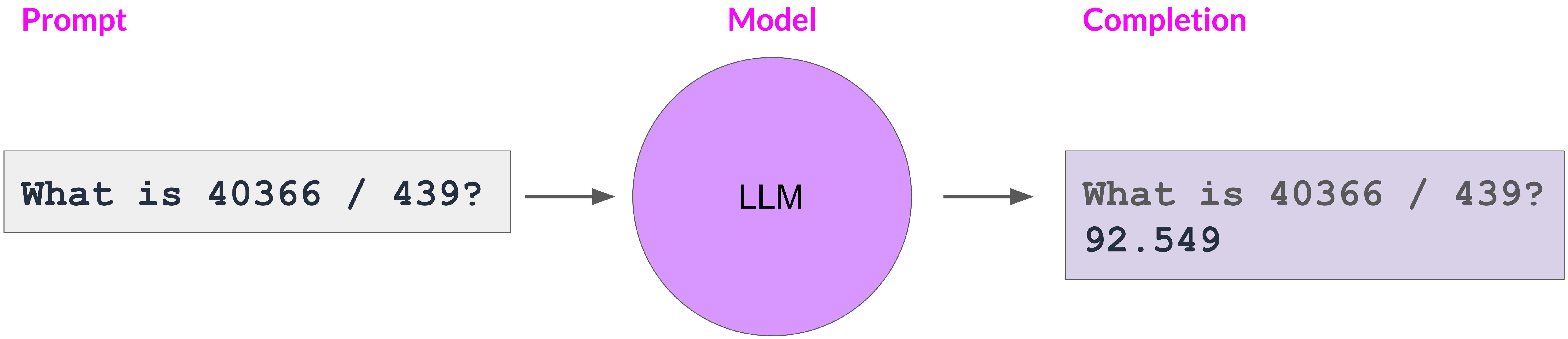
Chain-of-Thought Prompting can help LLMs reason



Source: Wei et al. 2022, "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models"

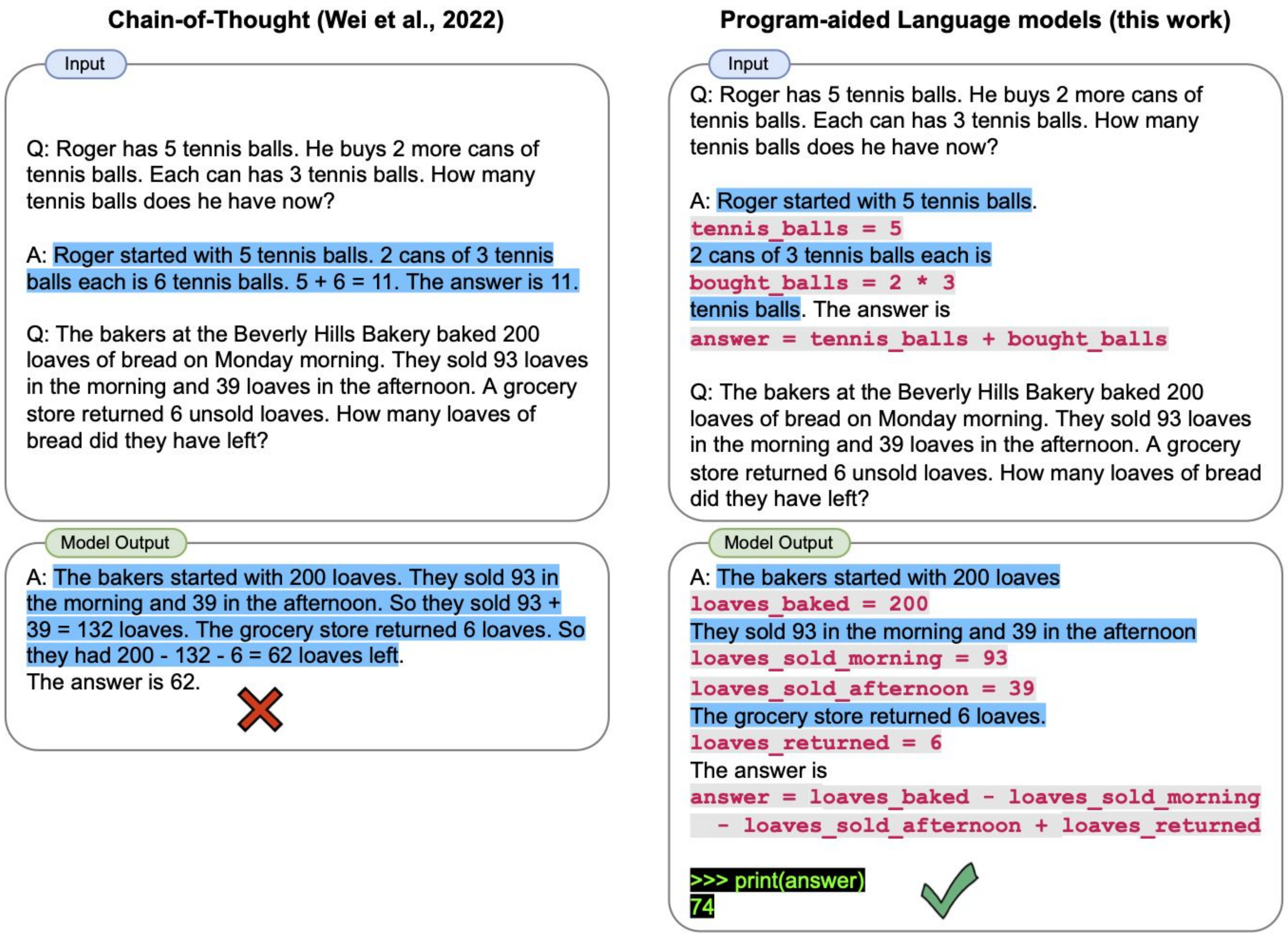
Program-aided Language Models

LLMs can struggle with mathematics



Program-aided language (PAL) models

LLM ✚ Code interpreter



Source: Gao et al. 2022, "PAL: Program-aided Language Models"

PAL example

Prompt with one-shot example

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

Answer:

```
# Roger started with 5 tennis balls
tennis_balls = 5
# 2 cans of tennis balls each is
bought_balls = 2 * 3
# tennis balls. The answer is
answer = tennis_balls + bought_balls
```

Q: The bakers at the Beverly Hills Bakery baked 200 loaves of bread on Monday morning. They sold 93 loaves in the morning and 39 loaves in the afternoon. A grocery store returned 6 unsold loaves. How many loaves did they have left?

PAL example

Prompt with one-shot example

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

Answer:

```
# Roger started with 5 tennis balls
tennis_balls = 5
# 2 cans of tennis balls each is
bought_balls = 2 * 3
# tennis balls. The answer is
answer = tennis_balls + bought_balls
```

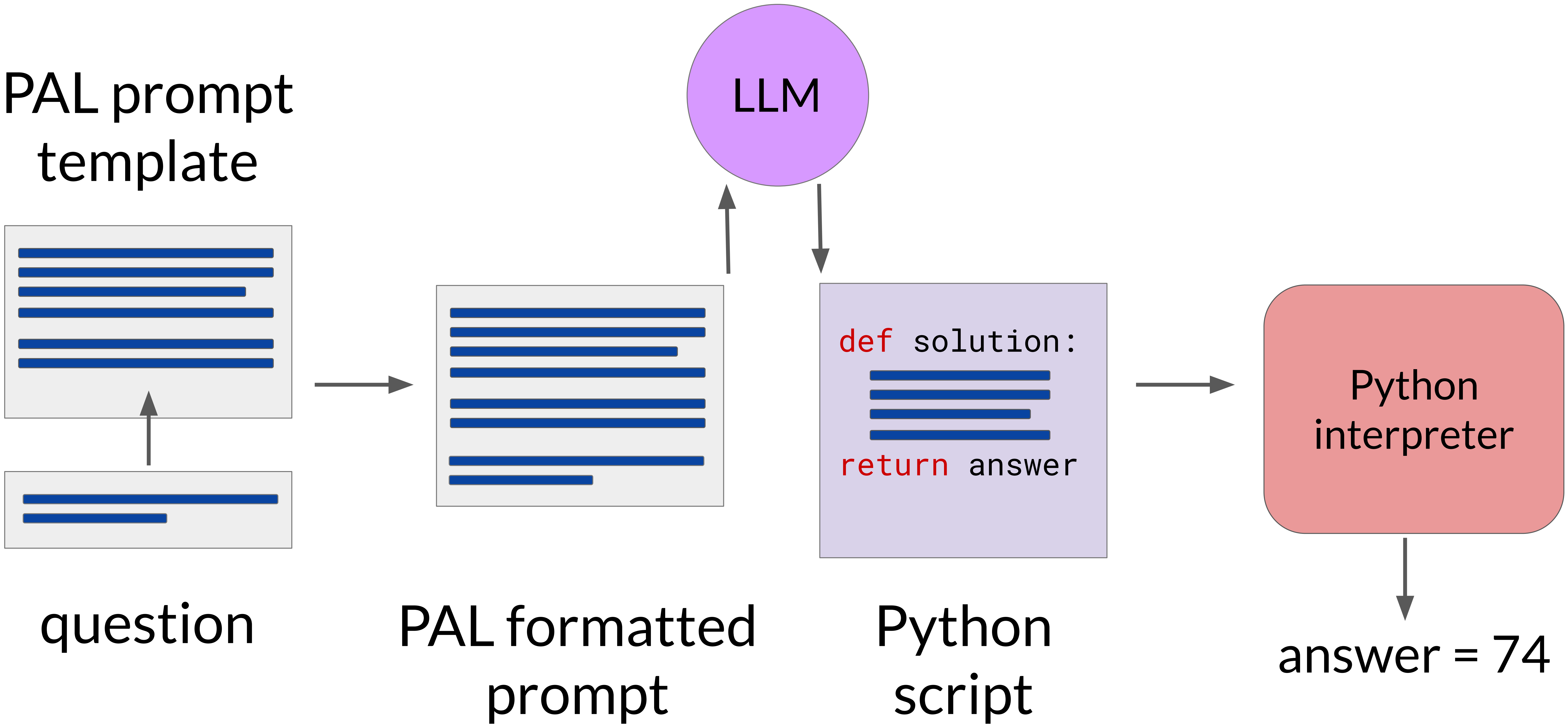
Q: The bakers at the Beverly Hills Bakery baked 200 loaves of bread on Monday morning. They sold 93 loaves in the morning and 39 loaves in the afternoon. A grocery store returned 6 unsold loaves. How many loaves did they have left?

