Reinforcement Learning for Large Language Models

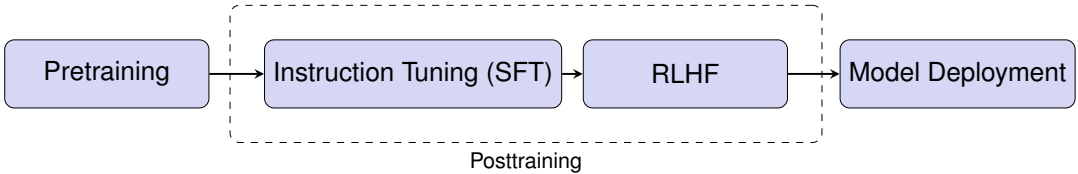
Instructor: Tong Zhang

Part 2: Reinforcement Learning from Human Feedback (RLHF)

Outline

- **Background**
- **Reinforcement Learning from Human Feedback (RLHF)**: Ziegler et al. (2019). "Fine-tuning language models from human preferences". *arXiv preprint arXiv:1909.08593*
- **InstructGPT**: Ouyang et al. (2022). "Training language models to follow instructions with human feedback". *Advances in Neural Information Processing Systems*
- **DPO and explicit reward-model policy learning**: Rafailov et al. (2024). "Direct preference optimization: Your language model is secretly a reward model". *Advances in Neural Information Processing Systems* , Lin et al. (2024). "On the Limited Generalization Capability of the Implicit Reward Model Induced by Direct Preference Optimization". *arXiv:2409.03650*

Recall Pretraining to Posttraining Pipeline



- **Pretraining:** Train on large text datasets.
- **Instruction Tuning (SFT):** Fine-tune on human instructions.
- **Reinforcement Learning from Human Feedback (RLHF):** Further refine the model with preference data.
- **Deployment:** Use the model for real-world applications.

Limits of Pretraining Alone

Pretraining teaches a model to continue text, but that is not the same as following user intent.

- A pretrained LLM learns likely continuations from Internet text.
- Users want outputs that are helpful, truthful, and safe.
- As a result, pretrained models may hallucinate, ignore instructions, or produce inappropriate answers.

This mismatch motivates posttraining methods such as SFT and RLHF.

Two Types of Human Data in RLHF

Data format for SFT: prompt + target answer

- **Prompt:** What is a black hole?
- **Answer:** A region where gravity is so strong that not even light can escape.

Typical data format for RLHF: preference ranking

- **Prompt:** What is the capital of France?
- **Answer 1 (chosen/preferred):** The capital is Paris.
- **Answer 2 (rejected):** It is known for the Eiffel Tower.

Pairwise preference data is the most common format, though absolute ratings are also possible.

From Preference Data to Reward Model

Preference data only tells us which answer is better:

$$a_1 \succ a_2 \mid x \quad \Longleftrightarrow \quad a_1 \text{ is preferred over } a_2 \text{ for prompt } x$$

We convert this into a scalar quality signal by learning a reward model

$$r_\theta(x, a),$$

which measures the quality of response a for prompt x. That is, preferred answers receive larger scores:

$$a_1 \succ a_2 \mid x \quad \Longleftrightarrow \quad r_\theta(x, a_1) > r_\theta(x, a_2).$$

Main Steps in RLHF

Human feedback tells us which outputs people prefer, but it does not directly tell us how to optimize the LLM.

Two main steps

1. **Reward learning:** learn a reward function from preference data.
2. **Policy learning:** optimize the LLM (called the policy in RL) so that it produces high-reward answers.

This is the basic procedure described in Ziegler et al. (2019). "Fine-tuning language models from human preferences". *arXiv preprint arXiv:1909.08593* .

Bradley–Terry Model

Given prompt x, answer $a_1 \succ a_2$ means that a_1 is preferred over a_2.

The Bradley–Terry model assigns the preference probability

$$\Pr(a_1 \succ a_2 \mid x) = \frac{e^{r_\theta(x,a_1)}}{e^{r_\theta(x,a_1)} + e^{r_\theta(x,a_2)}},$$

where $r_\theta(x, a)$ is the reward model.

This turns reward learning into a binary classification problem:

$$\text{either} \quad a_1 \succ a_2 \quad \text{or} \quad a_2 \succ a_1 \quad (\text{ignoring } a_1 = a_2).$$

One can also extend this idea to ranking one out of K answers using softmax, as in Ziegler et al. (2019). "Fine-tuning language models from human preferences". *arXiv preprint arXiv:1909.08593* .

