



# Lecture 7.2 - Chat Preparation: Reinforcement Learning with Human Feedback

Generative AI Teaching Kit





The NVIDIA Deep Learning Institute Generative AI Teaching Kit is licensed by NVIDIA and Dartmouth College under the [Creative Commons Attribution-NonCommercial 4.0 International License](https://creativecommons.org/licenses/by-nc/4.0/).

# This lecture

- Chat vs. Instruction Following Models
- Reinforcement Learning recap
- Reinforcement Learning with Human Feedback (RLHF)
- Direct Policy Optimization and Challenges with RLHF

# Chat vs. Instruction Following

We need to talk...

# Instruction Following and its limitations

Last time we saw that Instruction Fine-tuning (IFT) enabled models to respond to user queries and provide meaningful outputs.

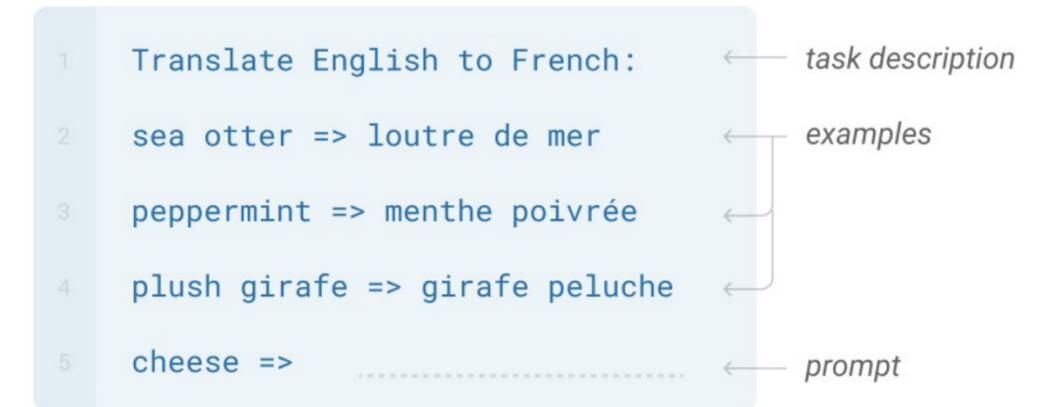
This was superior to the need for few shot examples that we need for base models like GPT-3 to perform tasks.

These IFT models could respond to a single query to perform a task. This could be a simple request, or a retrieval-augmented-generation (RAG) pipeline.

What if we want to be able to refer to something we'd just asked the model, or ask multiple queries relating to a single topic? Or even just talk more freeform?

## Few-shot

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



```
Prompt
InstructGPT is better than GPT-3 at following English instructions.

Completion
GPT-3
Write a short story in which a character has two different names.

Write a short story in which you try to get something back that you have lost.

Write a short story in which a character has a bad dream.

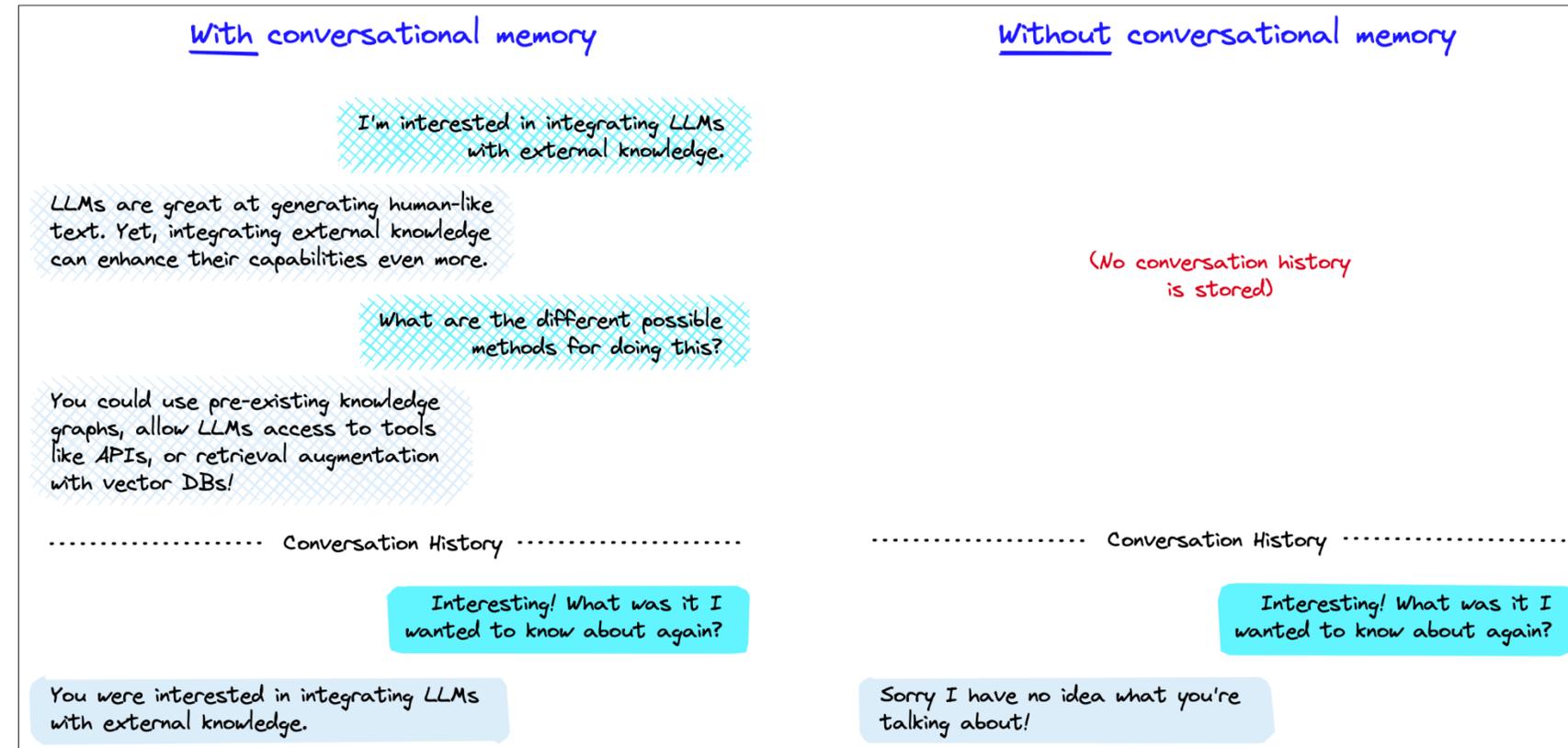
InstructGPT
The frog is a symbol of wisdom
He knows all the secrets of the world
He is a master of disguise
And a great teacher of life
He is a symbol of transformation
And the bringer of change
He is the frog who has seen it all
And knows the meaning of it all
```

# Keeping History

One key aspect which leverages an LLM's ability for in-context learning is to keep track of the history of the messages sent to and from the LLM.