Completion, CoT reasoning (blue) , and PAL execution (pink)

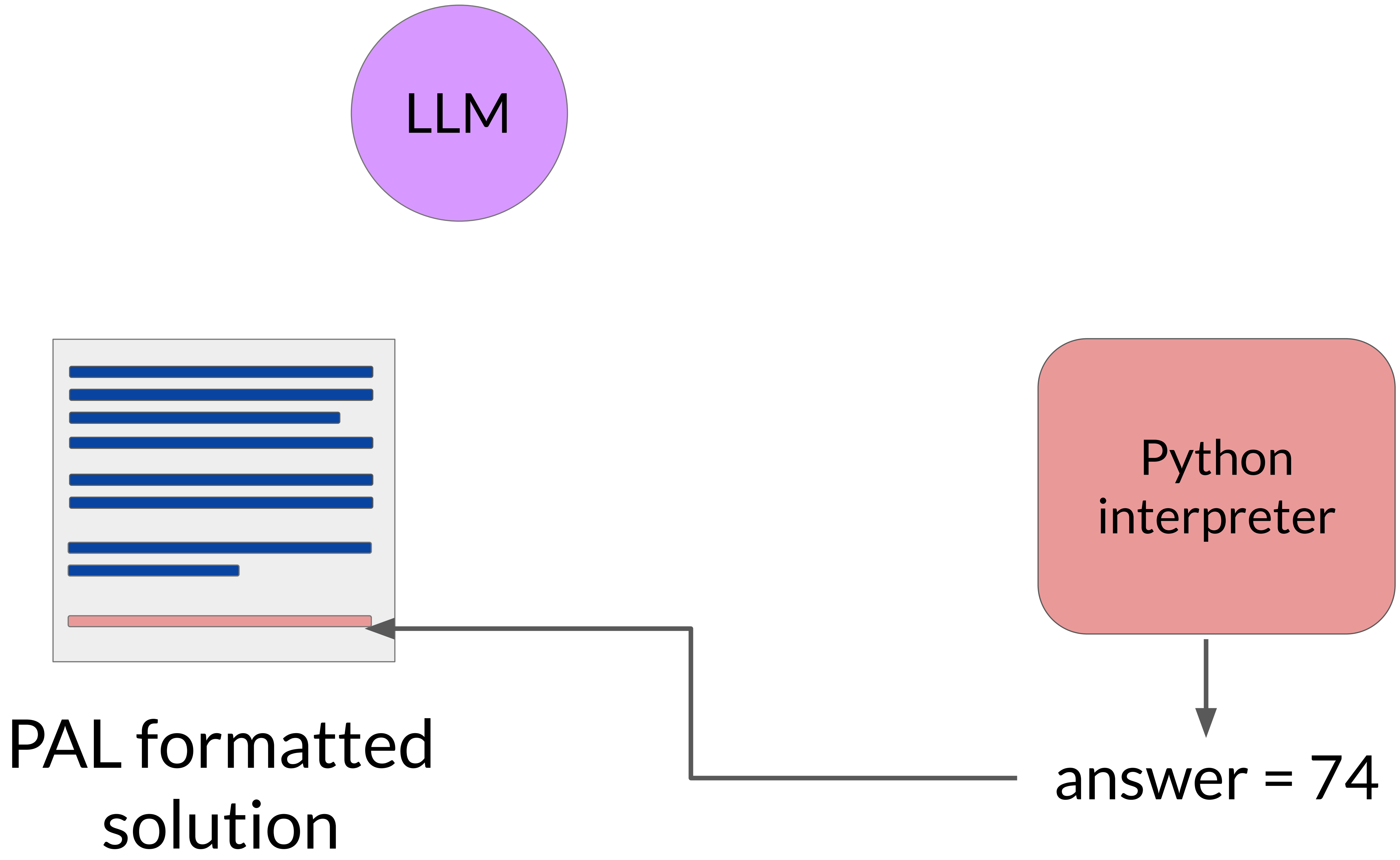
Answer:

```
# The bakers started with 200 loaves
loaves_baked = 200
# They sold 93 in the morning and 39 in the afternoon
loaves_sold_morning = 93
loaves_sold_afternoon = 39
# The grocery store returned 6 loaves.
loaves_returned = 6
# The answer is
answer = loaves_baked
  - loaves_sold_morning
  - loaves_sold_afternoon
  + loaves_returned
```

