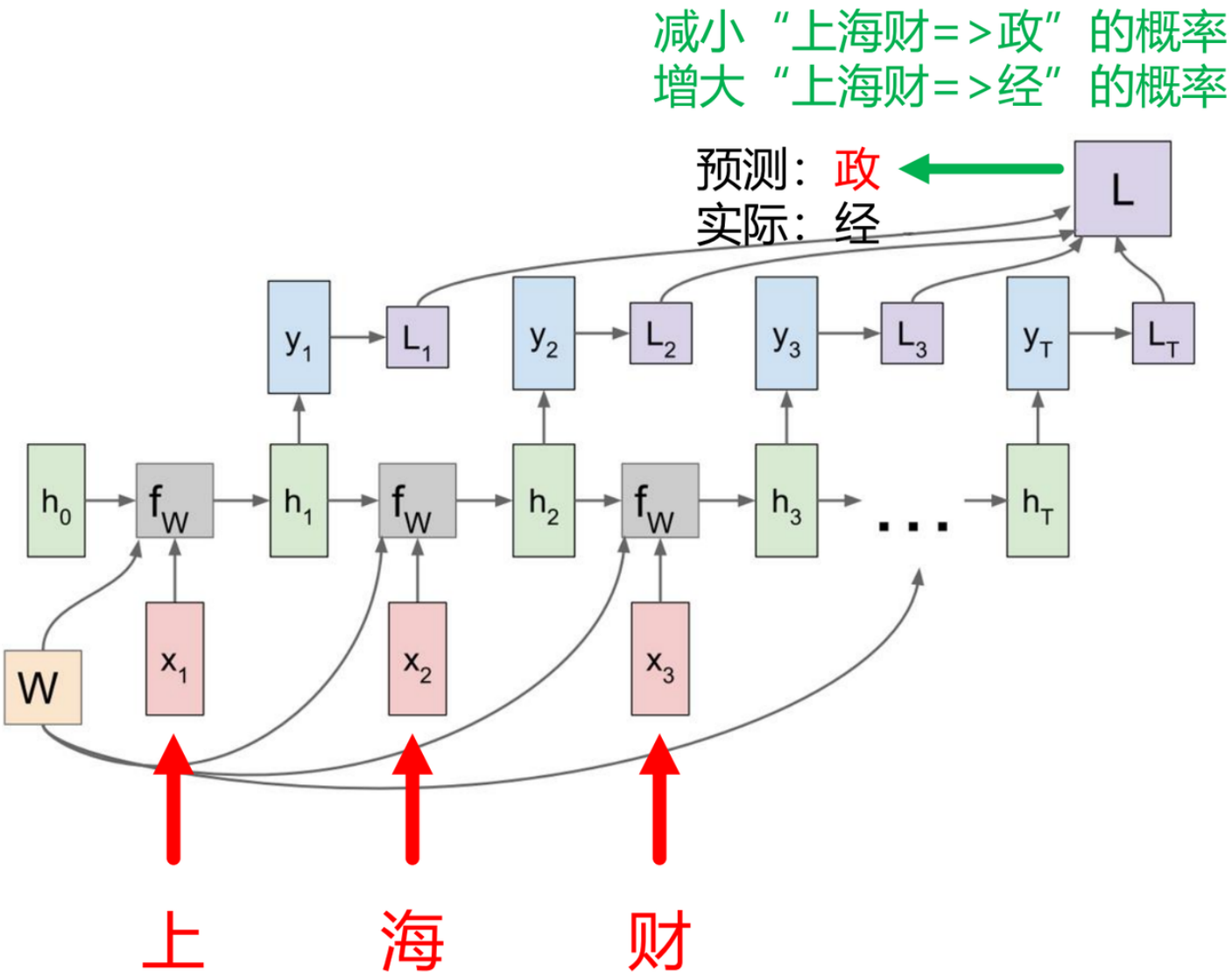
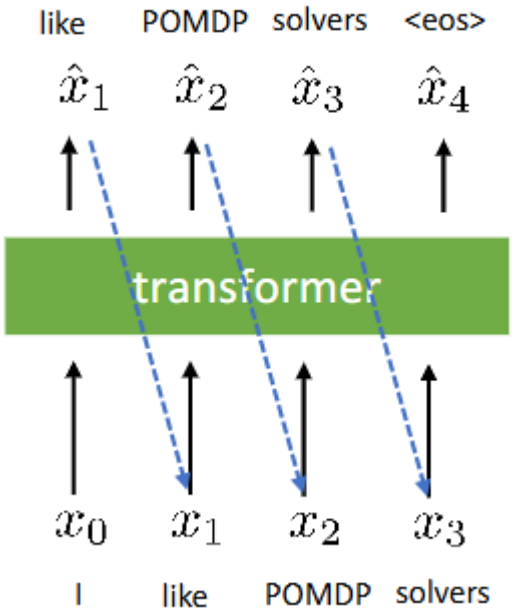
RLHF/RLAIF/DPO/ORPO