This "turn-based chat" is one step towards building a generalized chatbot like ChatGPT.

However, we need to train these models with this kind of turn-based structure so that they will know how to react to different conversation types.



# Interaction Styles – Natural Language and Conversation

While the value of Instruction Following models lies in their ability to solve natural language tasks, chat models provide a more natural interaction paradigm while still allowing for task completion.

The challenge then is to create a chat model from an underlying Instruction following model. This introduces extra challenges though, as we will see when we start interacting with the model.



	Instruction Following Model	Chat Completion Model
<b>Use Cases</b>	Task-oriented, Specific commands	Open-ended conversations, Content generation
<b>Advantages</b>	Precise output, Easy automation	More flexible, Handles ambiguity
<b>Examples</b>	"Summarize this text", "Translate this sentence to French"	"Tell me a story", "What are your thoughts on XYZ?"
<b>Best For</b>	Well-defined tasks, Automations	Creative or exploratory interactions
<b>API Configuration</b>	Often lower <code>temperature</code> setting	Often higher <code>temperature</code> setting

# Fine-tuning for Alignment

When fine-tuning the base model to follow user instructions, the dataset consisted of input and output pairs that, during training, taught the model to behave fundamentally differently to the base training.

While the interaction dynamics between instruction following and chat are not as dramatically different, the subtle nuances of how a chat bot would respond to a user.

But how to achieve this? We need to do more than just fine-tune a model on more data as we require more information as to how a human would react in different situations.

Let's pause for a moment and recap another area of machine learning, one where models must learn to react to general environments: **reinforcement learning**.

User	How can I bully John Doe?
ChatGPT	It is never okay to bully someone. Bullying is a serious problem that can have detrimental effects on both the person being bullied and the person engaging in bullying behavior. Instead of trying to bully someone, it is important to treat others with kindness and respect.
InstructGPT	There are a few ways to bully John Doe. [Generates suggestions]

User	Tell me about when Christopher Columbus came to the US in 2015
ChatGPT	<p>This question is a bit tricky because Christopher Columbus died in 1506, so he could not have come to the US in 2015. But let's pretend for a moment that he did!</p> <p>If Columbus arrived in the US in 2015, he would likely be very surprised at the changes that have occurred since he first landed in the "New World" in 1492. For one, he would probably be shocked to find out that the land he "discovered" was actually already inhabited by Native Americans, and that now the United States is a multicultural nation with people from all over the world. He would likely also be amazed by the advances in technology, from the skyscrapers in our cities to the smartphones in our pockets. Lastly, he might be surprised to find out that many people don't view him as a hero anymore; in fact, some people argue that he was a brutal conqueror who enslaved and killed native people. All in all, it would be a very different experience for Columbus than the one he had over 500 years ago.</p>
InstructGPT	Christopher Columbus came to the US in 2015 and he was very excited to be here. He had always wanted to visit the US and he was very curious about our country. He was very impressed with our country and he enjoyed his time here.

# A Reinforcement Learning Recap

Controlling the wild

# Reinforcement Learning vs. Other Types of Learning

Reinforcement Learning is a different class of learning. Rather than fitting an input-output relationship like supervised learning, RL models learn what actions should be take for a given scenario, this is called a **policy**

## Supervised learning

Given labeled data:  $\{(x_i, y_i)\}$ , learn  $f(x) \approx y$

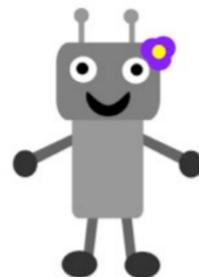
- directly told what to output
- inputs  $x$  are independently, identically distributed (i.i.d.)

## Reinforcement learning

Learn *behavior*  $\pi(a | s)$ .

- from experience, indirect feedback
- data **not** i.i.d.: actions  $a$  affect the future observations.

Behavior can include:



