Direct Preference Optimization

Your Language Model is Secretly a Reward Model

Rebecca El Chidiac Ruben Ifrah

based on Rafailov et al., Stanford University, 2023

17/11/2025

Motivation: why preference-based alignment?

Large Language Models are powerful but not inherently aligned.

- LLMs are trained to **predict the next token**, not to follow human values.
- Raw models often produce:
  - incorrect or misleading answers
  - harmful or unsafe content
  - biased, toxic, or unethical outputs
  - overly verbose or unhelpful responses

Preference-based alignment teaches models what humans consider:

- good vs bad answers
- safe vs harmful behaviors
- helpfulness, harmlessness, honesty

From likelihood to preferences

- **Pretraining:** learn $p_\theta(\text{next token} \mid \text{context})$ on web-scale data.
- This gives:
  - fluent, knowledgeable models
  - but **no guarantee** of being helpful / safe.
- **Alignment:** add a layer that says

"among all plausible answers, which ones do humans actually prefer?"

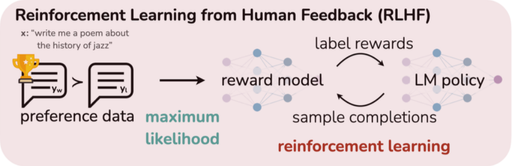
Outline

Reinforcement Learning with Human Feedback (RLHF)

Goal: Train an LLM on a given task while staying aligned with human preferences.

RLHF usually proceeds in **3 steps**:

1. Supervised Fine-Tuning (SFT)
2. Training a Reward Model
3. Policy Optimization with RL + KL regularization



Context

- *Christiano et al., Deep Reinforcement Learning from Human Preferences* (2017) ;
- before 2017 : only supervised fine-tuning ;
- first application of RLHF around 2019 / 2020 ;
- democratize around 2021 / 2022.

Step 1: Supervised Fine-Tuning

Data: high-quality human-written answers.

- Collect pairs (x, y) where:
  - x: user prompt
  - y: good, human-written answer
- Fine-tune a **base LLM** by maximum likelihood:

$$\max_{\theta_{\mathsf{SFT}}} \mathbb{E}_{(x,y)}\Big[\log \pi_{\theta_{\mathsf{SFT}}}(y \mid x)\Big].$$

- Result: a model π^{SFT} that imitates good responses, but:
  - it has never seen **comparisons** between answers,
  - it may still hallucinate or be unsafe.

In short

- SFT $\rightarrow$ output THIS answer
- RLHF $\rightarrow$ make the output closer to this answer THAN to that answer

Step 2: training a reward model

- Collect **human preference data**:
  For each prompt x, sample two answers $(y_1, y_2) \sim \pi_{\text{SFT}}(y \mid x)$.
  Humans choose the preferred one: $\mathcal{D} = \left\{(x^{(i)}, y^{(i)}_{winner}, y^{(i)}_{loser})\right\}_{i=1}^{N}$

Bradley–Terry Preference Model

We assume human preferences satisfy:

$$p^*(y_w \succ y_l \mid x) = \frac{\exp(r^*(x, y_w))}{\exp(r^*(x, y_w)) + \exp(r^*(x, y_l))} = \sigma\big(r^*(x, y_w) - r^*(x, y_l)\big),$$

where $r^*(x, y)$ is an unknown scalar **reward**.

- Train a **reward model** $r_\phi(x, y)$ via maximum likelihood:

$$\mathcal{L}_R(r_\phi, \mathcal{D}) = -\mathbb{E}_{(x, y_{winner}, y_{loser}) \sim \mathcal{D}} \left[\log \sigma\big(r_\phi(x, y_{winner}) - r_\phi(x, y_{loser})\big)\right].$$

Step 3: policy optimization (RL stage)

- We want a policy π_θ that:
  - gets high reward from r_ϕ,
  - does not drift too far from a reference π_{ref} (usually SFT).
- Standard RLHF objective:

$$\max_{\pi_\theta} \mathbb{E}_{x,\, y\sim\pi_\theta(y|x)}[r_\phi(x,y)] - \beta\, \mathbb{D}_{\mathrm{KL}}(\pi_\theta(y \mid x) \,\|\, \pi_{\mathrm{ref}}(y \mid x)) \,.$$

Practically:

- PPO*-like RL: actor-critic RL, not detailed here...
- Training is **complex and unstable**: reward hacking, collapse, sensitive to hyperparameters and implementation details.