语言模型与人类偏好对齐

语言模型



Decoder - Autoregressive LM 自回归语言模型



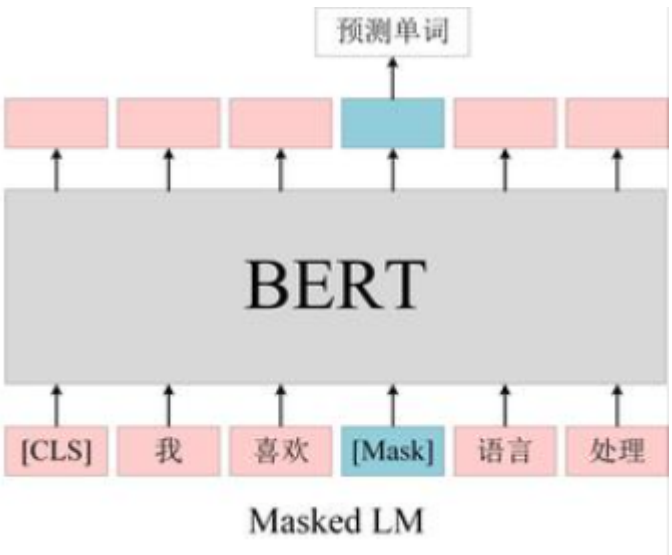
目标：根据上文预测下一个词元。

损失函数：给定序列 $x = (x_1, x_2, ..., x_T)$，最大化序列的似然函数:

$$L_{AR}(\theta) = -\frac{1}{T}\sum_{t=1}^{T} logP(x_t|x_{<t};\theta)$$

其中 $x_{<t} = x_1, \ldots, x_{t-1}$，$\theta$ 是模型参数。

Encoder - Masked LM



- 目标：根据上下文预测被随机掩码的词元。
- 损失函数：令 m 为一个掩码向量，其中 $m_t = 1$ 表示位置 t 的词元被掩码。$M = \{t : m_t = 1\}$ 是被掩码位置的集合。损失函数只在这些被掩码的位置上计算。

$$L_{MLM}(\theta) = -\frac{1}{||M||}\sum_{t\in M} logP(x_t|x_{t\notin M};\theta)$$

其中 $x_{t \notin M}$ 表示未被掩码的上下文词元。

有监督微调 (Supervised Fine-Tuning, SFT)

{"prompt": "解释一下量子计算的基本原理。",

"response": "量子计算是一种利用量子力学原理（如叠加和纠缠）来处理信息的新型计算模式..."}

对于一对 `(prompt, response)`，将其拼接成一个完整的文本序列：`[prompt] [response]`，之后使用Decoder架构训练，使用掩码技巧，只对于回答计算最大似然。

$$L_{SFT}(\theta) = -\frac{1}{T}\sum_{t=1}^{T} logP(y_t|x, y_{<t}; \theta)$$

人类偏好对齐 (Alignment)

Making language models bigger does not inherently make them better at following a user’s intent. For example, large language models can generate outputs that are untruthful, toxic, or simply not helpful to the user. In other words, these models are not aligned with their users.

人类偏好的3H准则

- Helpfulness - 回答是否对于对话者有帮助，最难的一点
- Harmlessness — 回答是否无害，不能包括煽动情绪/歧视等毒性内容
- Honest — 知之为知之，不知为不知，不能捏造事实/出现幻觉

为什么需要RLHF?