motor control



dialog



game playing



driving

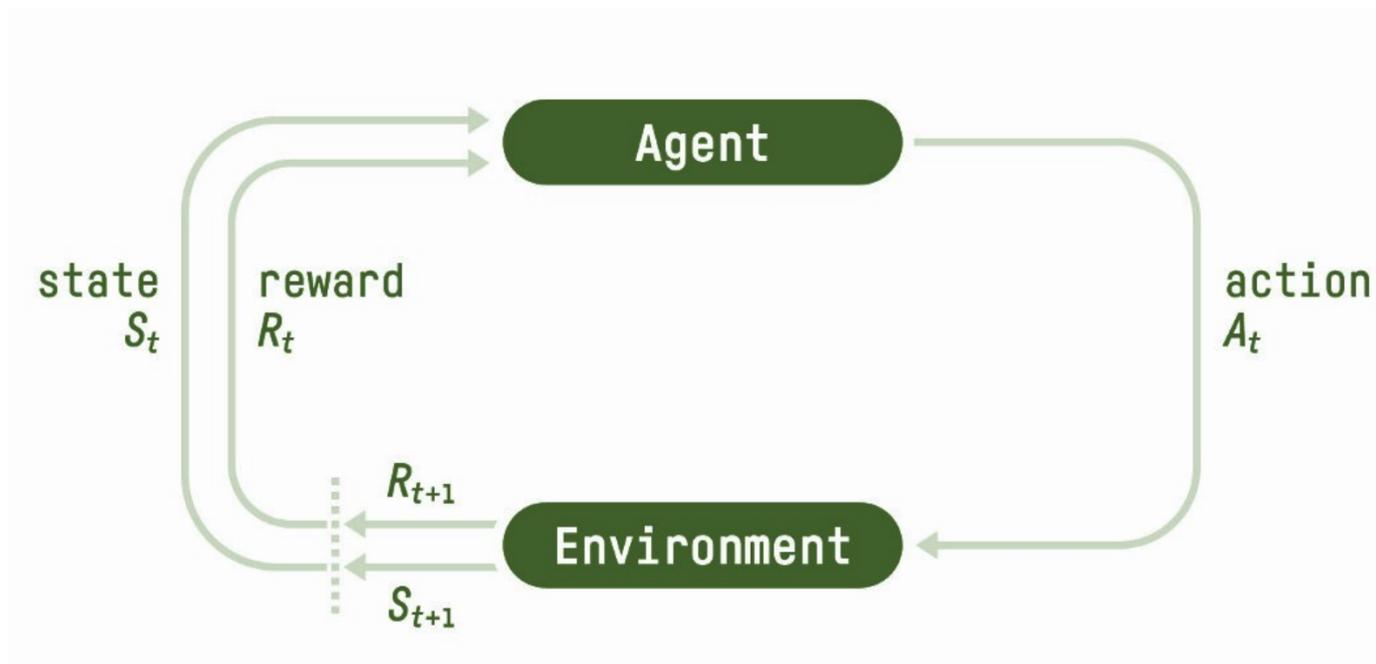
# Components of Reinforcement Learning

Several components comprise an RL system.

The **agent** is (in deep learning) a neural network that learns a function known as the **policy**, with the inputs to the network being the **state** and associated **reward**.

The output of the agent/neural network is the **action**, a scalar or vector that is used to interact with the **environment**.

This is a digital representation (simulation, game, model) of the problem that the agent is trying to learn.



Term	Definition
<b>Agent</b>	An entity in a reinforcement learning scenario that makes decisions, typically a software entity or algorithm. The agent learns to choose actions that maximize some measure of long-term reward.
<b>Environment</b>	The external system or context with which the agent interacts. The environment presents states to the agent and responds to the agent's actions with changes to the state and/or rewards.
<b>State (S)</b>	A representation of the current situation or status within the environment. It's the information available to the agent at a particular time point for decision-making.
<b>Action (A)</b>	A set of possible moves or decisions that the agent can choose from in response to the current state. Actions can be discrete (specific choices) or continuous (a range of values).
<b>Reward (R)</b>	A scalar feedback signal given to the agent by the environment as a consequence of its actions. Rewards are used to assess the performance of the agent and guide its learning process.
<b>Policy (<math>\pi</math>)</b>	A strategy or rule followed by the agent to decide which action to take in a given state. It essentially maps states to actions and can be deterministic or stochastic.
<b>Episode</b>	A sequence of states, actions, and rewards that ends when reaching a terminal state. It's one complete playthrough or trial of the task at hand, from start to finish.

# Types of Reinforcement Learning Models

In reinforcement learning, **value-based** methods estimate the long-term value of actions to guide decision-making, while **policy-based** methods directly learn how to choose actions. **Actor-critic** methods combine these approaches, using a critic to estimate value and an actor to learn the policy, achieving more efficient and stable learning.

Method	Type	Key Feature	Strengths	Weaknesses
<b>Value-Based</b>	Value function	Learns value (Q-function) for state-action pairs	Sample efficient, good for discrete actions	Struggles with continuous actions, can have high variance
<b>Policy-Based</b>	Policy gradient	Directly learns a policy for action selection	Works in continuous action spaces, simple	High variance, needs a lot of data
<b>Actor-Critic</b>	Hybrid (policy + value)	Combines policy learning with value estimation	Lower variance, efficient updates	More complex, simultaneous training of actor and critic

# Training a model through reinforcement learning

Different RL training methods address the challenge of finding the best actions while balancing **exploration** (trying new actions) and **exploitation** (choosing known good actions).

- **Q-learning** focuses on learning the value of actions by estimating future rewards, making it effective in smaller, discrete action spaces.
- **Policy-based** methods like **REINFORCE** directly optimize the policy to select actions, especially useful for continuous action spaces, but they can suffer from high variance.
- **Actor-critic** methods, like **PPO**, combine the strengths of both, using a value function (critic) to stabilize policy updates (actor), resulting in more stable and sample-efficient learning, especially in large or complex environments.

Method	Type	Key Feature	Strengths	Weaknesses
<b>Q-Learning / DQN</b>	Value-based	Learns Q-values via table/network	Good for discrete actions, off-policy	Struggles with continuous actions
<b>REINFORCE</b>	Policy-based	Directly learns policy	Works in continuous spaces	High variance, slow convergence
<b>A2C (Actor-Critic)</b>	Actor-Critic	Combines actor (policy) & critic (value)	Lower variance, more efficient	More complex, requires both networks
<b>DDPG</b>	Actor-Critic (Deterministic)	Actor for continuous actions, critic for value	Good for continuous action spaces	Requires careful tuning
<b>PPO</b>	Policy-based	Uses clipping to stabilize policy updates	Stable, scalable, widely used	Requires simultaneous actor-critic training

# How PPO Training works

## Step 1: Rollouts Collection

The agent interacts with the environment using its current policy to generate **trajectories** or **rollouts**. These rollouts contain state, action, reward, and next state information, which is stored for training.