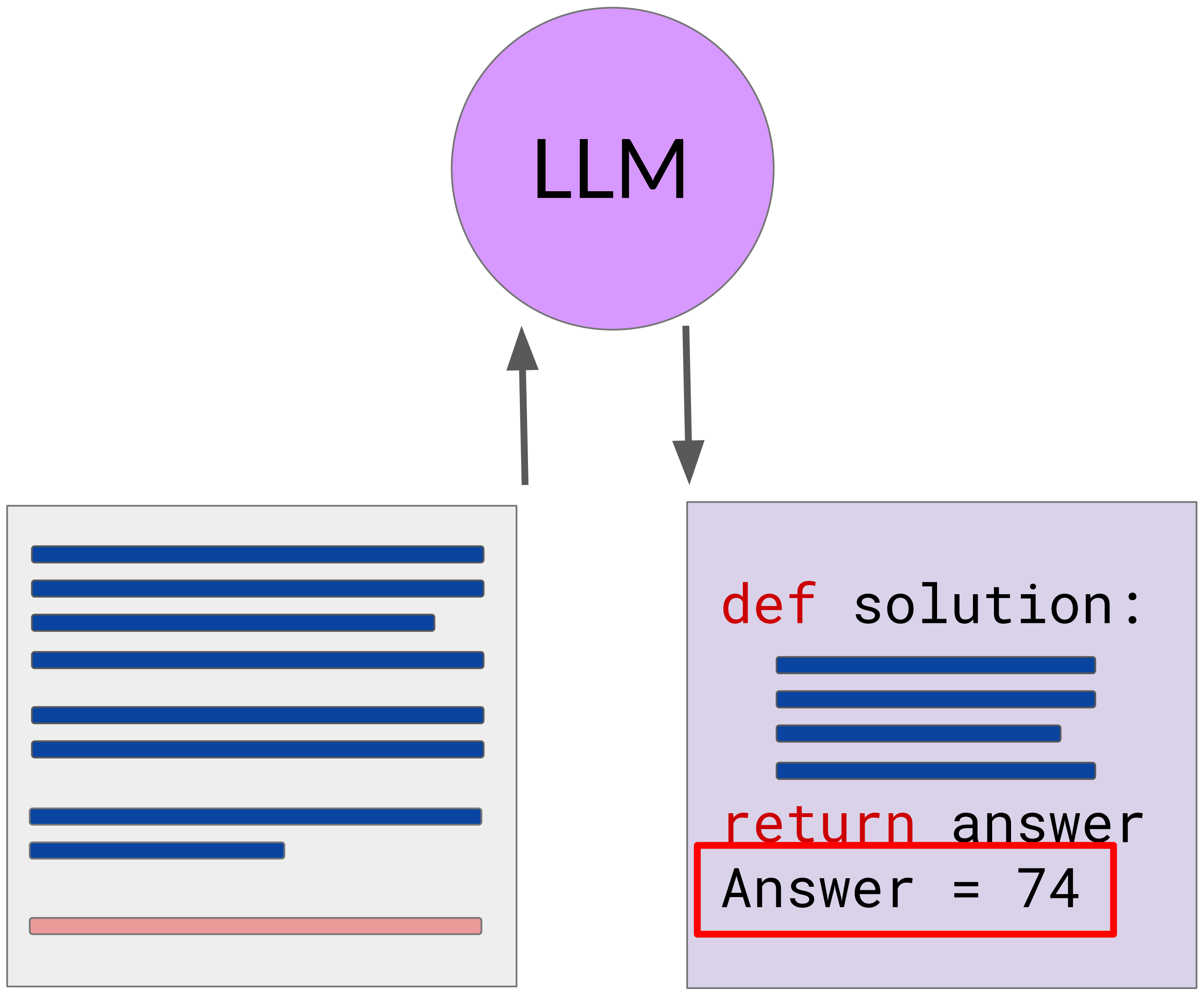
Program-aided language (PAL) models



Program-aided language (PAL) models



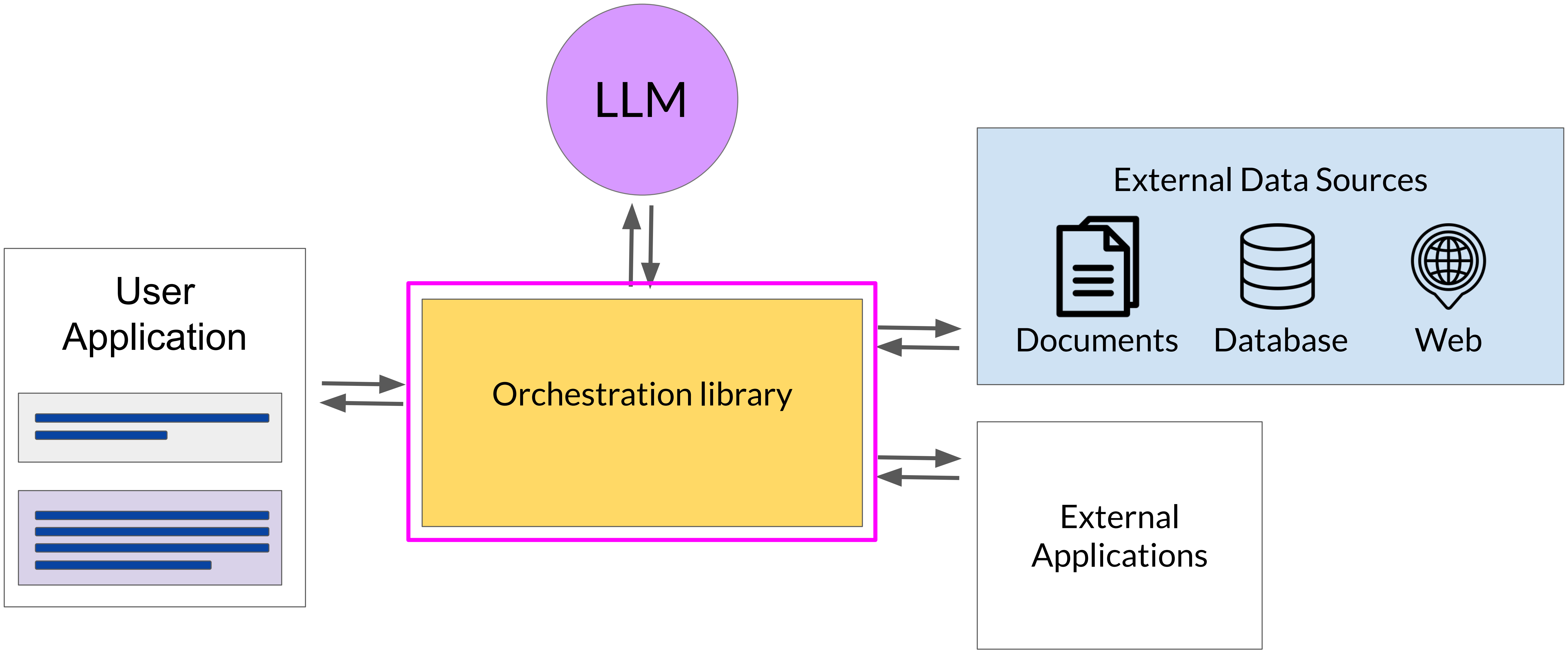
Program-aided language (PAL) models



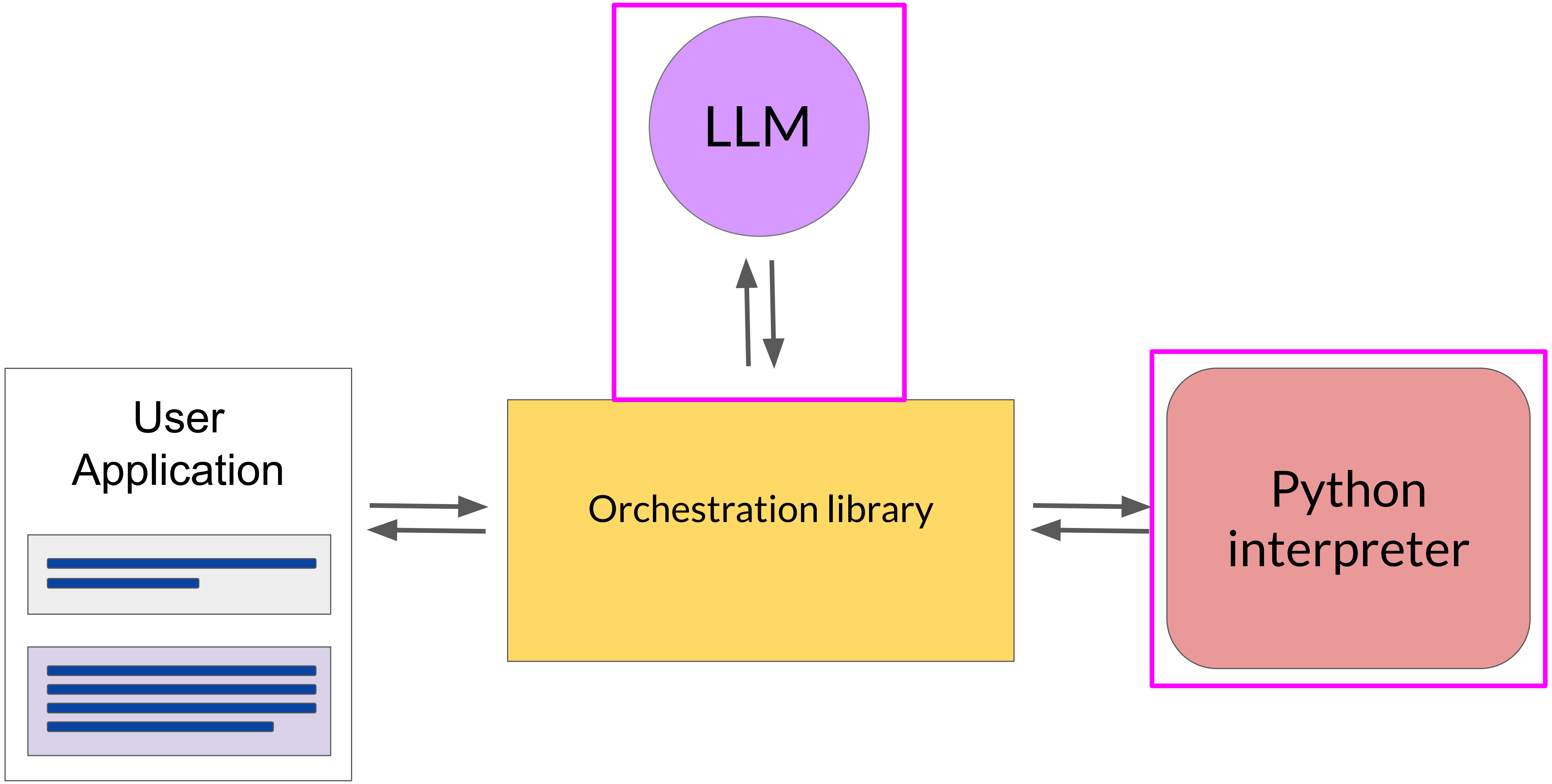
Completion with correct answer

PAL formatted solution

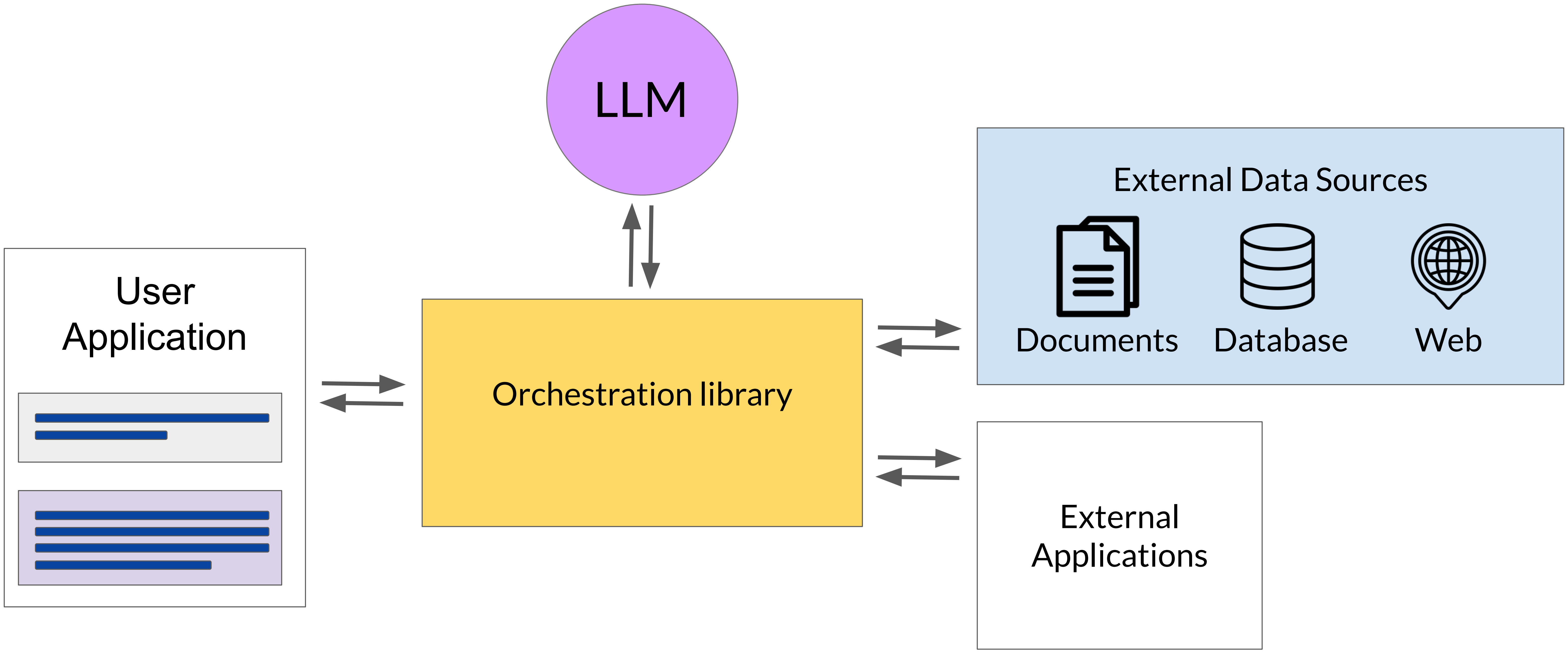
LLM-powered applications



PAL architecture

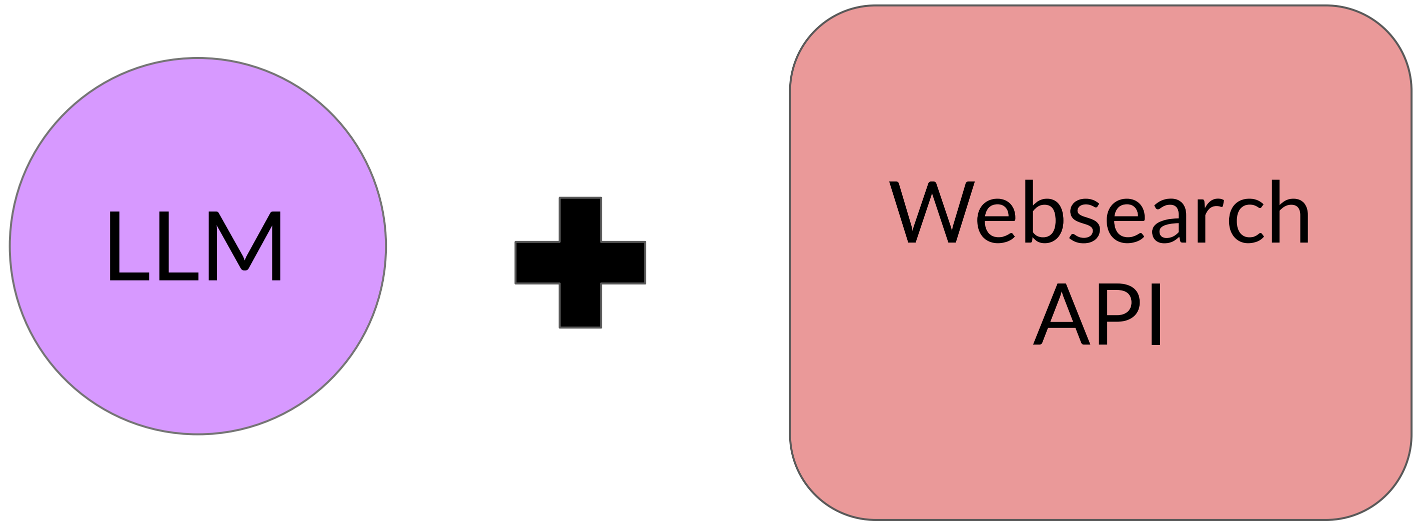


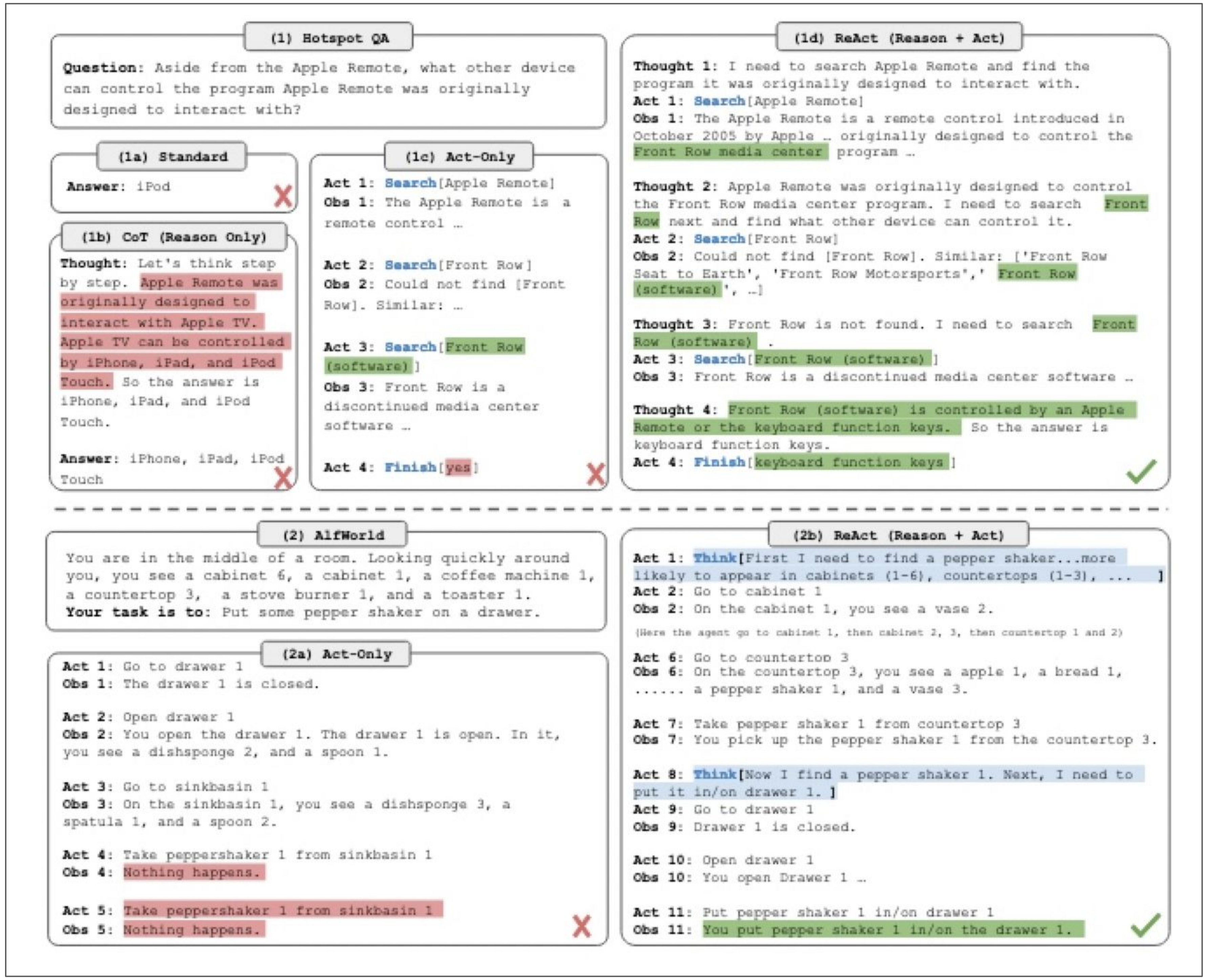
LLM-powered applications