*PPO = Proximal Preference Optimization

Comments

- Not really RL (in the 2017 paper "*reinforcement learning–flavored supervised learning*")
- still RL framework: PPO, reward maximization formulation of the problem

RLHF: intuition and pain points

in short

- Treat the reward model as a **critic**.
- Use RL to push the policy toward high-reward answers while keeping it close to the SFT model $\rightarrow$ result: improved policy π_θ compared to π_{SFT}

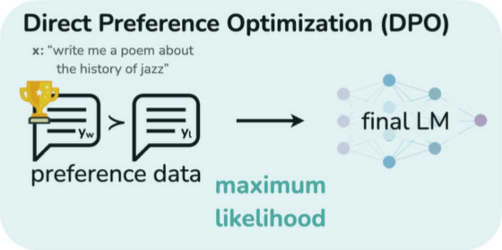
Pain points in practice

- Two models to train: reward + policy.
- RL training requires:
  - rollout generation during training,
  - careful tuning of PPO, clipping, value loss, etc.,
  - tricks manage exploration / exploitation tradeoff and to avoid reward exploitation.
- Expensive and engineering-heavy, especially at LLM scale.

Question: Can we get the benefits of RLHF *without* the RL stage?

Direct Preference Optimization (DPO)

- **Goal:** Train directly from human preference data **without explicit RL**.
- DPO derives a **maximum-likelihood** objective whose optimum is the RLHF-optimal policy.
- No separate reward model:
- Looks like **supervised learning** on preference pairs.



Context

- *Rafailov et al., DPO: Your Language Model is Secretly a Reward Model* (2023) ;
- rapidly adopted in open-source LLM training starting 2023 ;
- becoming standard for preference optimization in 2024 / 2025.

DPO: intuition introduction before the math

- In RLHF, we:
  1. learn a reward model r_ϕ,
  2. then run RL to find a policy that maximizes its expected reward.
- But human feedback is **pairwise preferences**: (x, y_w, y_l) : y_w preferred to y_l.
- DPO:
  - instead of turning this into a scalar reward first,
  - we **directly** adjust the policy so that $\pi_\theta(y_w \mid x)$ gets higher than $\pi_\theta(y_l \mid x)$, in a principled way that matches the RLHF optimum.

we can think of it as: applying logistic regression on "which answer humans prefer", skipping the reward derivation.

Assumptions for DPO–RLHF equivalence

We consider the RLHF objective (for fixed prompt x and given reward $r(x, \cdot)$):

$$J(\pi; r) = \mathbb{E}_{y \sim \pi(y|x)}[r(x, y)] - \beta\, \mathbb{D}_{\mathrm{KL}}(\pi(y \mid x) \,\|\, \pi_{\mathrm{ref}}(y \mid x)), \quad \begin{cases} \pi_r = \mathrm{argmax}_\pi J(\pi; r) \\ \pi^* = \pi_{r^*} = \mathrm{argmax}_\pi J(\pi, r^*) \end{cases}$$

DPO relies on three key assumptions:

1. **Bradley–Terry preference model** for human choices:

$$p^*(y_w \succ y_l \mid x) = \frac{e^{r^*(x, y_w)}}{e^{r^*(x, y_w)} + e^{r^*(x, y_l)}}.$$

2. **KL-regularized RLHF objective** as above, with temperature β.
3. **Shared support:** $\pi_{\mathrm{ref}}(y \mid x) > 0$ wherever $\pi^*(y \mid x) > 0$ (verified $\to$ Softmax).