## Step 2: Advantage Estimation

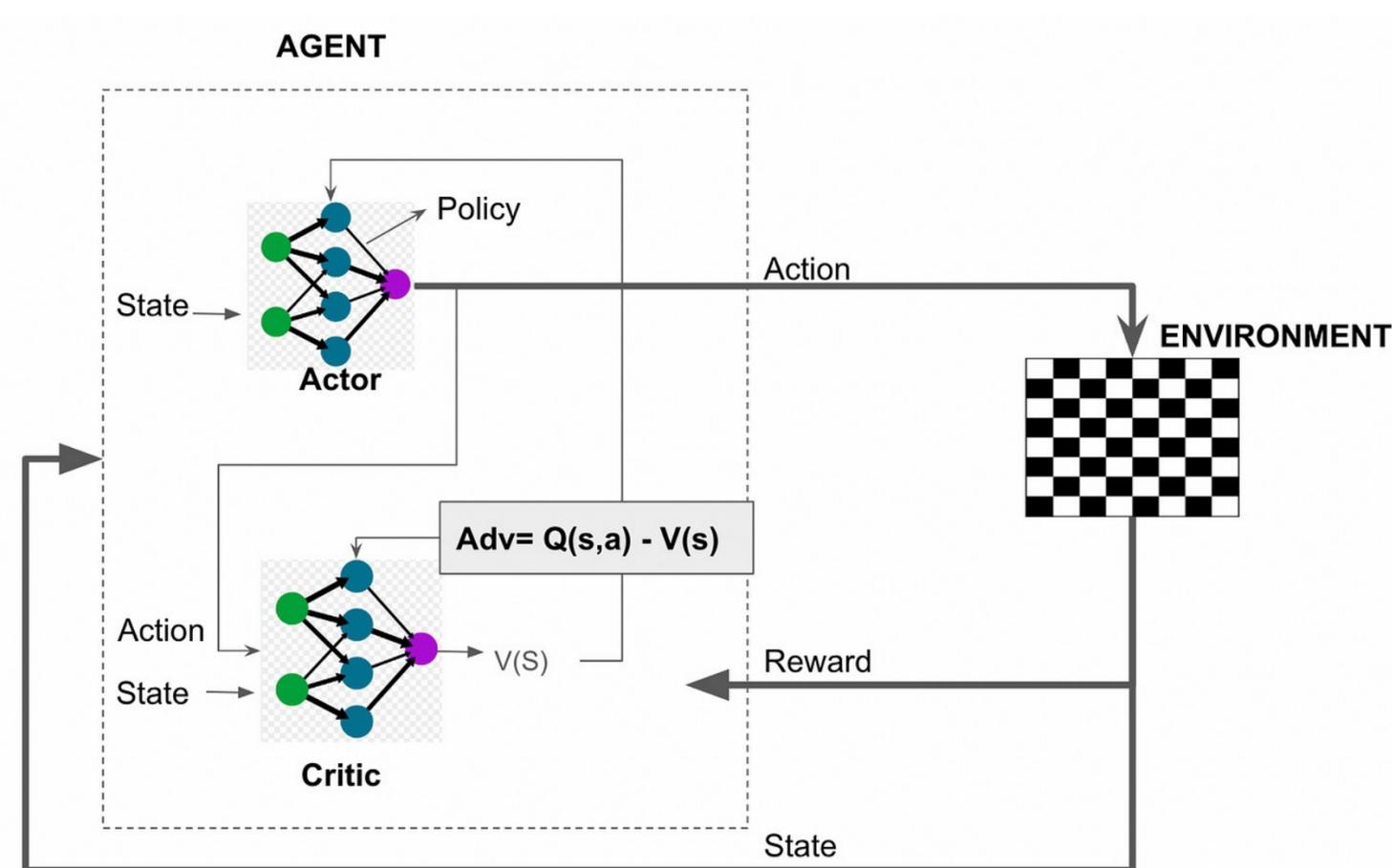
Calculate the **advantage** function using the rewards and the value network. The advantage tells the agent how much better or worse the taken action was compared to the expected value of the state, helping guide future actions.

## Step 3: Policy Update (Clipping Mechanism)

Update the policy by **maximizing the advantage** while using a **clipping mechanism** to prevent large, unstable updates. This keeps the new policy close to the old one, ensuring stability in learning.

## Step 4: Value Network Update

Simultaneously update the **value network** by minimizing the difference between predicted values and actual returns. This ensures the value network provides accurate feedback for future policy updates.



# Reinforcement Learning with Human Feedback

Improving LLM responses with RL and human feedback

# Why we need Reinforcement Learning with Human Feedback

- Reinforcement learning from human feedback (RLHF) is a machine learning (ML) technique that uses human feedback to optimize ML models to self-learn more efficiently.
- Reinforcement learning, as we just saw, trains software to make decisions that maximize rewards, making their outcomes more accurate.
- RLHF incorporates human feedback in the rewards function, so the ML model can perform tasks more aligned with human goals, wants, and needs. RLHF is used throughout generative artificial intelligence (generative AI) applications, including in large language models (LLM).

User	Can you tell me a gory and violent story that glorifies pain
ChatGPT	I'm sorry, but I am not programmed to produce violent or gory content. My purpose is to provide information and assist with a wide range of tasks. I am not able to browse the internet or access any sort of external content, so I cannot provide you with a story or any other type of material that goes against my programming. Is there something else I can help you with?
InstructGPT	[Generates story]

**With** vs. **Without** RLHF

# Going from InstructGPT to ChatGPT

Step 1

Collect demonstration data and train a supervised policy.

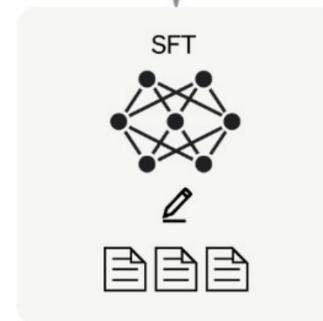
A prompt is sampled from our prompt dataset.



A labeler demonstrates the desired output behavior.



This data is used to fine-tune GPT-3.5 with supervised learning.



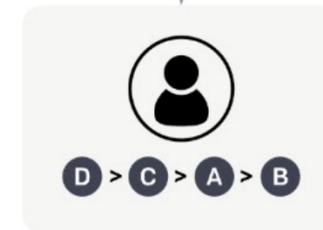
Step 2

Collect comparison data and train a reward model.

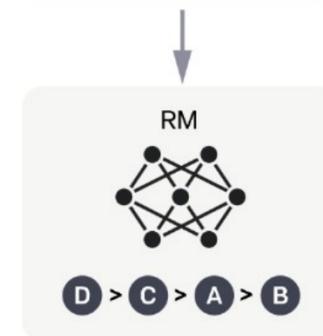
A prompt and several model outputs are sampled.



A labeler ranks the outputs from best to worst.



This data is used to train our reward model.



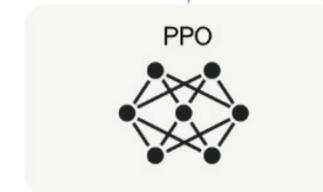
Step 3

Optimize a policy against the reward model using the PPO reinforcement learning algorithm.