ReAct: Combining reasoning and action in LLMs

ReAct: Synergizing Reasoning and Action in LLMs

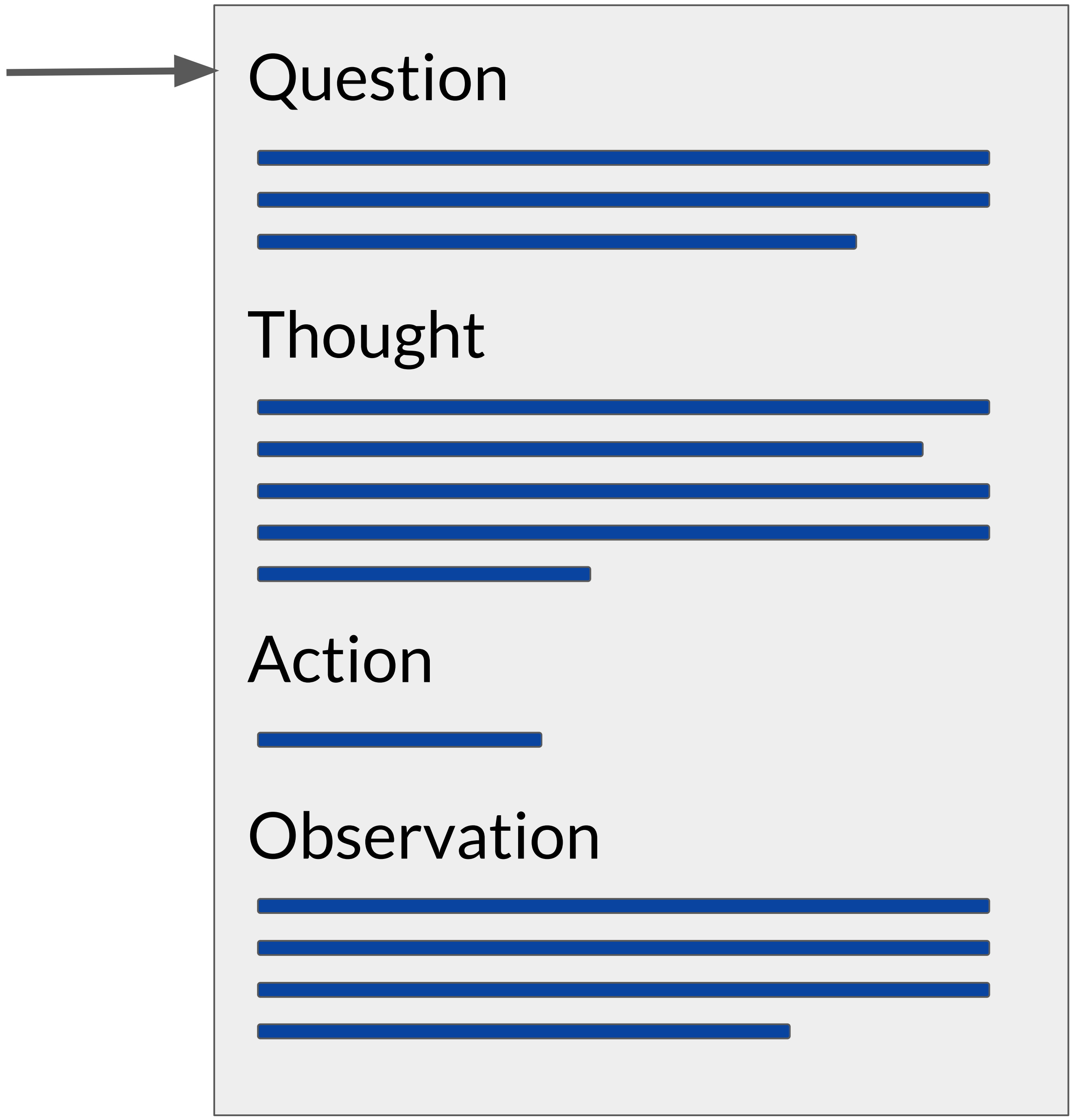




HotPot QA: multi-step question answering
Fever: Fact verification

Source: Yao et al. 2022, "ReAct: Synergizing Reasoning and Acting in Language Models"

ReAct: Synergizing Reasoning and Action in LLMs



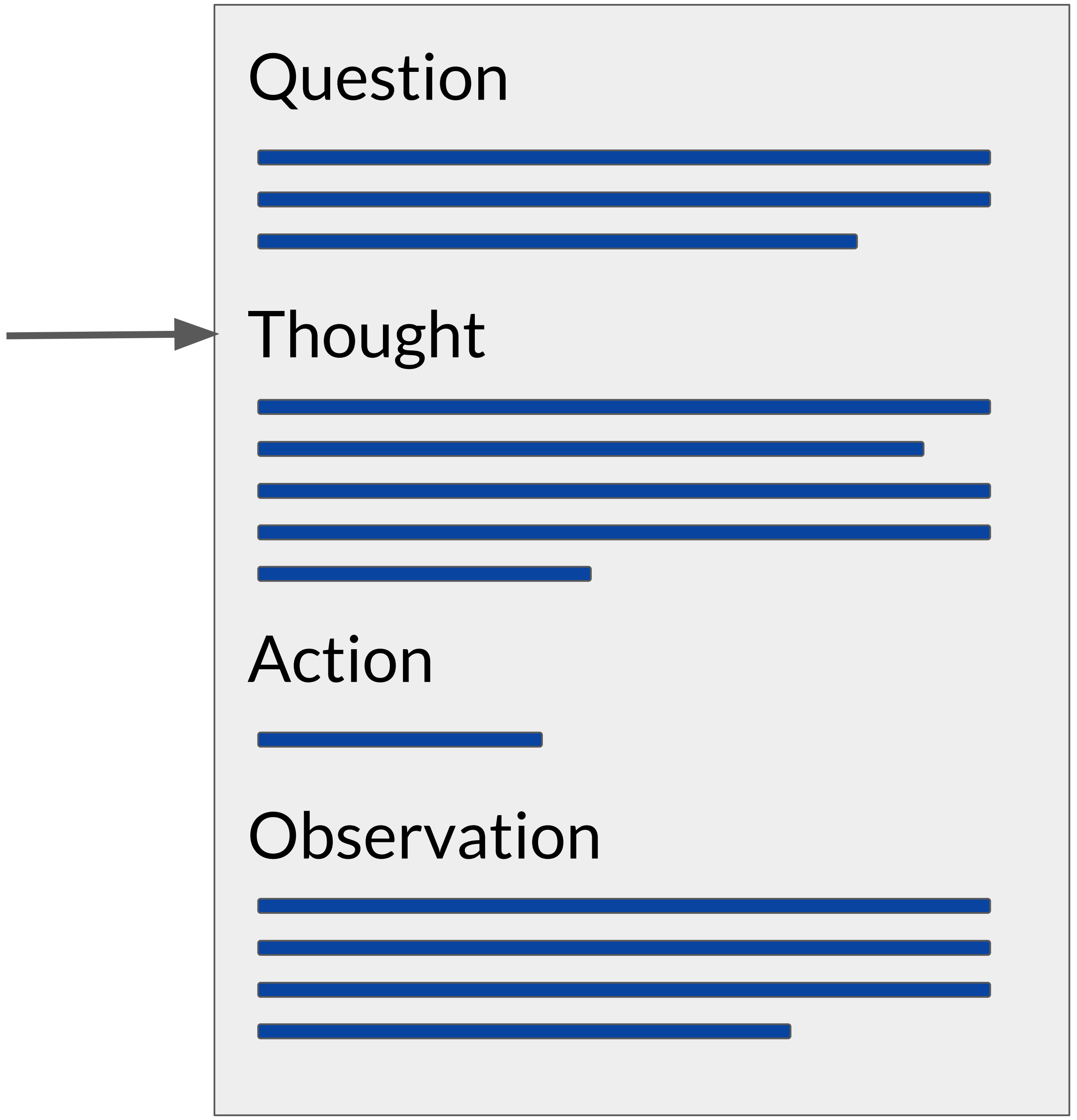
Question: Problem that requires advanced reasoning and multiple steps to solve.

E.g.