(1) and (2) already assumed for RLHF, (3) is the new assumption. Under these , the RLHF-optimal policy has a **closed form**, which we use to derive DPO.

Step 1: RLHF optimal policy as exponential tilt

Fixing x, let's maximize $J(\pi)$ over $\pi(\cdot \mid x)$[1] : $\Big($ recall $\mathbb{D}_{\mathsf{KL}}(P \,\|\, Q) = \sum_y P(x) \log \frac{P(y)}{Q(y)}\Big)$:

$$J(\pi; r) = \underbrace{\sum_y \pi(y \mid x)\, r(x, y)}_{\mathbb{E}_{y\sim\pi(y|x)}[r(x,y)]} - \beta \underbrace{\sum_y \pi(y \mid x) \log \frac{\pi(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)}}_{\mathbb{D}_{\mathrm{KL}}\left(\pi(y|x) \,\|\, \pi_{\mathrm{ref}}(y|x)\right)}$$

- Strictly concave functional over $\pi(\cdot \mid x)$.
- Using Lagrange multipliers for the constraint $\sum_y \pi(y \mid x) = 1$, the optimum satisfies:

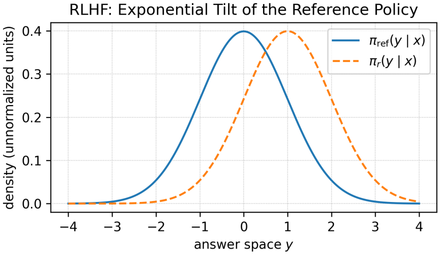
$$\pi_r(y \mid x) = \frac{1}{Z(x)}\, \pi_{\mathrm{ref}}(y \mid x) \exp\left(\frac{1}{\beta}\, r(x, y)\right),$$

where $Z(x)$ is a normalizing constant.

[1] Full derivation provided in Annex A–D at the end of the presentation.

$$\pi_r(y \mid x) = \frac{1}{Z(x)}\, \pi_{\mathrm{ref}}(y \mid x)\, \exp\left(\frac{1}{\beta}\, r(x, y)\right),$$

Interpretation: RLHF acts as an *exponential tilt* of the reference policy by the reward.



Taking logs: $$\log \pi_r(y \mid x) = \log \pi_{\mathrm{ref}}(y \mid x) + \tfrac{1}{\beta} r(x, y) - \log Z(x).$$

Solve for the reward: $$r(x, y) = \beta \log\left(\frac{\pi_r(y|x)}{\pi_{\mathrm{ref}}(y|x)}\right) + \beta \log Z(x) = \beta \log\left(Z(x)\frac{\pi_r(y|x)}{\pi_{\mathrm{ref}}(y|x)}\right)$$

Key point: up to an additive term $\beta \log Z(x)$ (which does not depend on y), the reward is just a *log-ratio* between the optimal policy and the reference.

Step 3: plug into Bradley–Terry preferences

Recall Bradley–Terry: $p^*(y_w \succ y_l \mid x) = \frac{\exp(r^*(x,y_w))}{\exp(r^*(x,y_w))+\exp(r^*(x,y_l))}$

We plug $r^*(x,y) = \log\left(Z(x)\frac{\pi^*(y|x)}{\pi_{\text{ref}}(y|x)}\right)^{\beta}$ (formula works for any reward):

$$p^*(y_w \succ y_l \mid x) = \frac{Z(x)^{\beta} e^{\beta \log \frac{\pi^*(y_w|x)}{\pi_{\text{ref}}(y_w|x)}}}{Z(x)^{\beta} e^{\beta \log \frac{\pi^*(y_w|x)}{\pi_{\text{ref}}(y_w|x)}} + Z(x)^{\beta} e^{\beta \log \frac{\pi^*(y_l|x)}{\pi_{\text{ref}}(y_l|x)}}}$$