SFT之后的模型虽然有一定改进，但存在局限性，通常不能直接用，需要借助RL进一步训练优化：

SFT模型的局限性

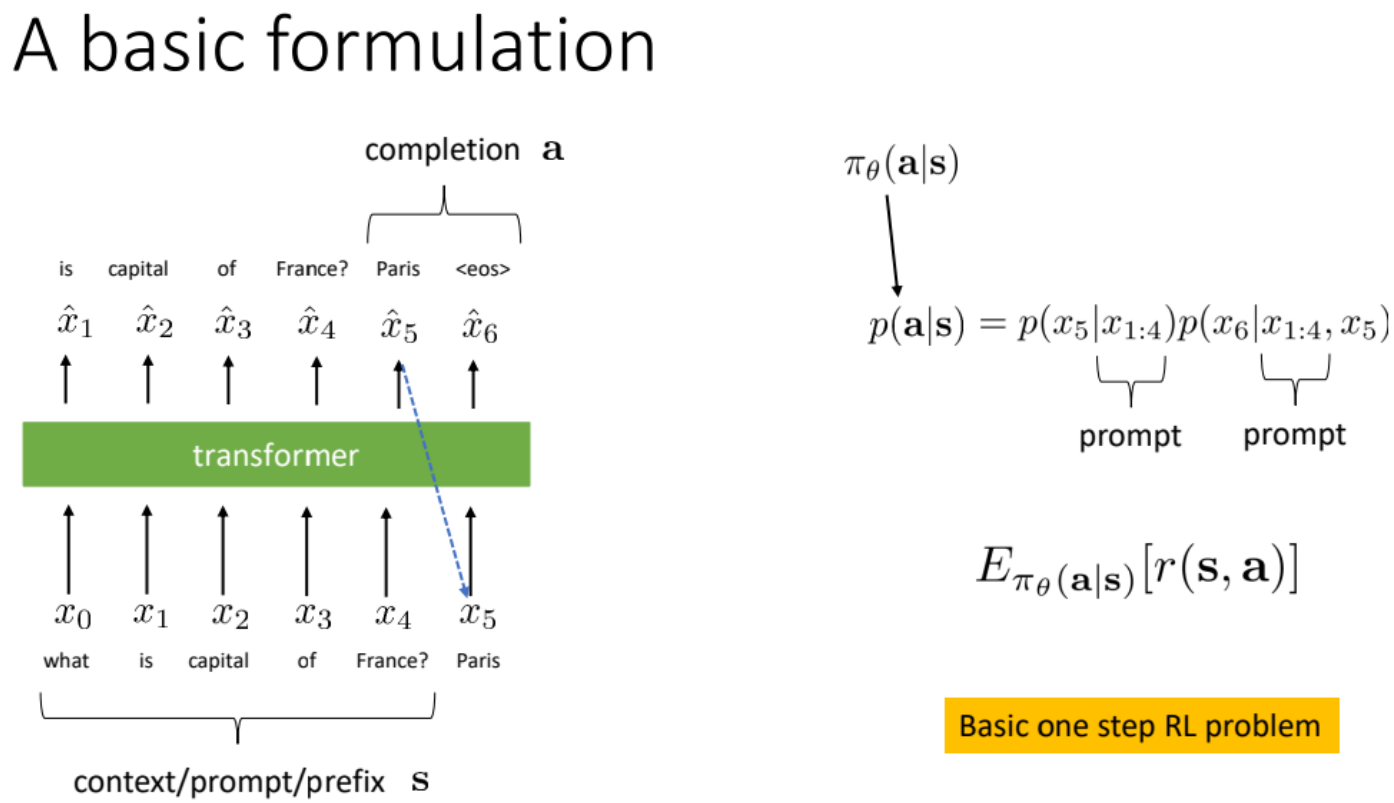
- 缺乏负反馈机制：SFT通过提供带有明确输入-输出对的数据集来调整模型，只能告知模型哪些token是“正确”的，无法直接指出哪些是不应该出现的错误token，可能产生不符合期望的结果。
- 单向注意力结构：由于transformer架构特点，SFT中的每个token只依赖于前文信息，忽略了后续内容的影响，不能从整体上优化生成内容。
- 过度自信泛化：当模型接收过多正面示例时，可能会过于相信某些模式，忽略其他可能性，在处理复杂指令或涉及伦理道德等问题时表现不佳。

RLHF的作用

- 允许负反馈：RLHF可以通过惩罚机制教会模型避免生成不合适的回应。例如用户对某个答案不满意，可降低相关状态价值，促使模型未来做出更好选择。
- 全局视角优化：RLHF考虑整个句子级别的奖励分配，让每个token的概率调整都基于整体上下文，而非孤立片段，使生成的文本更连贯、合理。
- 增强对话安全性和可控性：RLHF可通过对生成内容实施严格筛选标准，如过滤敏感词汇，提高模型的安全性和可靠性，使其输出更符合人类价值观和社会规范。

基于人类反馈的强化学习（Reinforcement Learning from Human Feedback, RLHF）

RL语境下的语言模型



Language models and policy gradients

$$\nabla_\theta \log \pi_\theta(\mathbf{a}|\mathbf{s}) = \nabla_\theta \log p(x_5|x_{1:4}) + \nabla_\theta \log p(x_6|x_{1:4}, x_5)$$

$$\nabla_\theta E_{\pi_\theta(\mathbf{a}|\mathbf{s})}[r(\mathbf{s},\mathbf{a})] = E_{\pi_\theta(\mathbf{a}|\mathbf{s})}[\nabla_\theta \log \pi_\theta(\mathbf{a}|\mathbf{s}) r(\mathbf{s},\mathbf{a})]$$

samples from $\pi_\theta(\mathbf{a}|\mathbf{s})$

REINFORCE-style estimator

$$\approx \frac{1}{N}\sum_i \nabla_\theta \log \pi_\theta(\mathbf{a}_i|\mathbf{s}) r(\mathbf{s},\mathbf{a}_i)$$

samples from $\bar{\pi}(\mathbf{a}|\mathbf{s})$

importance-weighted estimator (e.g., PPO)

$$\approx \frac{1}{N}\sum_i \frac{\pi_\theta(\mathbf{a}_i|\mathbf{s})}{\bar{\pi}(\mathbf{a}_i|\mathbf{s})} \nabla_\theta \log \pi_\theta(\mathbf{a}_i|\mathbf{s}) r(\mathbf{s},\mathbf{a}_i)$$

Why might we prefer this?