Reward Model Learning

Given preference training ata

$$\{(x, a_1 \succ a_2)\},$$

where a_1 is the preferred answer and a_2 is the less preferred answer, we fit the reward model $r_\theta(x, a)$.

A standard loss is the logistic regression loss (binary cross-entropy loss):

$$\min_\theta \sum_{(x, a_1 \succ a_2)} \ln\left(1 + e^{r_\theta(x,a_2) - r_\theta(x,a_1)}\right).$$

The binary cross-entropy loss encourages the preferred answer to receive a larger reward.

Learning a Policy from the Reward Model

A learned policy π is a fine-tuned LLM that generates response a from prompt x.

What is a good policy?

- A good policy should produce high quality responses, measured by high reward.
- Moreover, as a generative model, it should also produce diverse responses (when the sampling temperature is nonzero).

To learn a good policy, we should balance reward versus diversity.

Reward versus Diversity

Alignment quality of an LLM policy π may be measured by the average reward

$$\mathbb{E}_x \mathbb{E}_{a\sim\pi(\cdot|x)} r(x,a),$$

where x is the prompt and a is the answer generated by π.

Diversity of π may be measured by the entropy

$$-\mathbb{E}_x \mathbb{E}_{a\sim\pi(\cdot|x)} \ln \pi(a|x).$$

For both quantities, larger numbers are better.

Reward versus Diversity During RLHF Training

The model π_{ref} (reference model) is the LLM before RLHF training:

- retains broad pretrained knowledge and produces diverse responses.
- However, it is not well aligned with human preferences:

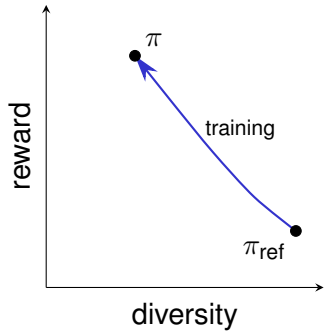
$$\mathbb{E}_x \mathbb{E}_{a \sim \pi_{\text{ref}}(\cdot|x)} r(x, a) \qquad \text{is relatively small.}$$

During RLHF training: $\pi_{\text{ref}} \to \pi$

- The average reward increases:

$$\mathbb{E}_x \mathbb{E}_{a \sim \pi(\cdot|x)} r(x, a).$$

- The policy π moves away from π_{ref}.
- As optimization focuses on high-reward responses, diversity often decreases.



Diversity Control with KL Regularization

A key idea in RLHF training is to keep π close to π_{ref}:

- preserve output diversity,
- preserve pretrained knowledge.

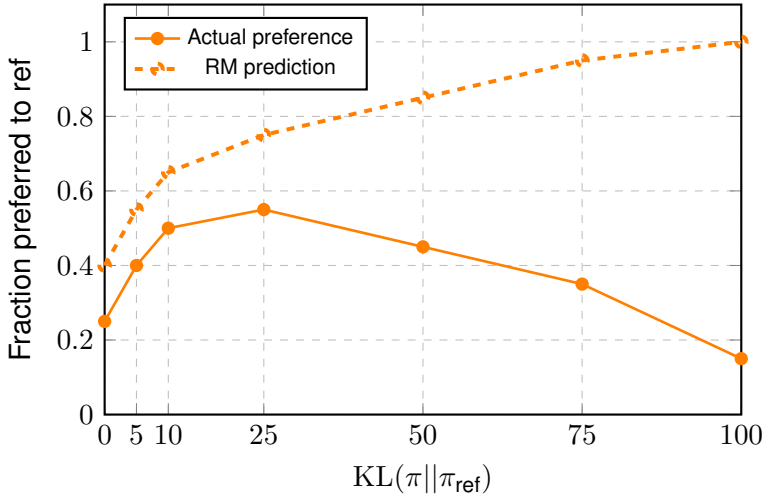
A common control is KL regularization:

$$\mathrm{KL}(\pi \| \pi_{\text{ref}}) = \mathbb{E}_x \mathbb{E}_{a \sim \pi(\cdot|x)} \ln \frac{\pi(a|x)}{\pi_{\text{ref}}(a|x)}.$$

Small KL-divergence helps preserve diversity and pretrained behavior.