The partition function $Z(x)^{\beta}$ cancels between numerator and denominator, giving:

$$p^*(y_w \succ y_l \mid x) = \sigma\left(\beta \log \frac{\pi^*(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi^*(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right),$$

We expressed human preferences purely in terms of the optimal policy and the reference, without needing to use $r^*(x,y)$

Step 4: From RLHF optimum to DPO objective

- We do not know π^*, but we **know** that human preference data was generated according to:

$$p^*(y_w \succ y_l \mid x) = \sigma\left(\beta \log \frac{\pi^*(y_w \mid x)}{\pi_{\mathrm{ref}}(y_w \mid x)} - \beta \log \frac{\pi^*(y_l \mid x)}{\pi_{\mathrm{ref}}(y_l \mid x)}\right).$$

- $\rightarrow$ fit a parametric policy π_θ by MLE on observed preferences, using this form but replacing π^* by π_θ.

Thus we consider the negative log-likelihood:

$$\boxed{\mathcal{L}_{\mathrm{DPO}}(\pi_\theta; \pi_{\mathrm{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\mathrm{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\mathrm{ref}}(y_l|x)}\right)\right]}$$

Observation

- exactly a logistic regression objective applied to the log-probability difference $\log \pi_\theta(y_w|x) - \log \pi_\theta(y_l|x)$.
- DPO reduces preference optimization to *supervised learning*.

DPO loss:

$$\mathcal{L}_{\mathrm{DPO}}(\pi_\theta; \pi_{\mathrm{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\mathrm{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\mathrm{ref}}(y_l \mid x)}\right)\right].$$

Recall the RLHF reward model loss:

$$\mathcal{L}_R(\hat{r}_\theta, \mathcal{D}) = -\mathbb{E}_{(x,y_{winner},y_{loser})\sim\mathcal{D}}\left[\log \sigma\big(\hat{r}_\theta(x, y_{winner}) - \hat{r}_\theta(x, y_{loser})\big)\right].$$

$\rightarrow$ we can define the **implicit reward difference**: $\hat{r}_\theta(x,y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{\mathrm{ref}}(y|x)}$.

Then the gradient is[2]:

$$\nabla_\theta \mathcal{L}_{\mathrm{DPO}} = -\mathbb{E}\Big[\log \sigma\left(\beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\mathrm{ref}}(y_l \mid x)} - \beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\mathrm{ref}}(y_w \mid x)}\right) \big(\nabla_\theta \log \pi_\theta(y_w \mid x) - \nabla_\theta \log \pi_\theta(y_l \mid x$$

[2] Full derivation of the gradient provided in annex F

DPO training loop: looks like supervised learning

For each mini-batch of preference pairs (x, y_w, y_l):

1. Compute log-probabilities under π_θ: $\log \pi_\theta(y_w \mid x), \quad \log \pi_\theta(y_l \mid x)$
   $\rightarrow$ these are just probability of sequences y_w / y_l given prompt x
2. Compute log-probabilities under π_{ref} (frozen).
3. Form the DPO loss $\mathcal{L}_{\text{DPO}}$.
4. Backpropagate and update θ with standard optimizers (Adam, etc.).

- No explicit reward model.
- No PPO, no value function, no on-policy rollouts during training.
- Just log-likelihoods and a cross-entropy-like loss on preference pairs.
  $\rightarrow$ much closer to classical supervised learning with a custom loss function

How do we evaluate aligned LLMs?

- We are not optimizing perplexity but **human satisfaction**.
- Typical evaluation for RLHF / DPO:
  - choose a set of prompts x,
  - generate responses from:
    - baseline model (SFT or RLHF),
    - candidate model (DPO).
  - ask humans (or a strong judge model) which response they prefer.

Key metric: win-rate

$$\text{win-rate}(\text{DPO vs baseline}) = \mathbb{P}\big(\text{DPO answer is preferred}\big).$$

Optionally also measure:

- distance to reference: $\mathrm{KL}(\pi_\theta \,\|\, \pi_{\mathrm{ref}})$,
- reward according to a reward model (if available),
- standard task metrics (e.g. ROUGE for summarization).