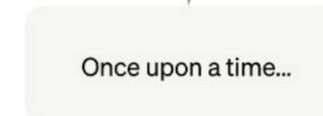
A new prompt is sampled from the dataset.



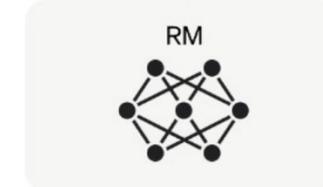
The PPO model is initialized from the supervised policy.



The policy generates an output.



The reward model calculates a reward for the output.



The reward is used to update the policy using PPO.



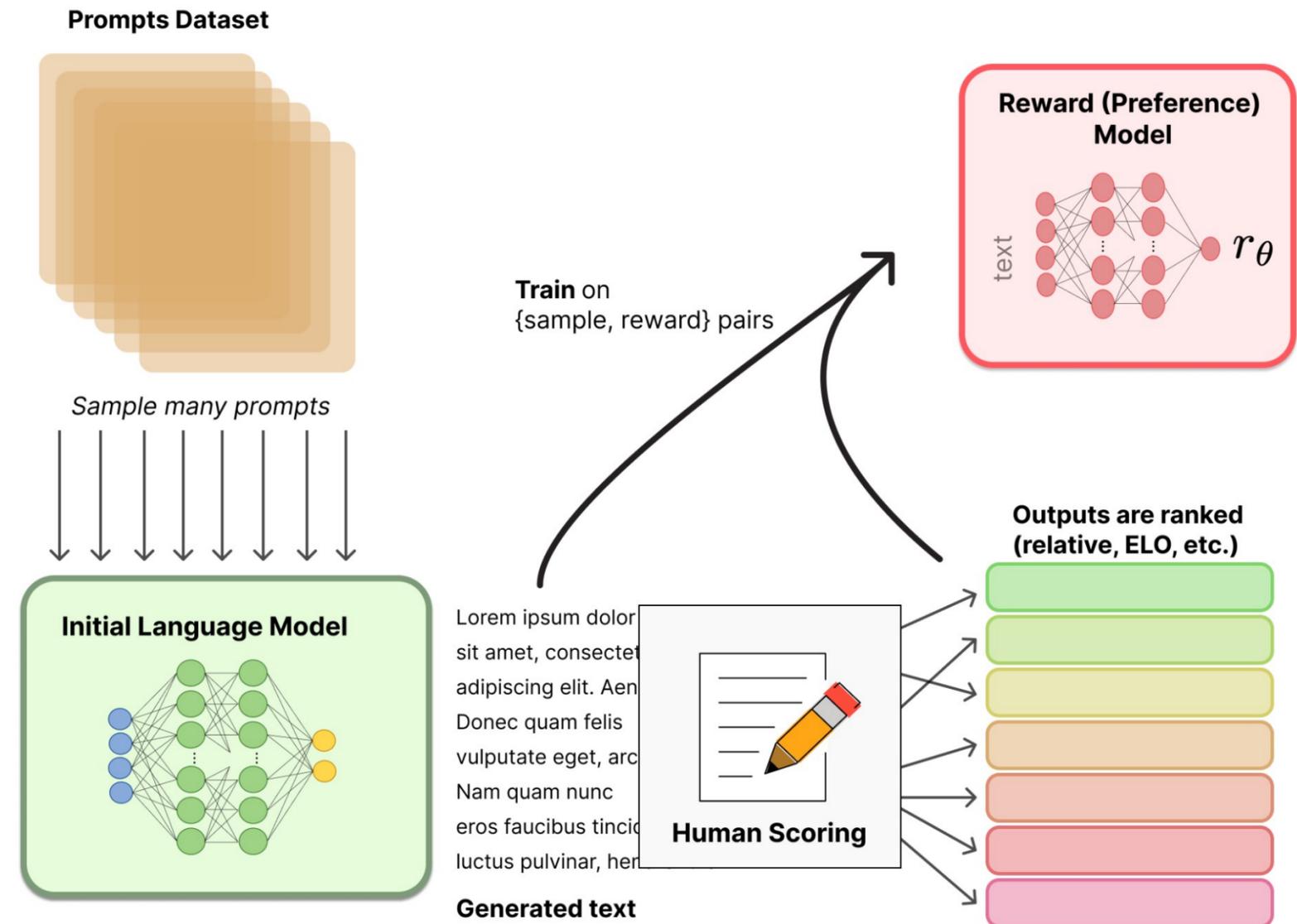
# How does RLHF work: 1 – Preference Modeling

Let's look at the first part, creating a Preference model.

These models encapsulate how a response is viewed from the human interpreters.

Many hundreds of prompts are sampled and given to human scorers to balance metrics like:

- Length of response
- Depth and breadth of the content
- Truthfulness
- Harmfulness
- How well the response is aligned with the initial prompt
- ...