“Which magazine was started first, *Arthur’s Magazine* or *First for Women*?”

Source: Yao et al. 2022, “ReAct: Synergizing Reasoning and Acting in Language Models”

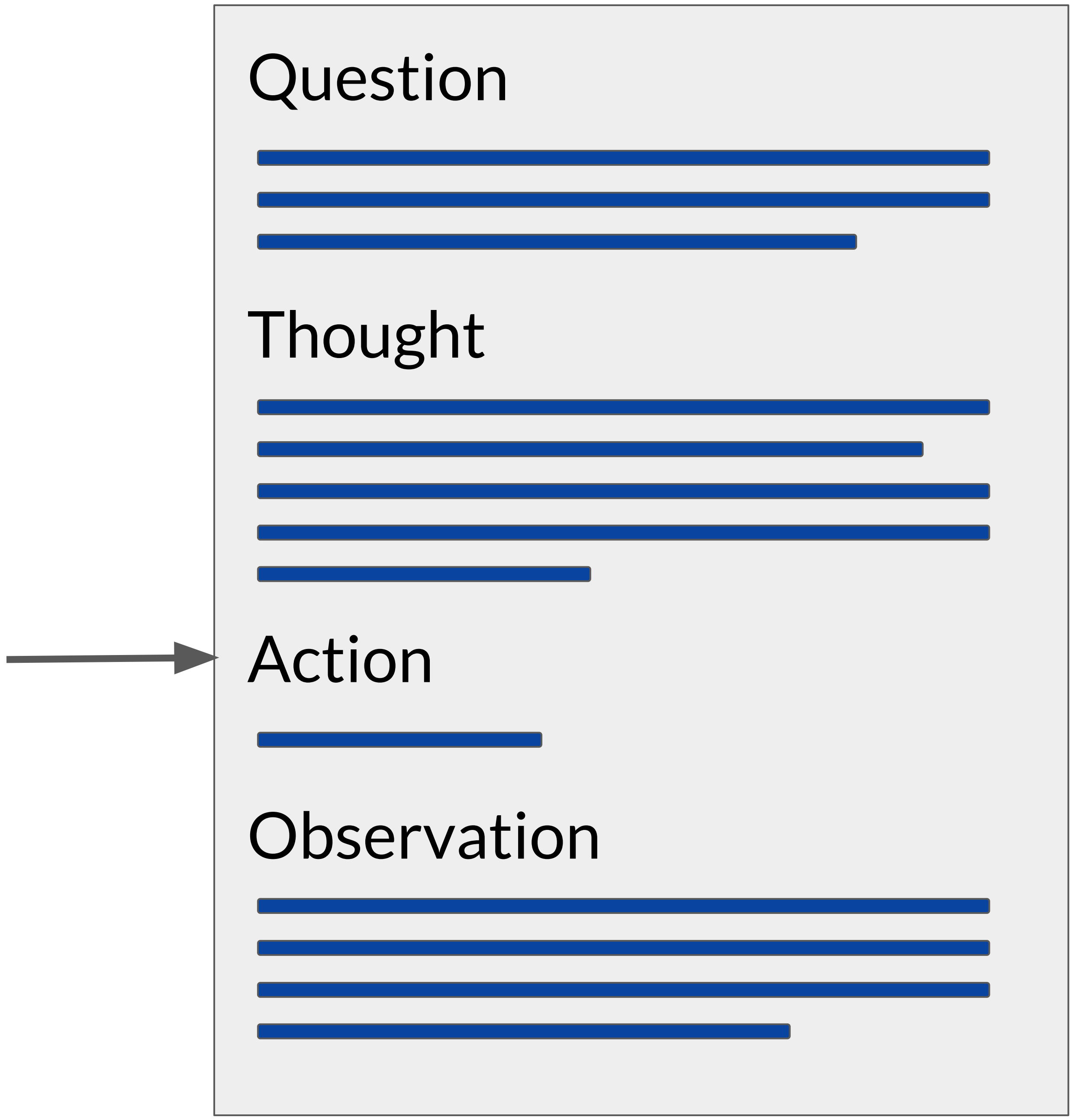
ReAct: Synergizing Reasoning and Action in LLMs



Thought: A reasoning step that identifies how the model will tackle the problem and identify an action to take.

“I need to search Arthur’s Magazine and First for Women, and find which one was started first.”

ReAct: Synergizing Reasoning and Action in LLMs



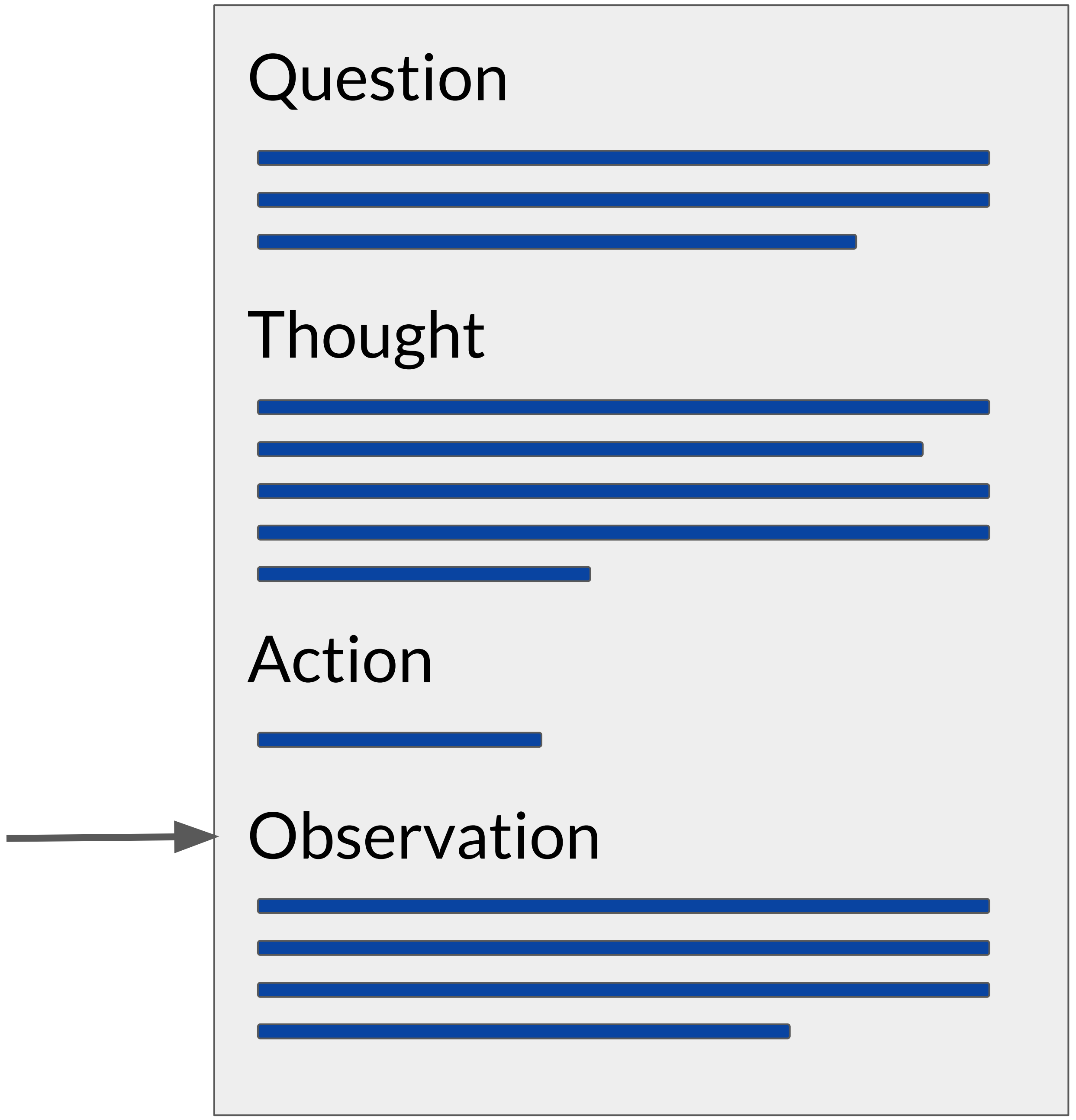
Action: An external task that the model can carry out from an allowed set of actions.

E.g.
`search``[entity]`
`lookup``[string]`
`finish``[answer]`

Which one to choose is determined by the information in the preceding thought.

`search``[Arthur's Magazine]`

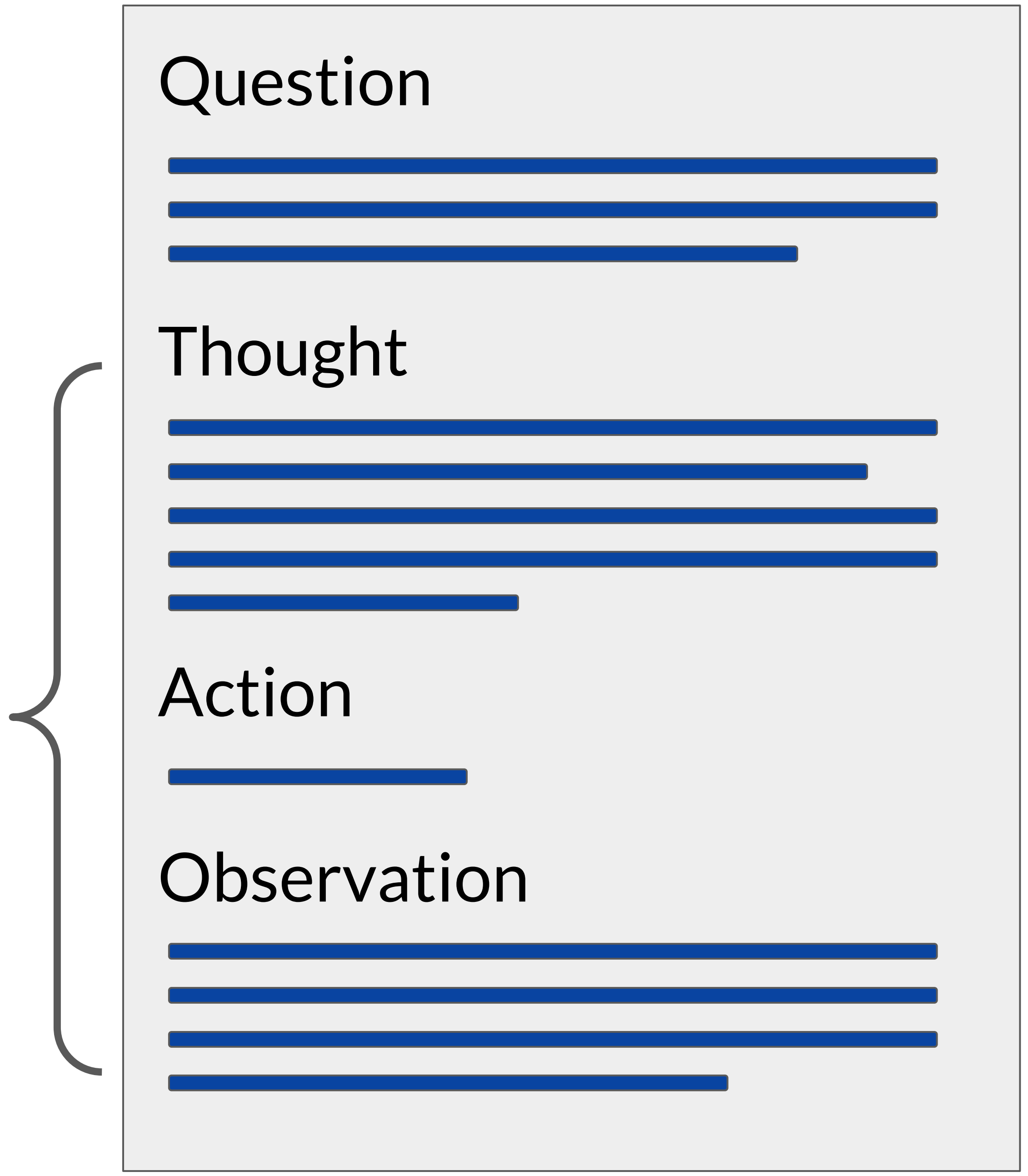
ReAct: Synergizing Reasoning and Action in LLMs



Observation: the result of carrying out the action

E.g.
"Arthur's Magazine (1844-1846) was an American literary periodical published in Philadelphia in the 19th century."