Human evaluation vs GPT-4-as-a-judge

- Human evaluation is the **gold standard** but expensive.
- The DPO paper also uses a strong model (GPT-4) as an **automatic judge**:
  - the judge is given the prompt and two answers (A/B),
  - asked which one is better along criteria like helpfulness, honesty, harmlessness.

Advantages:

- cheaper, scalable to thousands of comparisons,
- allows fast iteration during research.

Caveats:

- judge model has its own biases,
- evaluation quality depends on prompt design and model quality.

DPO vs RLHF: empirical results

Across tasks such as:

- dialogue / helpfulness and harmlessness,
- summarization (e.g., TL;DR),
- instruction-following tasks,

the paper reports:

Main empirical findings

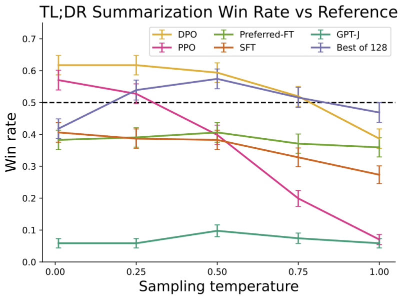
- DPO achieves **similar or higher win-rates** than PPO-based RLHF.
- For a given win-rate, DPO often stays **closer** to the reference model (lower KL).
- Training is more **stable** and simpler to scale.

Takeaway: in many settings, you can replace the RLHF stage by DPO and keep (or slightly improve) alignment quality with a much simpler pipeline.

Role of the temperature β in DPO

$$\text{Recall } \hat{r}_\theta(x, y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}.$$

- **Small β (cold):**
  - implicit rewards are small,
  - policy stays very close to π_{ref},
  - high safety / low drift, but may underfit preferences.
- **Large β (hot):**
  - larger reward differences,
  - policy moves more aggressively away from π_{ref},
  - can better fit preferences but risks over-optimization.



using GPT-4 as evaluator

In practice, β is tuned to balance:

$$\text{win-rate} \quad \text{vs} \quad \mathrm{KL}(\pi_\theta \,\|\, \pi_{\text{ref}}).$$

Comparison: DPO vs RLHF (qualitative)

Criterion	RLHF	DPO
Training pipeline complexity	High	Low
Need for reward model	Yes	No
Stability	Sensitive / unstable	Generally stable
Compute requirements	Very high	Low–moderate
Alignment quality	High (SOTA)	Comparable, often similar
Hyperparameter tuning	Heavy	Lighter (mostly β)
Implementation difficulty	Hard (RL)	Easy (log-loss)
Scalability	Good	Excellent

When RLHF is still useful

- RLHF keeps an **explicit reward model**:
  - can be reused to evaluate or monitor other policies,
  - allows reward shaping, multi-objective trade-offs (e.g. safety vs helpfulness),
  - fine-grained control over behaviors.
- RL methods can ingest **off-policy data**
- Today, DPO alongside RLHF for big labs
  Pretraining $\rightarrow$ SFT $\rightarrow$ DPO-like preference tuning $\rightarrow$ RLHF (PPO) $\rightarrow$ Safety tuning

	OpenAI, Google, ...	**Open-source**	**Academic Research**
RLHF	Yes	No	No
DPO	Emerging	Yes	Yes

In short

DPO is an extremely attractive default, but RLHF remains useful when you want strong control via an explicit reward function.

Conclusion

What is DPO?

A method that **directly** fits a policy to human preference data by maximizing the likelihood of observed choices under a model derived from the RLHF optimum.

Why does it matter?

- Same underlying assumptions as RLHF (Bradley–Terry + KL-regularized RL).
- Avoids the RL stage: simpler, more stable, cheaper.
- Empirically competitive with PPO-based RLHF on multiple tasks.

Thanks you :)

Questions ?