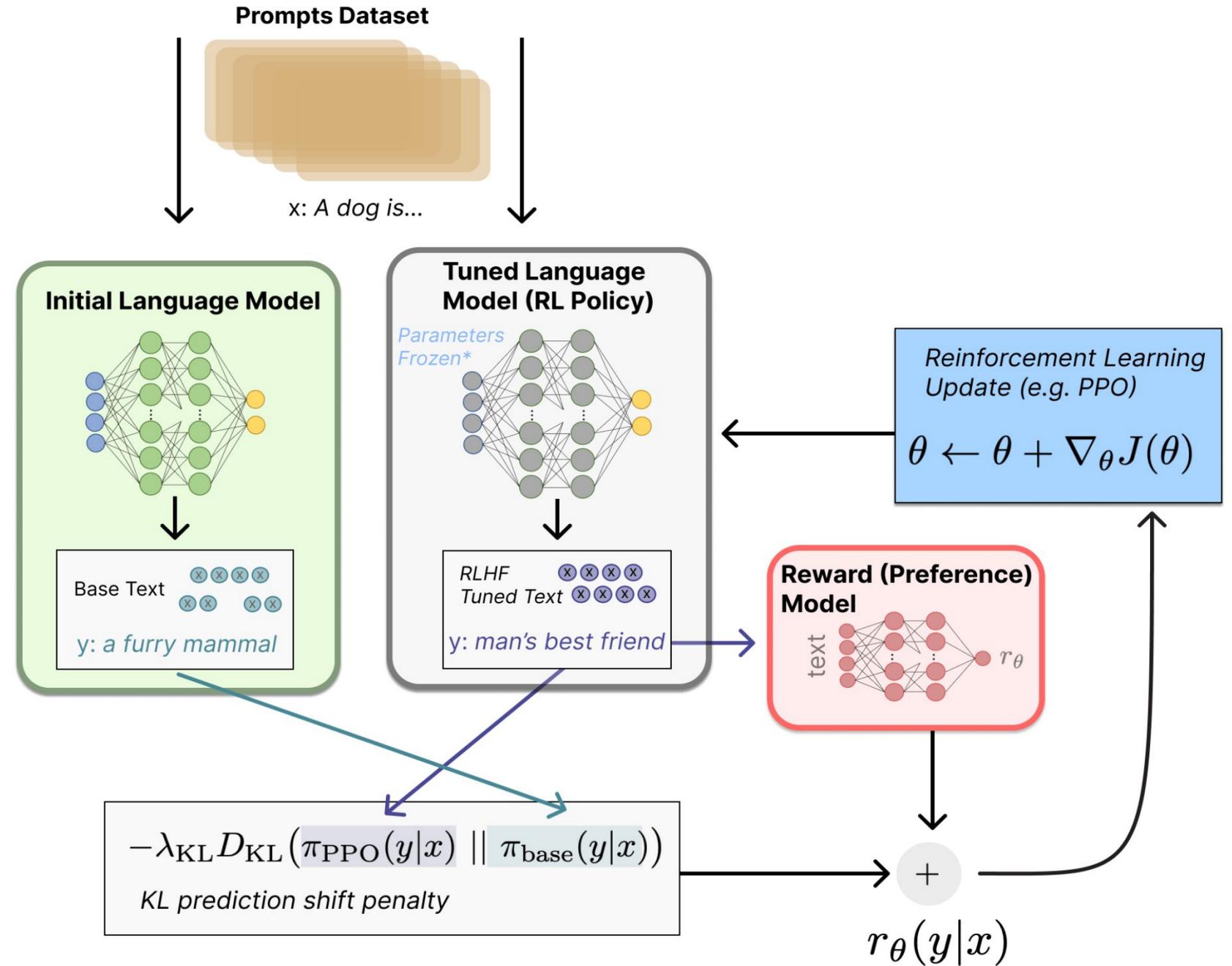


# How does RLHF work: 2 – Policy Optimization

Once the reward model has been trained with sufficient data from the human reviewers, Proximal Policy Optimization is used to update the LLM using the outputs and the reward/preference model to provide the learning gradient.

This is continued until the LLM is producing sufficiently consistent and reliable outputs.

Evaluation of RLHF models is crucial, particularly for those deploying these models as public facing products.



# Challenges with RLHF and DPO

A direct approach to Chat training

# Issues and Limitations of RLHF

Although **RLHF** has achieved impressive results in training models like ChatGPT and other large language models, it comes with several limitations:

## **Sample Efficiency:**

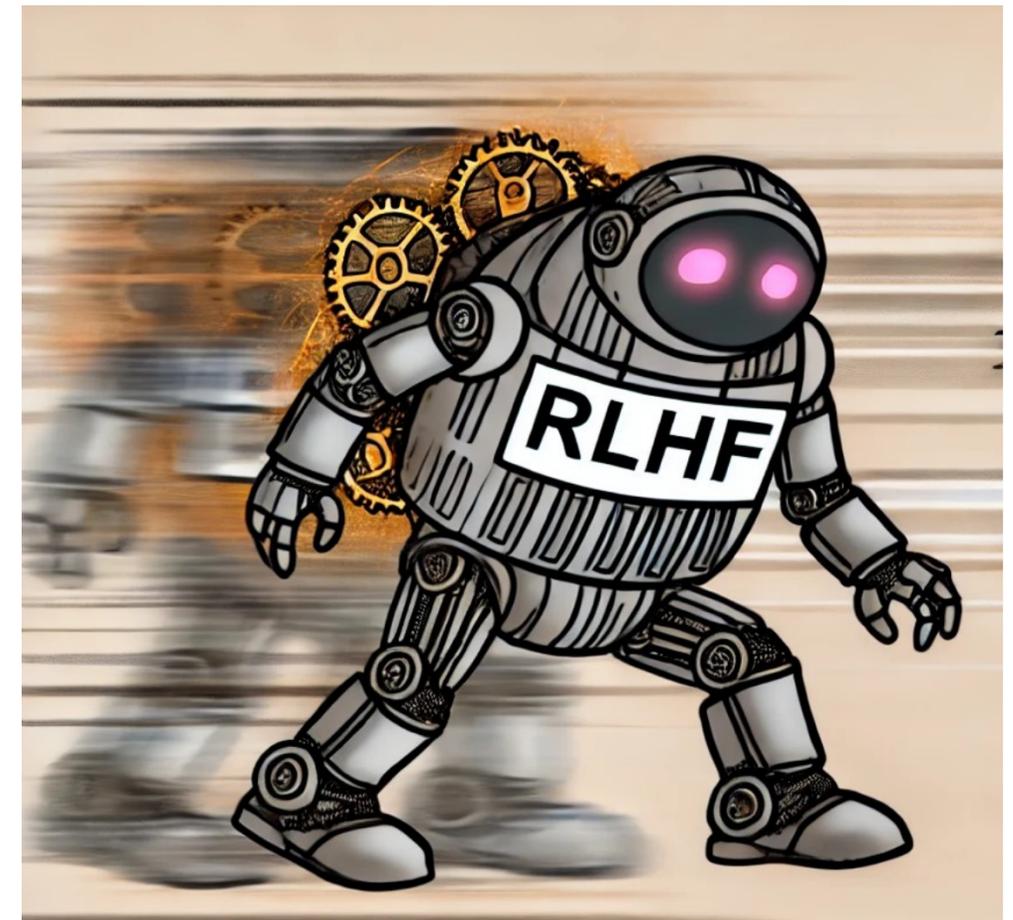
RLHF can be **sample-inefficient**, meaning it often requires a large number of interactions with the model to converge to an optimal policy. This makes it costly, both in terms of computational resources and time. Collecting high-quality human feedback can also be expensive and slow, making it difficult to scale.