ReAct: Synergizing Reasoning and Action in LLMs

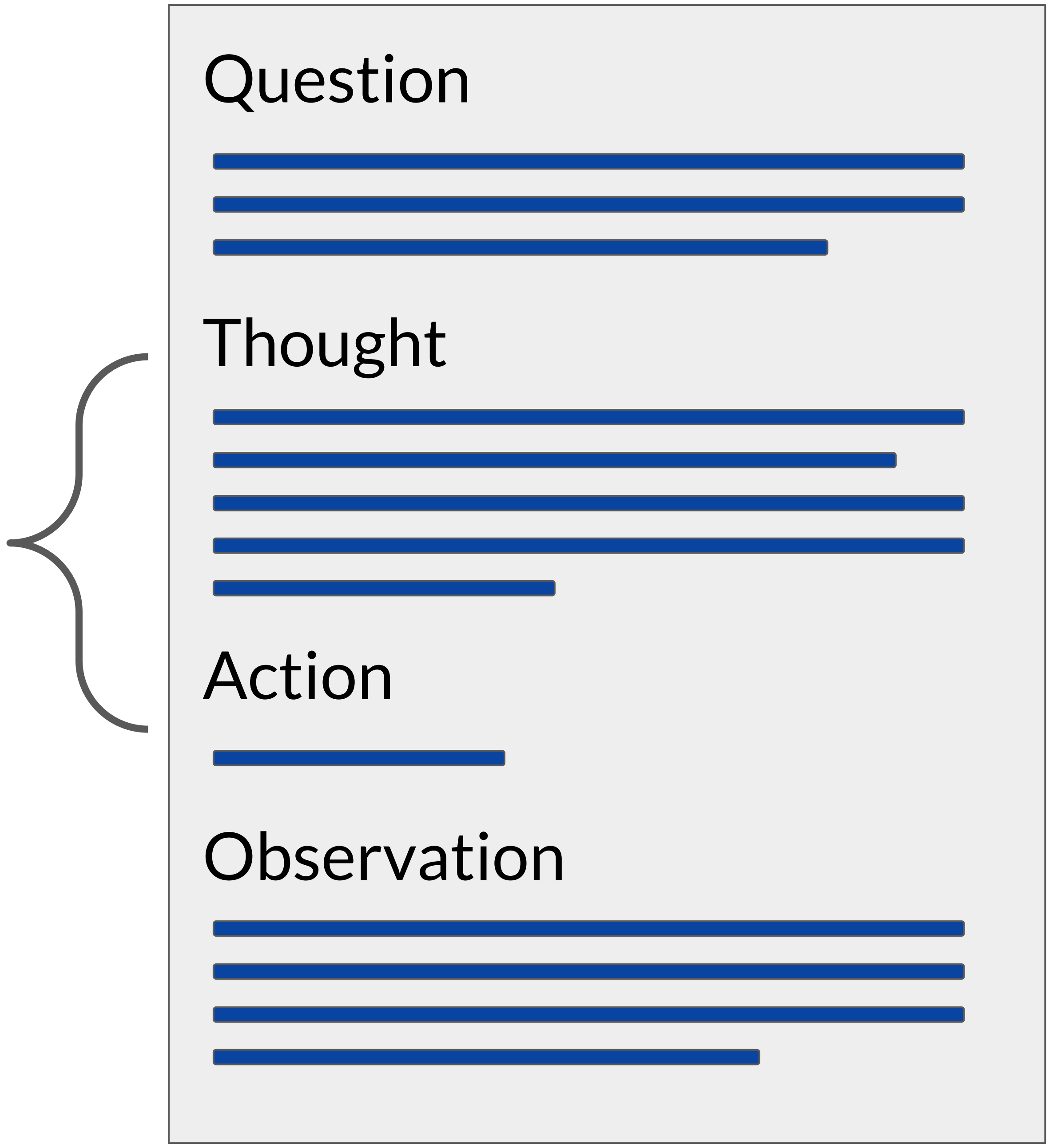


Thought 2:
"Arthur's magazine was started in 1844. I need to search First for Women next."

Action 2:
`search[First for Women]`

Observation 2:
"First for Women is a woman's magazine published by Bauer Media Group in the USA.[1] The magazine was started in 1989."

ReAct: Synergizing Reasoning and Action in LLMs



Thought 3:
"First for Women was started in 1989. 1844 (Arthur's Magazine) < 1989 (First for Women), so Arthur's Magazine as started first"

Action 2:
`finish[Arthur's Magazine]`

ReAct instructions define the action space

```
Solve a question answering task with interleaving Thought, Action,
Observation steps.

Thought can reason about the current situation, and Action can be
three types:
(1) Search[entity], which searches the exact entity on Wikipedia and
returns the first paragraph if it exists. If not, it will return
some similar entities to search.
(2) Lookup[keyword], which returns the next sentence containing
keyword in the current passage.
(3) Finish[answer], which returns the answer and finishes the task.
Here are some examples.
```