Annex A - why fixing prompt x?

RLHF objective (per prompt):

$$J_x(\pi) = \mathbb{E}_{y\sim\pi(\cdot|x)}[r(x,y)] - \beta\,\mathrm{KL}\big(\pi(\cdot|x)\,\|\,\pi_{\mathrm{ref}}(\cdot|x)\big).$$

- The global RLHF objective is:

$$\mathbb{E}_{x\sim \mathsf{data}}[J_x(\pi)].$$

- prompts x are independent in the expectation.
- Therefore, for derivation, we can **fix** x and optimize

$$\max_{\pi(\cdot|x)} J_x(\pi).$$

- This simplifies notation and the optimization: we treat $\pi(\cdot|x)$ as a distribution over the answer space for a single prompt.

Annex B - $J(\pi)$ concave over π?

$$J(\pi) = \underbrace{\sum_y \pi(y)\, r(y)}_{\text{linear}} - \beta \underbrace{\sum_y \pi(y) \log \frac{\pi(y)}{\pi_{\mathrm{ref}}(y)}}_{\mathrm{KL}(\pi \| \pi_{\mathrm{ref}})}$$

- view $\pi(y)$ as a long vector
- The first term is **linear** in π (hence concave).
- KL divergence $\mathrm{KL}(\pi \| \pi_{\mathrm{ref}})$ **strictly convex** in π.
- Therefore $-\beta\, \mathrm{KL}(\pi \| \pi_{\mathrm{ref}})$ is **strictly concave**.

Conclusion

Linear + strictly concave = strictly concave

- $J(\pi)$ has a single global maximum.
- Any stationary point from Lagrange multipliers is **the unique optimal policy**.

Annex C — Lagrangian for the RLHF Objective

We solve:

$$\max_{\pi} \sum_{y} \pi(y)\, r(y) - \beta \sum_{y} \pi(y) \log \frac{\pi(y)}{\pi_{\mathrm{ref}}(y)} \quad \text{s.t.} \quad \sum_{y} \pi(y) = 1.$$

Lagrangian:

$$\mathcal{L}(\pi, \lambda) = \sum_{y} \pi(y) r(y) - \beta \sum_{y} \pi(y)\big(\log \pi(y) - \log \pi_{\mathrm{ref}}(y)\big) + \lambda\Big(\sum_{y} \pi(y) - 1\Big).$$

Derivative w.r.t. $\pi(y)$:

$$\frac{\partial \mathcal{L}}{\partial \pi(y)} = r(y) - \beta\big(\log \pi(y) + 1 - \log \pi_{\mathrm{ref}}(y)\big) + \lambda = 0.$$

Solve for $\log \pi(y)$:

$$\log \pi^*(y) = \log \pi_{\mathrm{ref}}(y) + \frac{1}{\beta} r(y) + C(x),$$

where $C(x)$ is independent of y.

Annex D — Optimal RLHF Policy

Exponentiate the previous expression: $\log \pi^*(y) = \log \pi_{\mathrm{ref}}(y) + \frac{1}{\beta} r(y) + C(x)$

$$\pi^*(y|x) = \frac{1}{Z(x)} \pi_{\mathrm{ref}}(y|x) \exp\left(\frac{1}{\beta} r(x, y)\right),$$

where $Z(x)$ is the normalizing constant.

Interpretation

The optimal RLHF policy is an **exponential tilt** of the reference model:

$$\pi^* \propto \pi_{\mathrm{ref}} \cdot e^{r/\beta}.$$

This is the unique maximizer because $J(\pi)$ is concave.