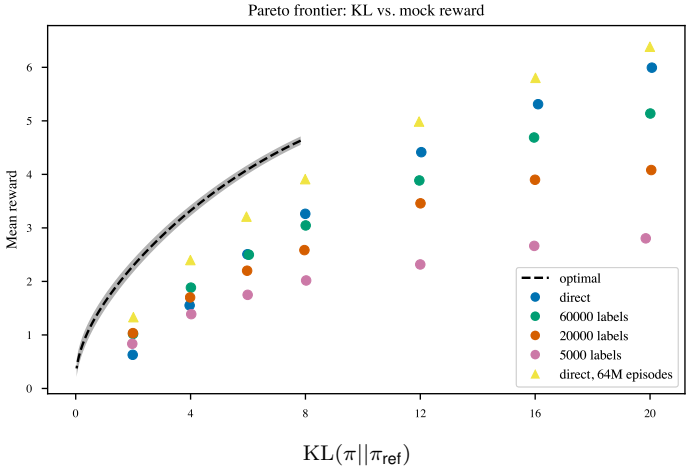
Remark: KL-regularization is natural because it keeps the new policy close to the reference model while discouraging collapse to a narrow set of answers.

Losing Diversity Can Lead to Reward Hacking



Typical phenomenon in RLHF training (Ziegler et al. (2019). "Fine-tuning language models from human preferences". *arXiv preprint arXiv:1909.08593*).

RLHF Learns a Trade-off between Quality and Diversity



- Quality: mean reward
- Loss of diversity: KL-divergence

Source: Ziegler et al. (2019). "Fine-tuning language models from human preferences". *arXiv preprint arXiv:1909.08593*

A Common RLHF Objective

A common formulation is to optimize

$$\pi = \arg\max_{\pi} \mathbb{E}_x \mathbb{E}_{a\sim\pi(\cdot|x)} \left[r(x,a) - \beta \ln \frac{\pi(a|x)}{\pi_{\text{ref}}(a|x)} \right],$$

where $\beta > 0$ is a regularization parameter.
This objective balances:

- **Generation quality**

$$\mathbb{E}_x \mathbb{E}_{a\sim\pi(\cdot|x)}[r(x,a)]$$

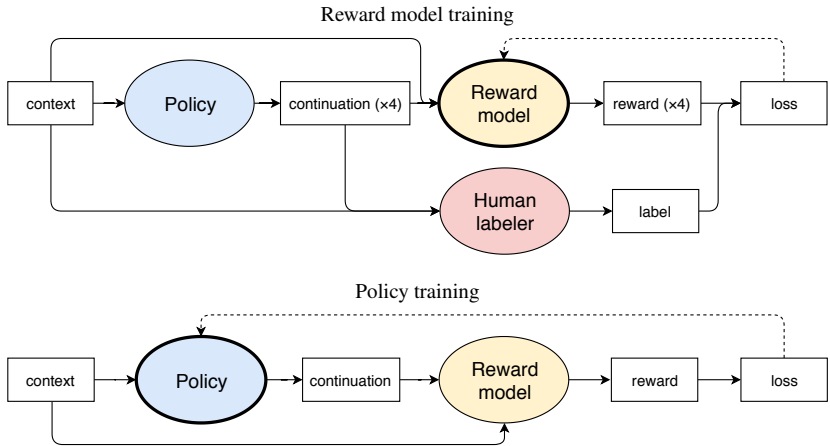
- **Closeness to the reference model**

$$\mathrm{KL}(\pi||\pi_{\text{ref}}) = \underbrace{\mathbb{E}_{x,a}[\ln \pi(a|x)]}_{\text{negative-entropy}} - \mathbb{E}_{x,a}[\ln \pi_{\text{ref}}(a|x)].$$

Remark: minimize KL tends to encourage larger entropy $\Rightarrow$ more diversity.

Source: Ziegler et al. (2019). "Fine-tuning language models from human preferences". *arXiv preprint arXiv:1909.08593*

The Complete RLHF Process



The paper uses human preference over one out of four answers, instead of only two answers.

Source: Ziegler et al. (2019). "Fine-tuning language models from human preferences". *arXiv preprint arXiv:1909.08593*

From GPT-3 to InstructGPT

One early milestone toward chat-based LLMs is InstructGPT, which applies SFT and RLHF to GPT-3 Ouyang et al. (2022). "Training language models to follow instructions with human feedback". *Advances in Neural Information Processing Systems* .

We start with pretrained GPT-3:

- Supervised fine-tuning (SFT) using human demonstrations
- Reward model learning using human preference data
- Policy learning (RLHF) using PPO with KL penalty to stay close to SFT

InstructGPT Training Pipeline

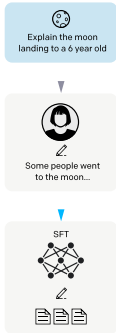
Step 1

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 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.