Language models and policy gradients

$$\nabla_\theta E_{\pi_\theta(\mathbf{a}|\mathbf{s})}[r(\mathbf{s},\mathbf{a})] \approx \underbrace{\frac{1}{N}\sum_i \frac{\pi_\theta(\mathbf{a}_i|\mathbf{s})}{\bar{\pi}(\mathbf{a}_i|\mathbf{s})} \nabla_\theta \log \pi_\theta(\mathbf{a}_i|\mathbf{s}) r(\mathbf{s},\mathbf{a}_i)}_{\hat{\nabla}(\theta, \bar{\pi}, \{\mathbf{a}_i\})}$$

importance-weighted estimator (e.g., PPO)

1. sample batch $\mathcal{B} = \{\mathbf{a}_i\}$, $\mathbf{a}_i \sim \pi_\theta(\mathbf{a}|\mathbf{s})$
2. evaluate $r(\mathbf{s},\mathbf{a}_i)$ for each $\mathbf{a}_i \in \mathcal{B}$
3. $\bar{\pi} \leftarrow \pi_\theta$
4. sample *minibatch* $\mathcal{M} \subset \mathcal{B}$
5. $\theta \leftarrow \theta + \alpha \hat{\nabla}(\theta, \bar{\pi}, \mathcal{M})$

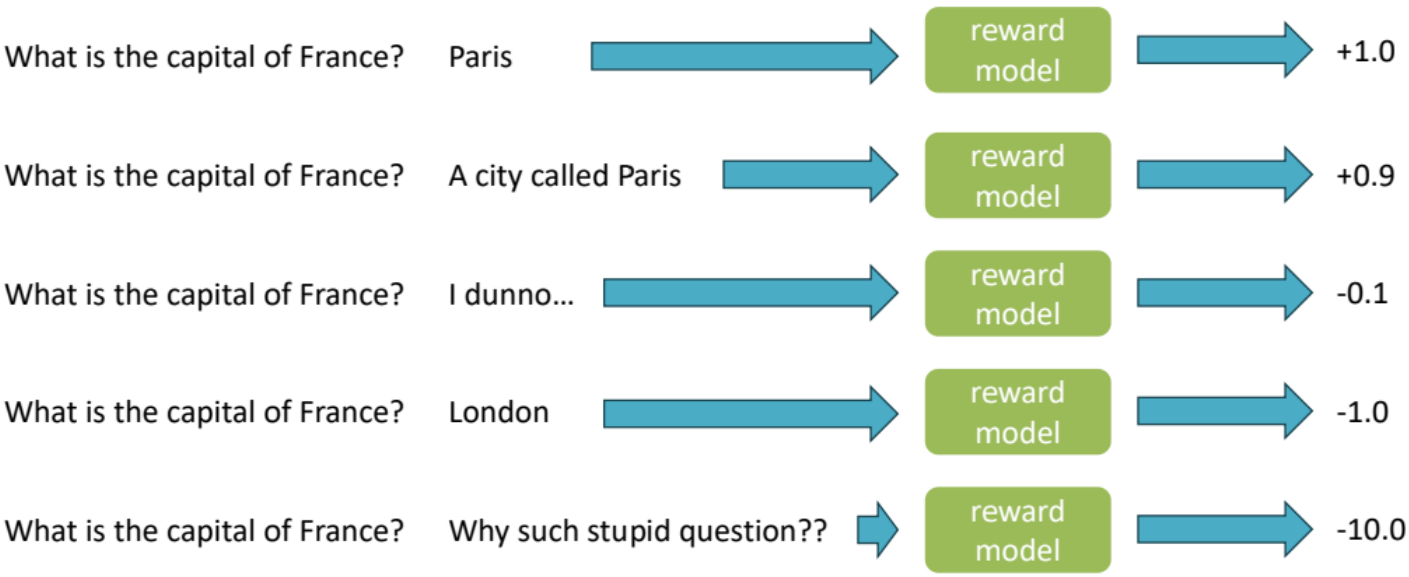
what is this?

repeat K times

奖励模型 （Reward Model, RM）

Learned rewards

What if $r(\mathbf{s}, \mathbf{a})$ is itself a neural network?