Annex E — Reward in Terms of Policy Ratios

Starting from:

$$\pi^*(y|x) = \frac{1}{Z(x)}\pi_{\mathrm{ref}}(y|x)\exp\left(\frac{1}{\beta}r(x,y)\right),$$

take log:

$$\log \pi^*(y|x) = \log \pi_{\mathrm{ref}}(y|x) + \frac{1}{\beta}r(x,y) - \log Z(x).$$

Rearrange:

$$r(x,y) = \beta\left(\log \frac{\pi^*(y|x)}{\pi_{\mathrm{ref}}(y|x)}\right) + \beta \log Z(x).$$

Key point

$\beta \log Z(x)$ does **not** depend on y. So reward differences depend only on:

$$\log \frac{\pi^*}{}.$$

Annex F — Plug into Bradley–Terry

Bradley–Terry preference model:

$$p(y_w \succ y_l|x) = \frac{\exp(r(x, y_w))}{\exp(r(x, y_w)) + \exp(r(x, y_l))}.$$

Plug the expression of $r(x, y)$:

$$p(y_w \succ y_l|x) = \sigma\left(\beta \log \frac{\pi^*(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi^*(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right).$$

Interpretation

Human preference probabilities can be written entirely in terms of optimal RLHF policy ratios vs the reference policy.

Annex G — DPO Objective via Maximum Likelihood

Assume preference data is generated by π^* as above, and replace π^* with a parametric π_θ.

$$p_\theta(y_w \succ y_l|x) = \sigma\left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\mathrm{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\mathrm{ref}}(y_l|x)}\right).$$

Negative log-likelihood over dataset:

$$\boxed{\mathcal{L}_{\mathrm{DPO}} = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\mathrm{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\mathrm{ref}}(y_l|x)}\right)\right].}$$

Conclusion

DPO directly optimizes a supervised loss whose optimum equals the RLHF-optimal policy π^*, but without any RL or reward model.

Annex F - derivation of the DPO gradient (1)

We start from the DPO loss:

$$\mathcal{L}_{\mathrm{DPO}} = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma(\hat{r}_\theta(x,y_w) - \hat{r}_\theta(x,y_l))\right],$$

with implicit reward:

$$\hat{r}_\theta(x,y) = \beta\log\frac{\pi_\theta(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)}.$$

Let

$$\Delta_\theta = \hat{r}_\theta(x,y_w) - \hat{r}_\theta(x,y_l).$$

Step 1 - Derivative of the negative log-sigmoid

$$\frac{d}{d\Delta_\theta}\big[-\log\sigma(\Delta_\theta)\big] = 1 - \sigma(\Delta_\theta) = \sigma(-\Delta_\theta).$$

Thus

$$\nabla_\theta\mathcal{L}_{\mathrm{DPO}} = \mathbb{E}\Big[\sigma\big(\hat{r}_\theta(x,y_l) - \hat{r}_\theta(x,y_w)\big)\ \nabla_\theta(\Delta_\theta)\Big].$$

Annex F - derivation of the DPO gradient (2)

Step 2 - Expand $\nabla_\theta(\Delta_\theta)$

$$\Delta_\theta = \beta[\log \pi_\theta(y_w \mid x) - \log \pi_\theta(y_l \mid x)] - \beta[\log \pi_{\mathrm{ref}}(y_w \mid x) - \log \pi_{\mathrm{ref}}(y_l \mid x)]\,.$$

Since π_{ref} is fixed (no θ dependence):

$$\nabla_\theta \Delta_\theta = \beta\left[\nabla_\theta \log \pi_\theta(y_w \mid x) - \nabla_\theta \log \pi_\theta(y_l \mid x)\right].$$

Step 3 - Final expression

$$\nabla_\theta \mathcal{L}_{\mathrm{DPO}} = -\mathbb{E}_{(x,y_w,y_l)}\Big[\sigma\big(\hat{r}_\theta(x, y_l) - \hat{r}_\theta(x, y_w)\big)\left(\nabla_\theta \log \pi_\theta(y_w \mid x) - \nabla_\theta \log \pi_\theta(y_l \mid x)\right)\Big].$$

Interpretation

DPO increases $\log \pi_\theta(y_w \mid x)$ and decreases $\log \pi_\theta(y_l \mid x)$, weighted by how strongly the current policy violates the observed preference.