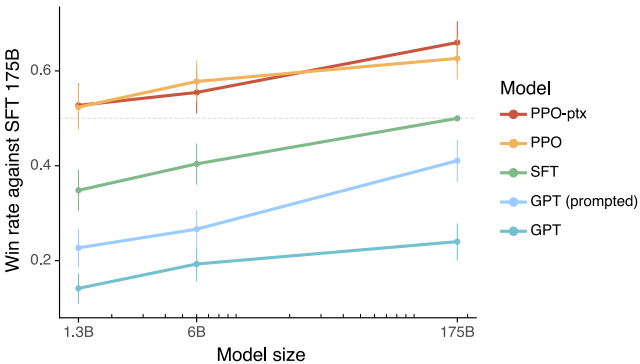
This data is used to train our reward model.

Step 3

Optimize a policy against the reward model using reinforcement learning.

Source: Ouyang et al. (2022). "Training language models to follow instructions with human feedback". *Advances in Neural Information Processing Systems*

InstructGPT Results



PPO-ptx: PPO training mixed with pretraining data.

Source: Ouyang et al. (2022). "Training language models to follow instructions with human feedback". *Advances in Neural Information Processing Systems*

- Human evaluation on prompts drawn from the OpenAI API distribution.
- Metric: fraction of outputs preferred over the **175B SFT baseline**.
- Error bars show **95% confidence intervals**.
- RLHF models (**PPO** and **PPO-ptx**) significantly outperform GPT-3 baselines.
- Notably, the **1.3B PPO-ptx model** is preferred to **175B GPT-3**.

Implications of IntructGPT

- Alignment can matter more than raw model scale,
- Posttraining strongly changes user experience,
- Large model + prompting alone is not enough to match RLHF.
- SFT can be further improved via RLHF.

An aligned small model can be more useful than a much larger base model.

InstructGPT Dataset Sizes for SFT, Reward Model (RM), and PPO

Table: Dataset sizes, in number of prompts.

SFT Data			**RM Data**			**PPO Data**		
split	source	size	split	source	size	split	source	size
train	labeler	11,295	train	labeler	6,623	train	customer	31,144
train	customer	1,430	train	customer	26,584	valid	customer	16,185
valid	labeler	1,550	valid	labeler	3,488			
valid	customer	103	valid	customer	14,399			

- SFT uses many more labeler-written prompts than customer prompts.
- Each RM prompt ranks multiple outputs, so the effective training size is much larger.
- PPO uses only customer prompts, reflecting the target deployment distribution.

RLHF Policy Learning Training Objective

A common policy learning formulation is to optimize

$$\pi = \arg\max_{\pi} Q(\pi), \qquad Q(\pi) = \mathbb{E}_x \mathbb{E}_{a\sim\pi(\cdot|x)} \left[r(x,a) - \beta \ln \frac{\pi(a|x)}{\pi_{\text{ref}}(a|x)} \right], \tag{1}$$

where $\beta > 0$ is a regularization parameter, and π_{ref} is the reference model.

This objective balances:

- **Generation quality**

$$\mathbb{E}_x \mathbb{E}_{a\sim\pi(\cdot|x)}[r(x,a)]$$

- **Generation diversity** (closeness to the reference model)

$$\mathrm{KL}(\pi||\pi_{\text{ref}})$$

Ziegler et al. (2019). "Fine-tuning language models from human preferences". *arXiv preprint arXiv:1909.08593*

Optimal Solution

The optimal policy solving ( ) is

$$\pi_*(a|x) = \frac{1}{Z(x)}\pi_{\text{ref}}(a|x)e^{r(x,a)/\beta}, \quad Z(x) = \sum_a \pi_{\text{ref}}(a|x)e^{r(x,a)/\beta}. \tag{2}$$

Proof. We have

$$\begin{aligned}
Q(\pi) =& \mathbb{E}_x \mathbb{E}_{a\sim\pi(a|x)}\left[r(x,a) - \beta \ln \frac{\pi(a|x)}{\pi_{\text{ref}}(a|x)}\right] \\
=& -\beta \mathbb{E}_x \mathbb{E}_{a\sim\pi(a|x)}\left[\ln \frac{\pi(a|x)}{\pi_{\text{ref}}(a|x)e^{r(x,a)/\beta}}\right] \\
=& -\beta \mathbb{E}_x \mathbb{E}_{a\sim\pi(a|x)}\left[\ln \frac{\pi(a|x)}{\pi_*(a|x)Z(x)}\right] = \underbrace{\beta \mathbb{E}_x[\ln Z(x)]}_{\text{constant}} - \beta \mathrm{KL}(\pi||\pi_*).
\end{aligned}$$