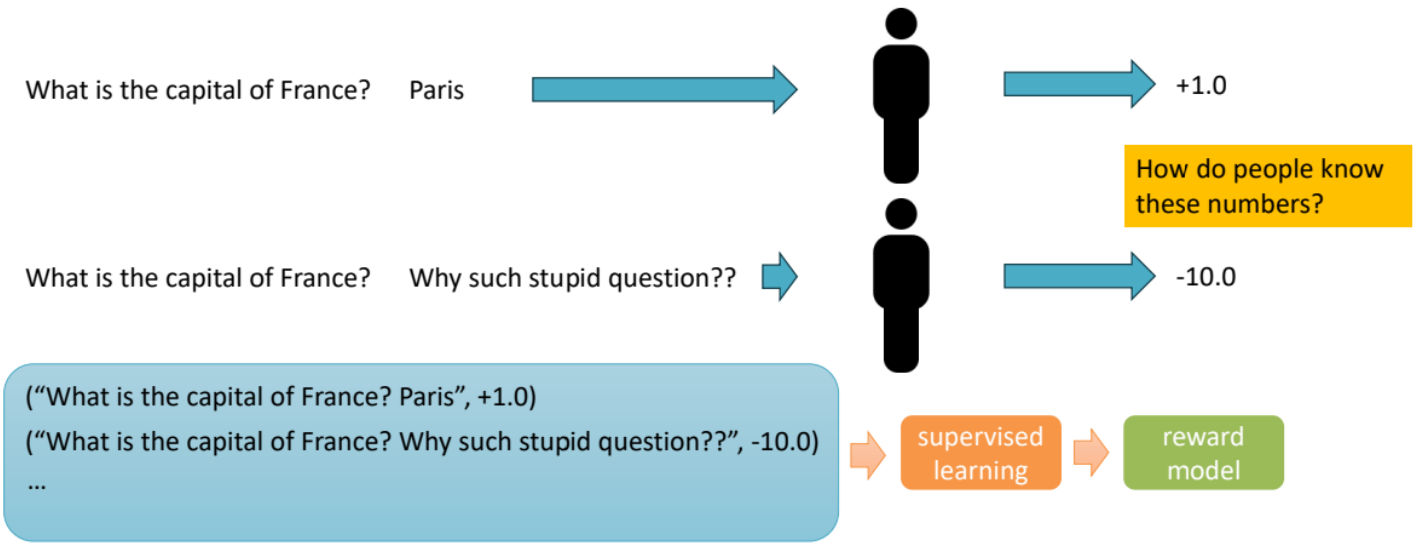


奖励模型要处理语言模型可能输出的各种不定长文本，本身也应该是一个语言模型。

$r(x, y) \to \mathbb{R}$

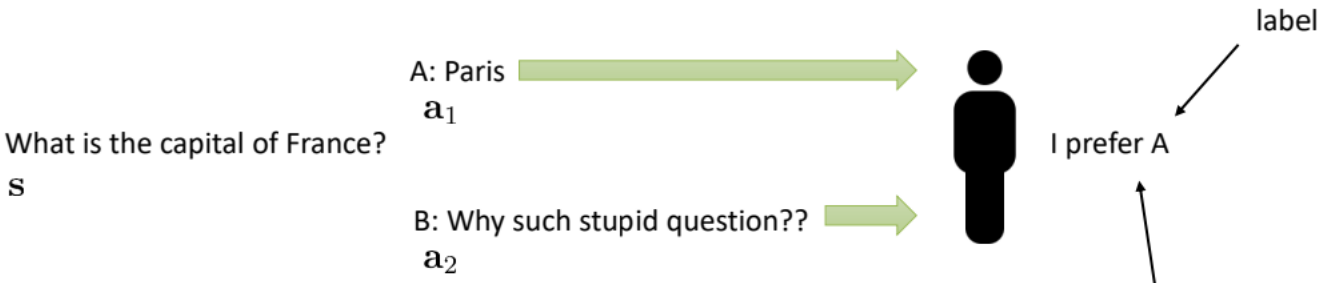
RL from human feedback

How do we train the reward model $r_\psi(\mathbf{s}, \mathbf{a})$?