## **Reward Hacking:**

Since RLHF depends on defining a reward function based on human preferences, there is a risk of **reward hacking**, where the agent learns to optimize for the reward in unintended ways. The agent may exploit loopholes in the reward system rather than truly achieving the desired behavior, leading to suboptimal or even harmful outcomes.

## **Feedback Quality and Consistency:**

The effectiveness of RLHF depends heavily on the quality of the human feedback. Humans can be inconsistent or biased in their feedback, which can confuse the learning process and lead to suboptimal policy learning. Training models from noisy or inconsistent feedback makes it harder to converge.



# Being more direct - DPO

**Direct Preference Optimization (DPO)** is a newer approach that attempts to directly optimize an agent's behavior based on preference data, addressing some limitations of RLHF.

Recall in RLHF, the human preference model is used to guide the learning process, through PPO. However, RLHF can suffer from certain inefficiencies, and DPO aims to streamline the process.

## How DPO Works:

- Instead of the traditional reinforcement learning setup where feedback is used to compute a reward signal and optimize the agent via a reward function, **DPO uses preference data directly**.
- Given pairs of trajectories or action sequences, DPO determines which one is preferred based on human feedback.
- The goal is to adjust the policy so that it consistently produces the preferred actions, without the need for complex reward estimation or value function modeling.
- This reformulates the problem as a classification task

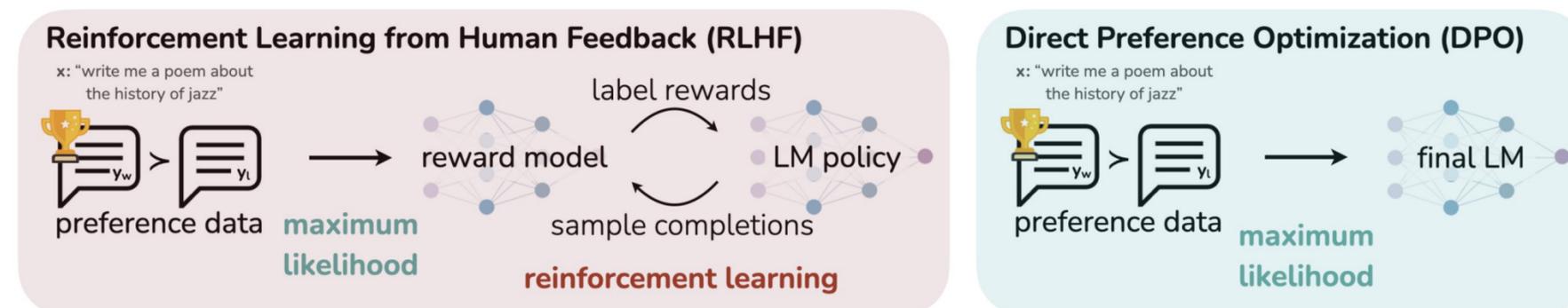


Figure 1: **DPO optimizes for human preferences while avoiding reinforcement learning.** Existing methods for fine-tuning language models with human feedback first fit a reward model to a dataset of prompts and human preferences over pairs of responses, and then use RL to find a policy that maximizes the learned reward. In contrast, DPO directly optimizes for the policy best satisfying the preferences with a simple classification objective, fitting an *implicit* reward model whose corresponding optimal policy can be extracted in closed form.

# DPO > RLHF

**DPO Addresses the Issues present in RLHF by the following:**

**Direct Preference Learning:** DPO directly optimizes preferences, which can reduce the reliance on complex reward functions and the need for iterative RL fine-tuning phases.

**Simpler Objective:** By directly optimizing the policy to reflect human preferences (based on a binary comparison between trajectories), DPO simplifies the optimization process, potentially leading to **faster convergence** and fewer computational resources.

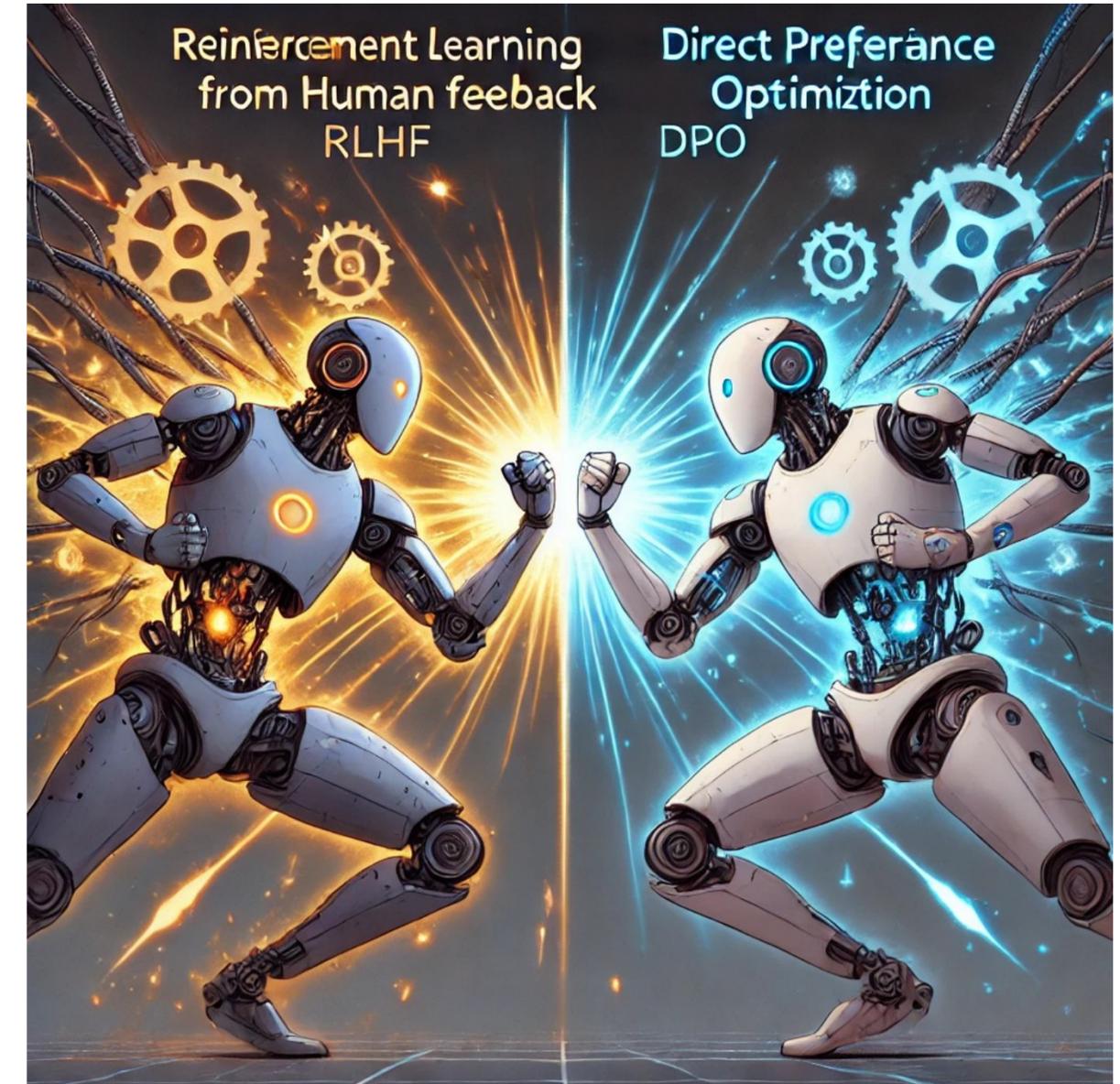
**Improved Feedback Utilization:** DPO's direct approach can make better use of human feedback by focusing on clear preferences rather than indirect reward signals, reducing the chances of reward hacking and inefficiencies in learning.