Therefore

$$Q(\pi_*) - Q(\pi) = \beta \mathrm{KL}(\pi||\pi_*) \geq 0.$$

Implicit Reward

Rearranging the optimal policy formula, the reward model is implied by the optimal policy:

$$r(x,a) = \underbrace{\beta \ln \frac{\pi_*(a|x)}{\pi_{\text{ref}}(a|x)}}_{\text{implicit reward}} + \beta \ln Z(x). \tag{3}$$

The term $\beta \ln Z(x)$ does not affect preference ranking because it doesn't depend on a.

For a pair (a_1, a_2) under prompt x, $\beta \ln Z(x)$ term cancels:

$$r(x,a_2) - r(x,a_1) = \beta \left[\ln \frac{\pi_*(a_2|x)}{\pi_{\text{ref}}(a_2|x)} - \ln \frac{\pi_*(a_1|x)}{\pi_{\text{ref}}(a_1|x)} \right].$$

Direct Preference Optimization: Logistic Form

We now recall the Bradley-Terry (BT) preference model:

$$\Pr(a_1 \succ a_2 \mid x) = \frac{1}{1 + \exp(r(x, a_2) - r(x, a_1))}.$$

Plug in the implicit reward formula:

$$\Pr(a_1 \succ a_2 \mid x) = \frac{1}{1 + \exp\left(\beta \left[\ln \frac{\pi_*(a_2|x)}{\pi_{\text{ref}}(a_2|x)} - \ln \frac{\pi_*(a_1|x)}{\pi_{\text{ref}}(a_1|x)}\right]\right)}.$$

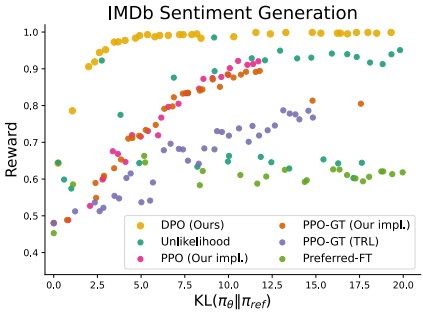
Direct Preference Optimization: Training Objective

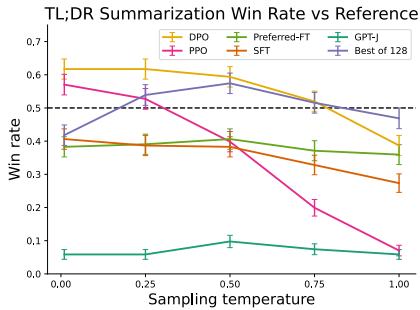
Replace unknown π_* with trainable policy π_θ and learn θ using maximum log-likelihood on preference data:

$$\max_\theta \mathbb{E}_{(x,a_1\succ a_2)} \ln \frac{1}{1+\exp\left(\beta \ln \frac{\pi_\theta(a_2|x)}{\pi_{\text{ref}}(a_2|x)} - \beta \ln \frac{\pi_\theta(a_1|x)}{\pi_{\text{ref}}(a_1|x)}\right)}.$$

- This formulation is DPO: direct preference optimization.
- It uses the relationship between optimal policy and implicit reward to optimize policy directly from preference data.
- No need for reward-model learning.

DPO Experiments





- Left: frontier of expected reward versus KL to the reference policy.
- Right: TL;DR GPT-4 evaluated summarization win rates versus human-written summaries.

Source: Rafailov et al. (2024). "Direct preference optimization: Your language model is secretly a reward model". *Advances in Neural Information Processing Systems*

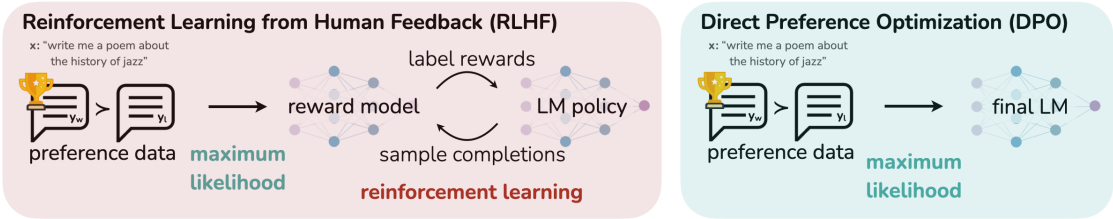
DPO Summary

DPO bypasses explicit reward-model training by using the closed-form relationship between KL-optimal policy and implicit reward.