训练方法是有监督学习，但是对于人类来讲，对于一段文字输出提供一个标量的打分较困难（老师批阅作文需要经过大量训练），但是对于两个回答给出排名较容易。

Rewards from preferences



Prediction: given $(\mathbf{s}, \mathbf{a}_1, \mathbf{a}_2)$, how likely is a person to prefer $\mathbf{a}_1$ over $\mathbf{a}_2$?

maximize likelihood w.r.t. params ψ of r

$$p(\mathbf{a}_1 \text{ preferred over } \mathbf{a}_2) = \frac{\exp(r_\psi(\mathbf{s}, \mathbf{a}_1))}{\exp(r_\psi(\mathbf{s}, \mathbf{a}_1)) + \exp(r_\psi(\mathbf{s}, \mathbf{a}_2))}$$

Bradley-Terry Model

$$p(y_w \succ y_l|x) = \frac{exp\{r_\psi(x, y_w)\}}{exp\{r_\psi(x, y_w)\} + exp\{r_\psi(x, y_l)\}} \tag{1}$$

$$= \frac{1}{1 + exp\{r_\psi(x, y_l) - r_\psi(x, y_w)\}} \tag{2}$$

$$= \frac{1}{1 + exp\{-[r_\psi(x, y_w) - r_\psi(x, y_l)]\}} \tag{3}$$

$$= \sigma(r_\psi(x, y_w) - r_\psi(x, y_l)) \tag{4}$$

这是一个design choice，假设人类头脑中真的有一个 $r_\psi(x, y)$ 进行打分，并通过 $\sigma(r_\psi(x, y_w) - r_\psi(x, y_l))$ 奖励函数差的sigmoid来反映做出偏好的概率。

数据格式与损失函数

$\mathcal{D} = (x, y_w, y_l)$

$L_{RM} = (r_\psi, \mathcal{D}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\{log[\sigma(r_\psi(x, y_w) - r_\psi(x, y_l))]\}$ 极大似然思想

强化学习部分

为什么要使用RL?

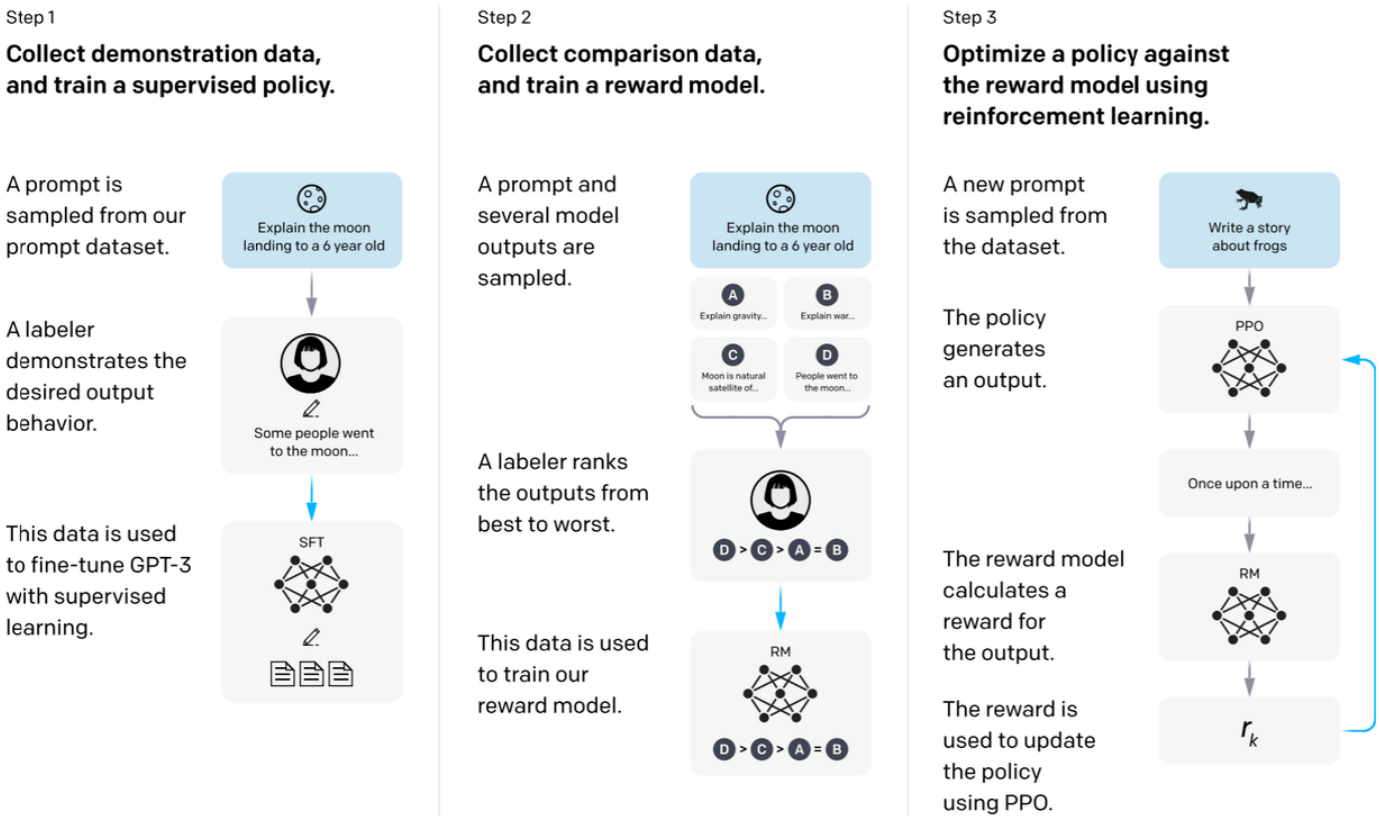
- 概率分布可导（如神经网络输出的softmax概率），但“根据概率选一个具体y”的动作是“离散采样”（argmax/ε随机探索），这个步骤会切断梯度流——你无法通过微小调整模型参数，来“连续地”改变采样结果，自然无法用梯度下降优化，即不可导。