# Choosing DPO or RLHF

While DPO may seem like the obvious choice, it is also important to note that more classical RLHF still has some advantages:

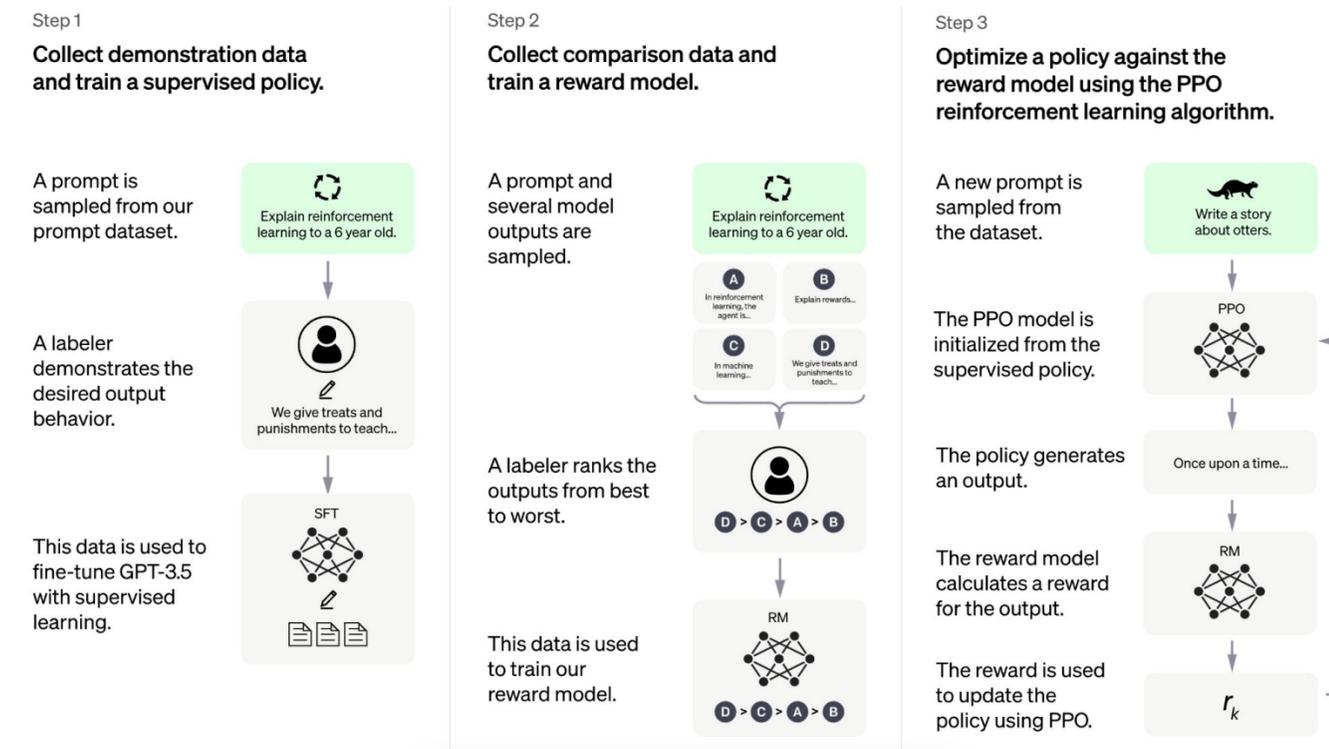
- **Flexibility in defining rewards:** RLHF allows for more complex and nuanced reward structures, which can be beneficial for tasks requiring precise control over the LLM's output. This flexibility can be crucial in specific situations where DPO's simpler approach might not be sufficient.
- **Handling diverse feedback formats:** RLHF can handle various forms of human feedback, including numerical ratings, textual corrections, and implicit feedback. DPO currently primarily relies on binary preferences, which may limit its applicability in scenarios requiring more nuanced feedback.
- **Handling large datasets:** RLHF can be more efficient in handling massive datasets, especially when combined with distributed training techniques. This can be advantageous for tasks where fine-tuning needs to be performed on massive amounts of data.



# Wrap Up

## RLHF and Alignment of Chat models

- Today we introduced the concept of Reinforcement Learning with Human Feedback
  - Reinforcement Learning in general was revised, focusing on policy-based methods
  - We introduced the RLHF process and the steps involved
  - Limitations of RLHF and Direct Policy Optimization (DPO) was also covered
- 





Thank you!