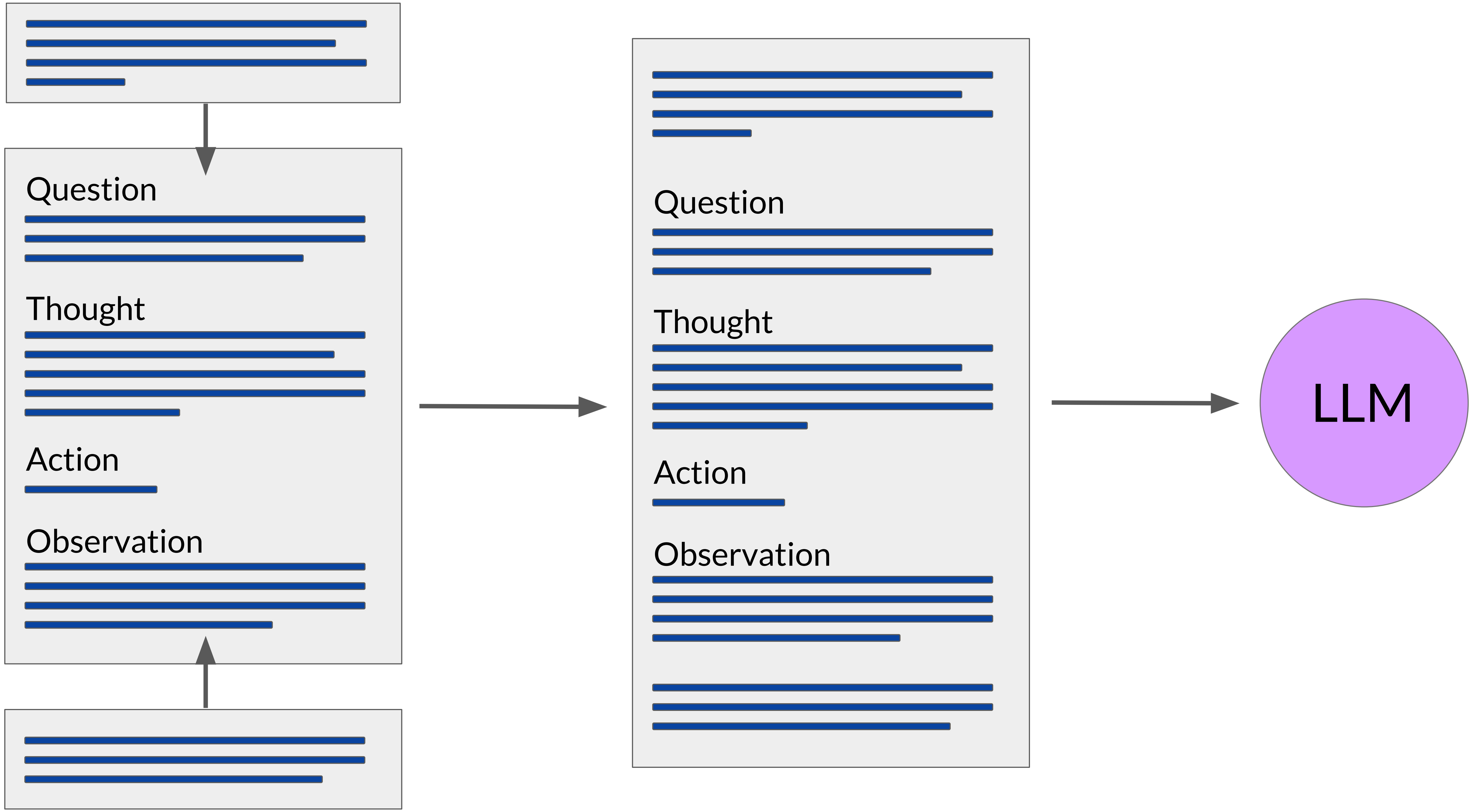
Building up the ReAct prompt

Instructions

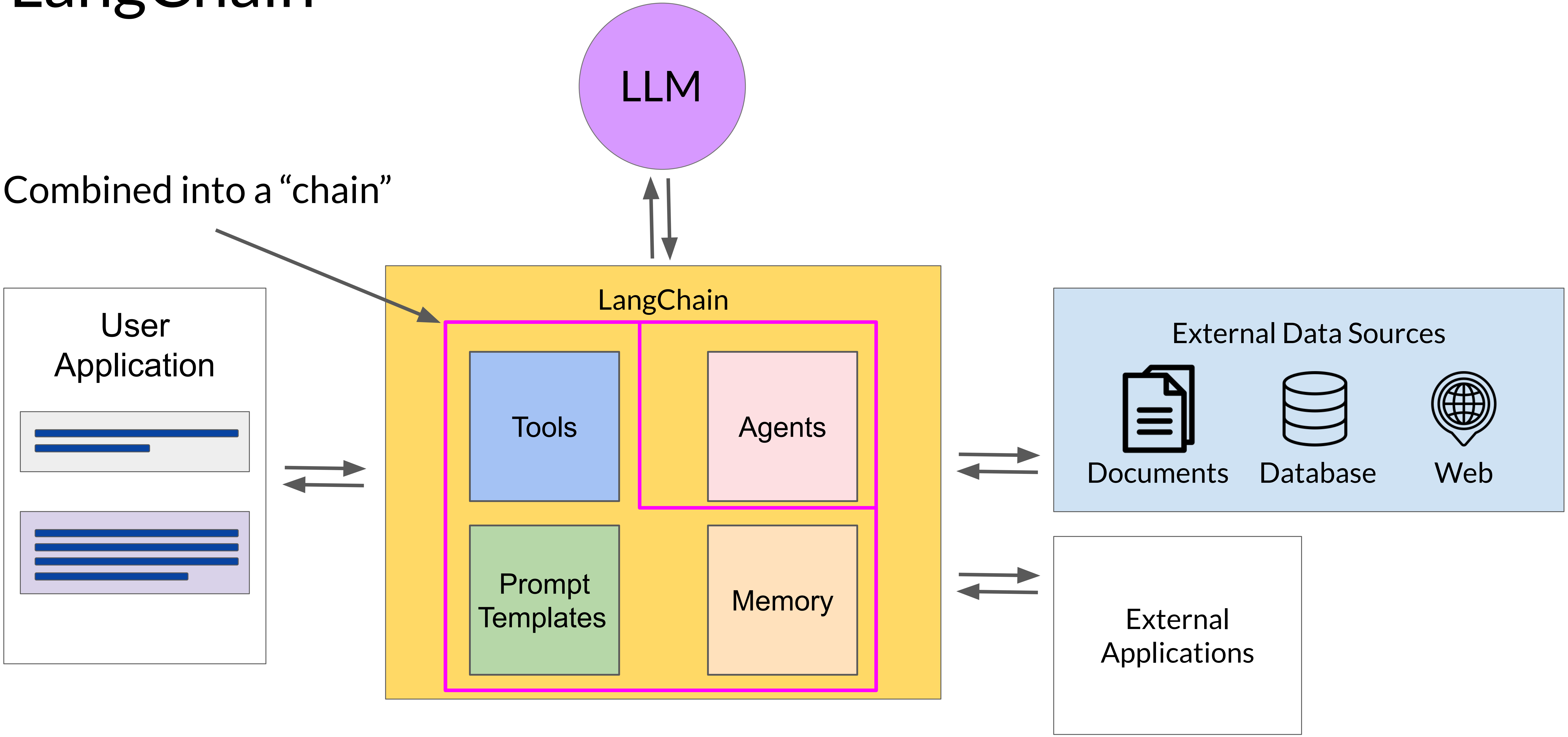
ReAct example

(could be more than one example)

Question to be answered



LangChain



LLM
Combined into a "chain"
User Application
LangChain
Tools
Agents
Prompt Templates
Memory
External Data Sources
Documents
Database
Web
External Applications