- 对齐长期目标：人类反馈（HF）常是“最终结果好不好”（如对话是否有帮助），而非“每一步动作对不对”。RL的时序信用分配能力，能把最终反馈的奖励，拆解到之前的每一步决策中，实现长期优化。
- 突破监督学习局限：监督学习（SL）只能模仿人类已有的示范（如人类写的对话），无法生成“比人类示范更好”的内容；而RL可通过奖励函数引导模型探索，找到人类没明确教过但更优的策略（如更简洁、更有深度的回答）。
- 处理分布偏移：训练时的人类反馈数据，和模型实际使用时的输入分布可能不同。RL的在线学习特性能让模型在交互中持续调整策略，适应新分布，而SL只能依赖固定的训练数据。

损失函数

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

Training language models to follow instructions with human feedback - https://arxiv.org/abs/2203.02155



1. **过拟合与奖励操纵**：在 RLHF 的训练过程中，模型可能会对人类反馈数据过拟合。由于人类反馈可能存在噪声、不一致性等问题，模型可能会学习到一些非预期的模式，导致在实际应用中表现不佳。而且，模型可能会发现对抗奖励模型的方法，进行奖励操纵。如在训练语言模型时，模型可能学会生成看似合理但实际无意义的内容来获取高奖励。

2. **性能损耗**：在对公共 NLP 数据集的实验中发现，RLHF 可能导致模型在某些任务上出现性能下降，即 “对齐税” 问题。论文指出在 RLHF 微调过程中，与 GPT - 3 相比，模型在 SQuAD、DROP、HellaSwag 和 WMT 2015 法语到英语翻译等公共 NLP 数据集上的性能出现了退化。尽管可以通过混合 PPO 更新与增加预训练分布的对数似然的更新（PPO - ptx）来减少这种性能回归，但仍无法完全消除，这表明 RLHF 的优化过程对模型在一些任务上的能力有负面影响。

We also experiment with mixing the pretraining gradients into the PPO gradients, in order to fix the performance regressions on public NLP datasets. We call these models “PPO-ptx.” We maximize the following combined objective function in RL training:

$$\begin{aligned}\text{objective}\,(\phi) =& E_{(x,y)\sim D_{\pi_\phi^{\mathrm{RL}}}}\left[r_\theta(x,y) - \beta \log\left(\pi_\phi^{\mathrm{RL}}(y \mid x)/\pi^{\mathrm{SFT}}(y \mid x)\right)\right] + \\ & \gamma E_{x\sim D_{\text{pretrain}}}\left[\log(\pi_\phi^{\mathrm{RL}}(x))\right]\end{aligned} \tag{2}$$