Practical consequences:

- Simpler pipeline: preference data $\rightarrow$ policy optimization via implicit reward formula.
- Supervised logistic objective, often easier to tune.
- Reference model π_{ref} constrains final model behavior through log-ratios.

Do We Need Reward Model?



Source: Rafailov et al. (2024). "Direct preference optimization: Your language model is secretly a reward model". *Advances in Neural Information Processing Systems*

Two approaches:

- Left: Standard RLHF practice: construct reward model, then train policy.
- Right: DPO bypasses reward model and trains policy from preference data directly. DPO policy induces an implicit reward model.

Do we need reward modeling?

It is still beneficial to train reward model

It is easier to train a high quality reward model than a high quality policy.

Recall that in InstructGPT, a 6B reward model is used to learn a 175B GPT-3 policy model.

Conceptual: judgment is easier than generation.

Practical consequence:

- Implicit DPO reward can be inferior on out-of-distribution data.
- Consequently, DPO policy can be inferior on out-of-distribution data.

Lin et al. (2024). "On the Limited Generalization Capability of the Implicit Reward Model Induced by Direct Preference Optimization". *arXiv:2409.03650*

Iterative DPO

Iterative DPO uses an explicit reward model $r(x, a)$ as a judge while the current policy model generates new preference data over several iterations.

Iterate:

1. Update policy π based on the current preference data.
2. Use the updated policy π to generate K responses for each prompt.
3. Rate responses using the explicit reward model $r(x, a)$, and select the highest- and lowest-scoring responses to construct pairwise preference data.

This procedure learns from an explicit reward model that can be more robust to out-of-distribution data.

Lin et al. (2024). "On the Limited Generalization Capability of the Implicit Reward Model Induced by Direct Preference Optimization". *arXiv:2409.03650*

Xiong et al. (2024). "Iterative Preference Learning from Human Feedback: Bridging Theory and Practice for RLHF under KL-Constraint". *International Conference on Machine Learning*

Yuan et al. (2024). "Self-rewarding language models". *Forty-first International Conference on Machine Learning*

Experiments with Iterative DPO

RM	Base	Iter 0	Iter 1	Iter 2
EXRM	5.88	10.09	13.56	13.86
DPORM	5.88	10.09	11.02	11.13

Table: Iterative DPO length-controlled win-rate (%) over GPT-4 on AlpacaEval.

- EXRM uses an explicit reward model to judge newly generated policy outputs.
- DPORM uses the implicit reward model induced by DPO.
- The explicit reward model gives stronger iterative improvement in this setting.

Source: Lin et al. (2024). "On the Limited Generalization Capability of the Implicit Reward Model Induced by Direct Preference Optimization". *arXiv:2409.03650*

Summary

- After pretraining, we need to align LLM's behavior using:
  - human demonstrations (instruction tuning)
  - human preference data (RLHF: reward model + policy learning)
- A reward model converts preference rankings into a scalar optimization signal.
- KL regularization helps balance quality (reward) and diversity.
- InstructGPT shows that alignment can matter more than raw model size.
- DPO shows how preference optimization can be written as a direct supervised objective through an implicit reward model.
- Explicit reward models can still be valuable for judging new policy outputs, especially in iterative or out-of-distribution settings.

References

- **RLHF Original Paper**: Ziegler et al. (2019). "Fine-tuning language models from human preferences". *arXiv preprint arXiv:1909.08593*
- **InstructGPT**: Ouyang et al. (2022). "Training language models to follow instructions with human feedback". *Advances in Neural Information Processing Systems*
- **DPO**: Rafailov et al. (2024). "Direct preference optimization: Your language model is secretly a reward model". *Advances in Neural Information Processing Systems*
- **Explicit RM for iterative DPO**: Lin et al. (2024). "On the Limited Generalization Capability of the Implicit Reward Model Induced by Direct Preference Optimization". *arXiv:2409.03650* , Xiong et al. (2024). "Iterative Preference Learning from Human Feedback: Bridging Theory and Practice for RLHF under KL-Constraint". *International Conference on Machine Learning* , Yuan et al. (2024). "Self-rewarding language models". *Forty-first International Conference on Machine Learning*