The significance of scale: application building

*Bert-base

LLM powered application architectures

Building generative applications

Infrastructure e.g. Training/Fine-Tuning, Serving, Application Components

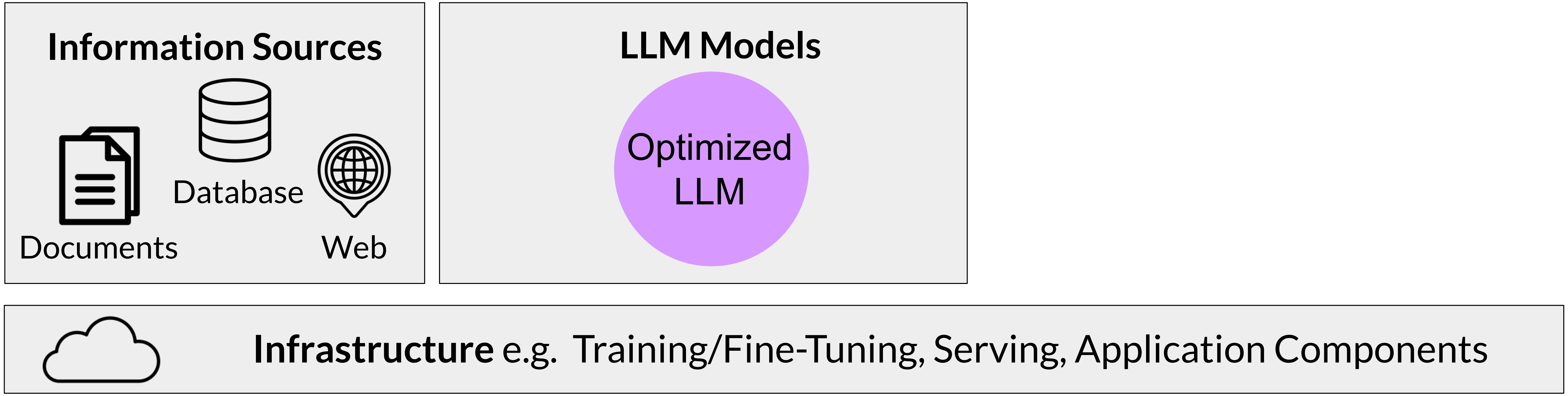
Building generative applications

LLM Models

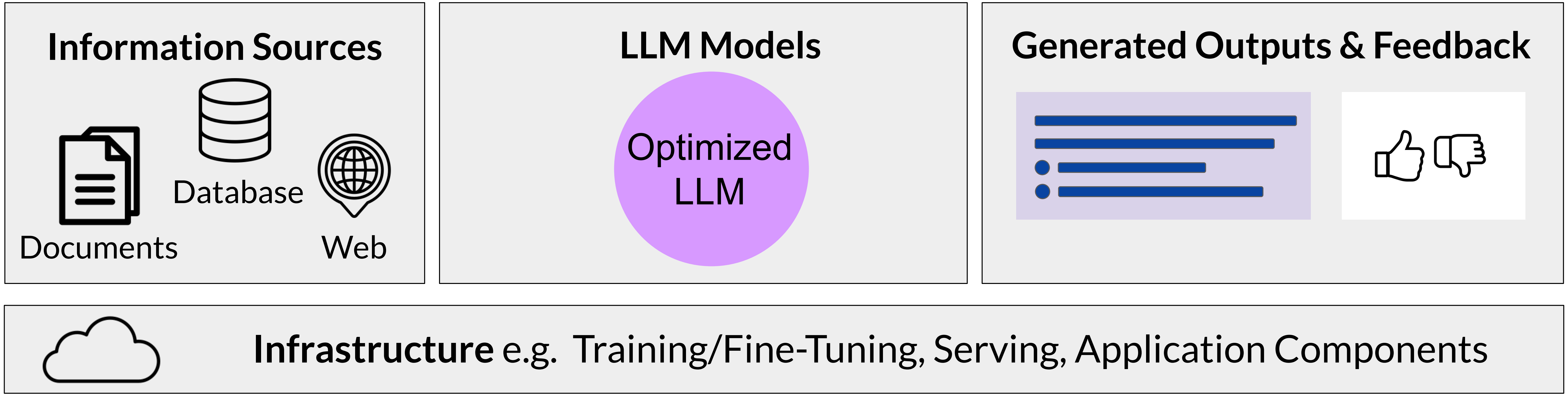
Optimized LLM

Infrastructure e.g. Training/Fine-Tuning, Serving, Application Components

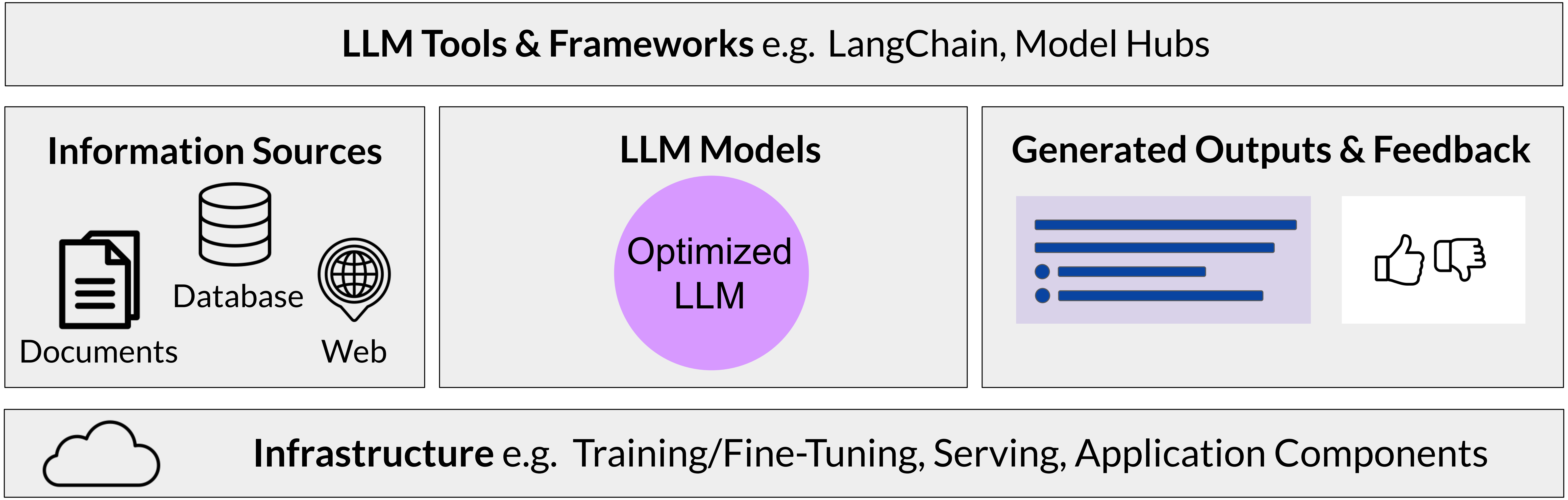
Building generative applications



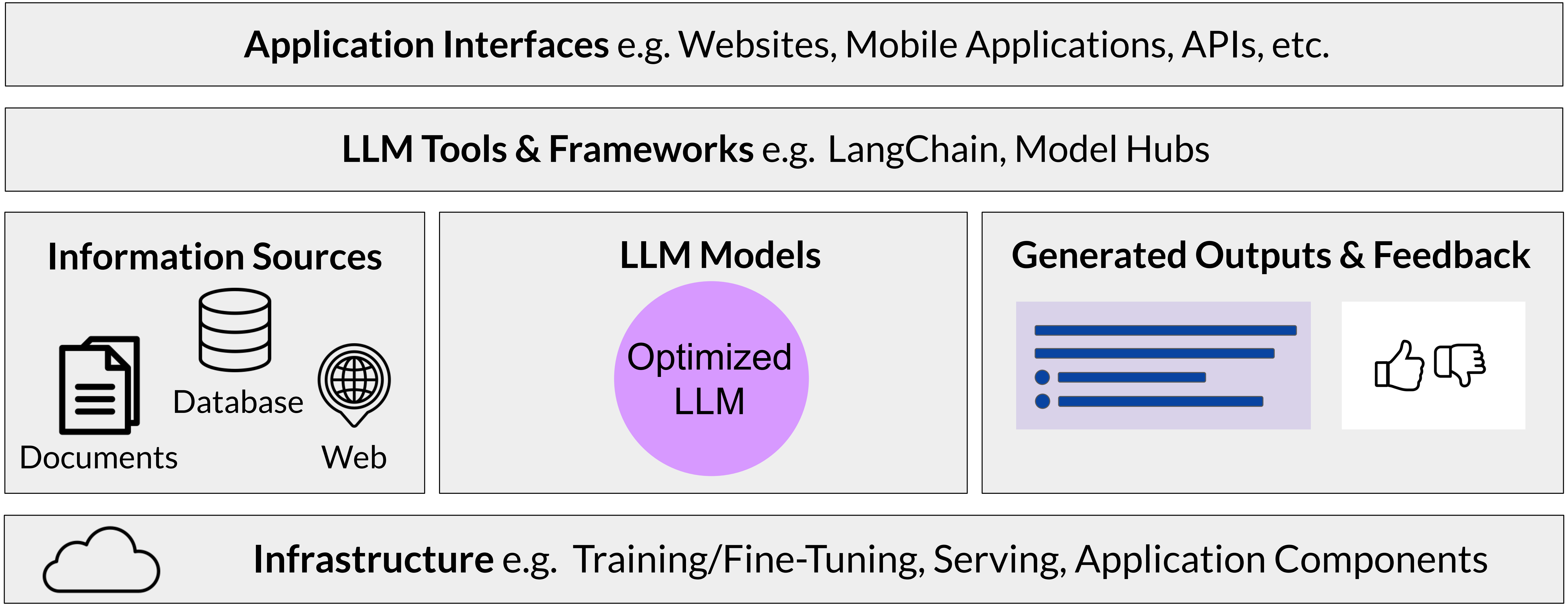
Building generative applications



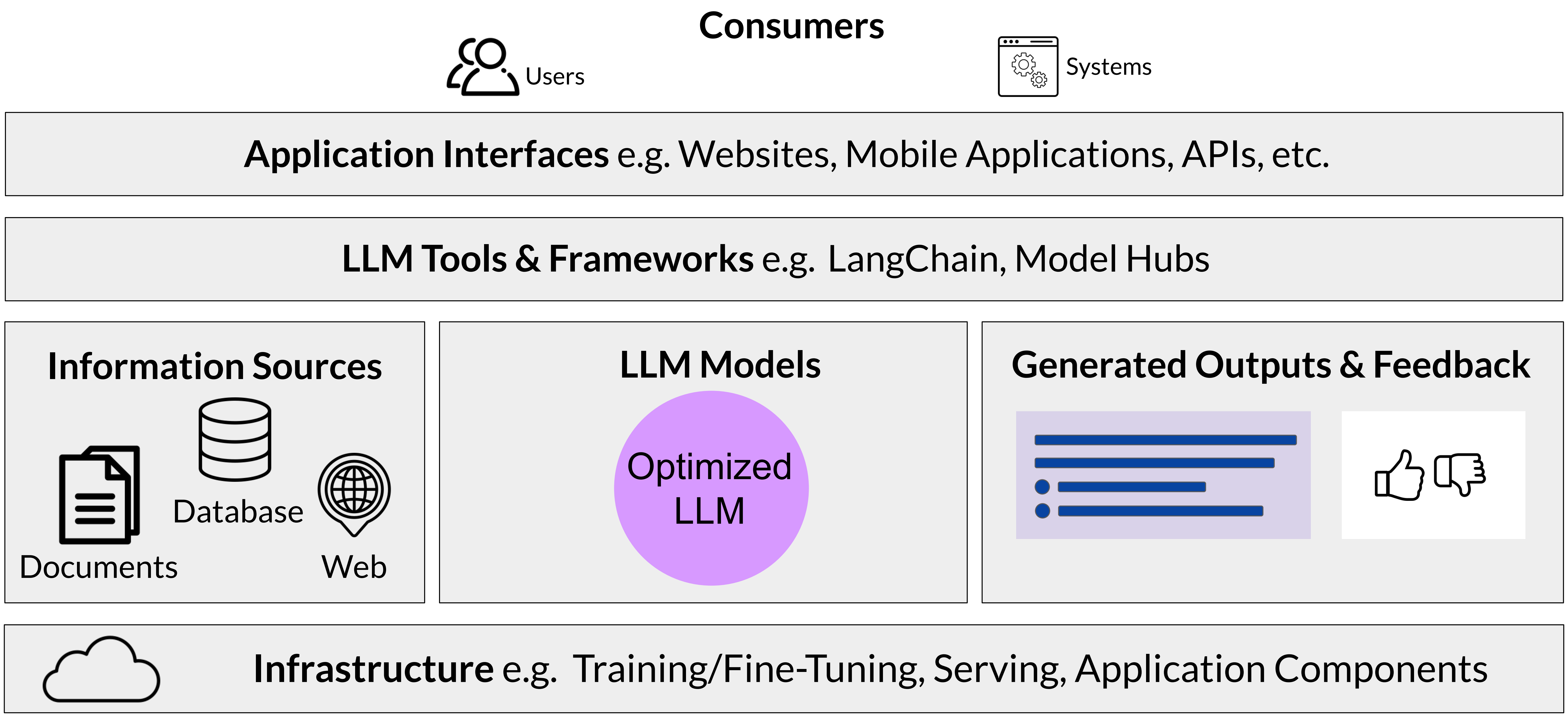
Building generative applications



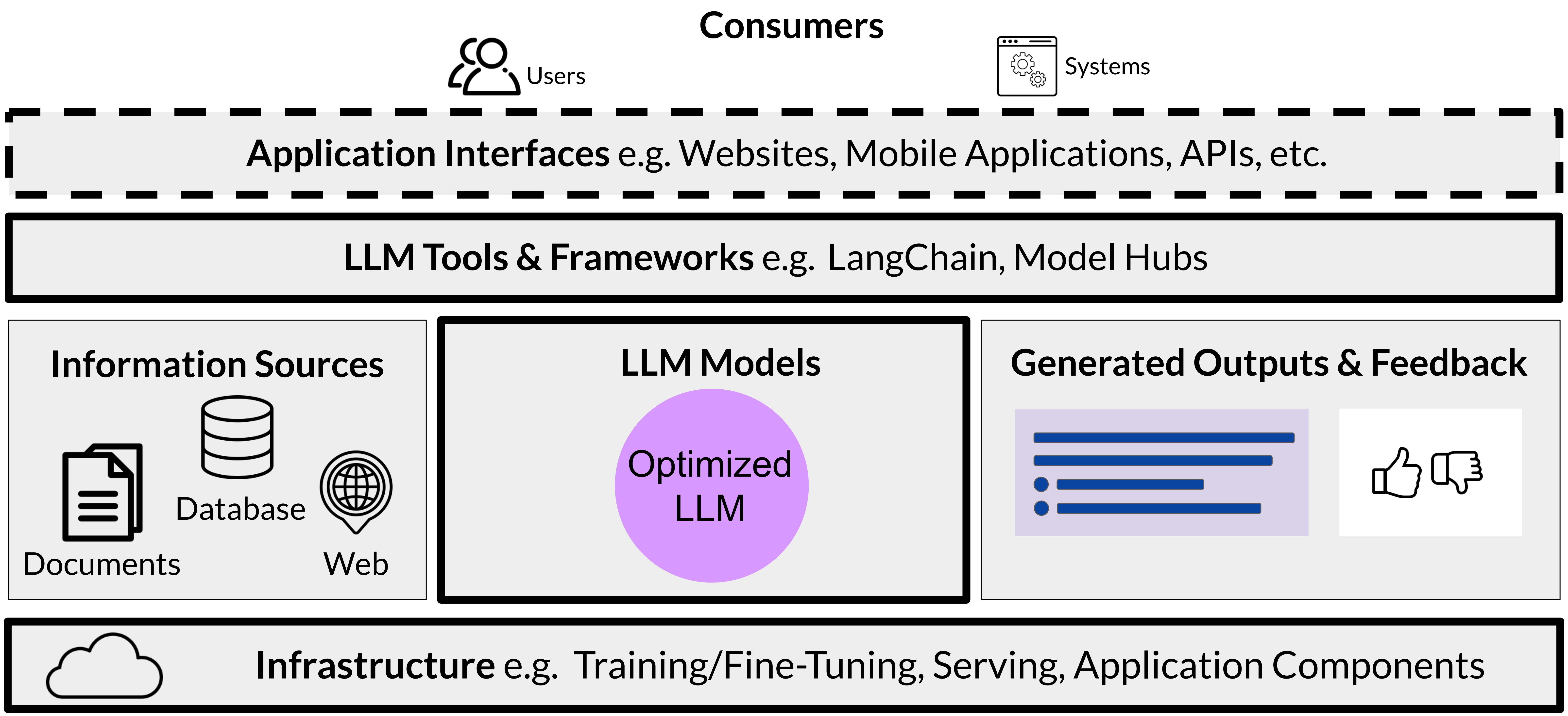
Building generative applications



Building generative applications



Building generative applications



Conclusion, Responsible AI, and on-going research

Responsible AI

Dr. Nashlie Sephus

Responsible AI
Dr. Nashlie Sephus

Responsible AI

Dr. Nashlie Sephus

On-going research

- Responsible AI

Responsible AI

Special challenges of responsible generative AI

- Toxicity
- Hallucinations
- Intellectual Property

Toxicity

LLM returns responses that can be potentially harmful or discriminatory towards protected groups or protected attributes

How to mitigate?

- Careful curation of training data
- Train guardrail models to filter out unwanted content
- Diverse group of human annotators

Hallucinations

LLM generates factually incorrect content

How to mitigate?

- Educate users about how generative AI works
- Add disclaimers
- Augment LLMs with independent, verified citation databases
- Define intended/unintended use cases

Intellectual Property

Ensure people aren't plagiarizing, make sure there aren't any copyright issues

How to mitigate?

- Mix of technology, policy, and legal mechanisms
- Machine "unlearning"
- Filtering and blocking approaches

Responsibly build and use generative AI models

- Define use cases: the more specific/narrow, the better
- Assess risks for each use case
- Evaluate performance for each use case
- Iterate over entire AI lifecycle

On-going research

- Responsible AI
- Scale models and predict performance
- More efficiencies across model development lifecycle
- Increased and emergent LLM capabilities