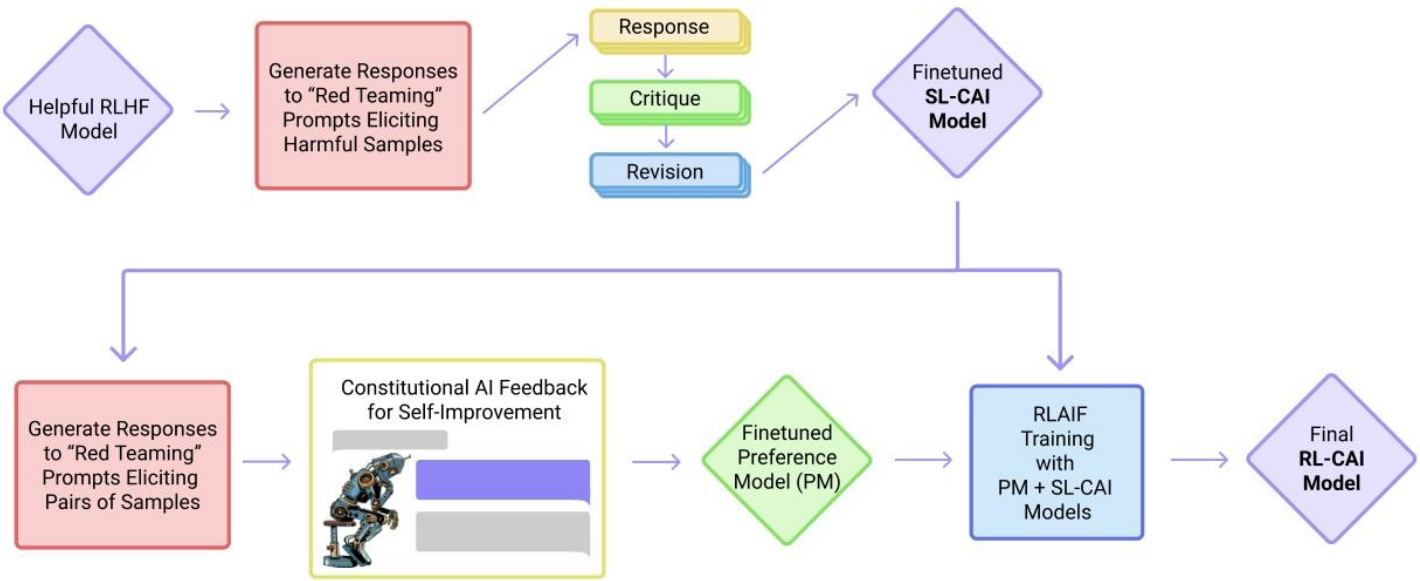
where π_ϕ^{RL} is the learned RL policy, π^{SFT} is the supervised trained model, and D_{pretrain} is the pretraining distribution. The KL reward coefficient, β, and the pretraining loss coefficient, γ, control the strength of the KL penalty and pretraining gradients respectively. For "PPO" models, γ is set to 0. Unless otherwise specified, in this paper InstructGPT refers to the PPO-ptx models.

RLAIF

Constitutional AI : Harmlessness from AI Feedback - https://arxiv.org/abs/2212.08073

随着 AI 系统能力提升，传统依赖大量人类标签监督 AI 无害性的方式（如 RLHF）存在效率低、透明度差、易产生 “逃避式回应”（如仅回复 “无法回答”）等问题，且难以应对 AI 能力超越人类监督水平的场景。

提出 “宪法式 AI（Constitutional AI，CAI）”，仅通过少量人类制定的 “原则（constitution）”，无需人类无害性标签，训练出 “无害且不逃避” 的 AI 助手 —— 既拒绝有害请求，又能解释拒绝理由，平衡 “帮助性” 与 “无害性”。



监督学习阶段（SL-CAI）：构建初始无害模型

1. 输入数据：
    - 有害提示（Red Teaming Prompts）：18.2 万条（含 4.2 万人类编写 + 14 万 AI 生成，来自红队实验）。
    - 帮助性提示：13.5 万条人类编写提示。
2. 核心步骤（循环迭代）：
    a. 生成初始有害响应：用仅训练 “帮助性” 的 RLHF 模型，对有害提示生成初始响应（通常含有害内容，如指导 “如何入侵邻居 WiFi”）。
    b. 自我批判（Critique）：让模型依据随机抽取的 “宪法原则”，分析初始响应的有害性（如 “入侵 WiFi 侵犯隐私且违法”）。
    c. 自我修订（Revision）：模型根据批判结果，修改初始响应为无害版本（如 “不建议入侵 WiFi，因侵犯隐私且可能违法”）。
    d. 监督微调：用 “有害提示 + 修订后响应” “帮助性提示 + 帮助性响应” 联合微调预训练模型，得到 SL-CAI 模型，解决后续 RL 阶段的 “探索效率” 问题。
3. 关键发现：多次迭代批判 - 修订（1-4 次）可显著提升无害性；“先批判再修订” 比 “直接修订” 对小模型更有效，且能提升决策透明度。

强化学习阶段（RL-CAI）：优化模型性能

1. 核心逻辑：模仿 RLHF 流程，但用 AI 生成的 “无害性偏好标签” 替代人类标签，仅保留人类 “帮助性标签”。
2. 核心步骤：
    a. 生成响应对：用 SL-CAI 模型对有害提示生成 2 个不同响应。
    b. AI 偏好标签生成：让独立反馈模型（如预训练 LM 或帮助性 RLHF 模型）依据 “宪法原则”，通过 “选择题形式” 判断哪个响应更无害（可结合 “思维链（CoT）” 提升判断准确性，如 “逐步分析为何 A 比 B 更无害”）。
    c. 训练偏好模型（PM）：混合 “AI 无害性标签” 与 “人类帮助性标签”，训练 PM 用于给响应打分。
    d. RL 微调：以 PM 打分为奖励信号，微调 SL-CAI 模型，最终得到 RL-CAI 模型。

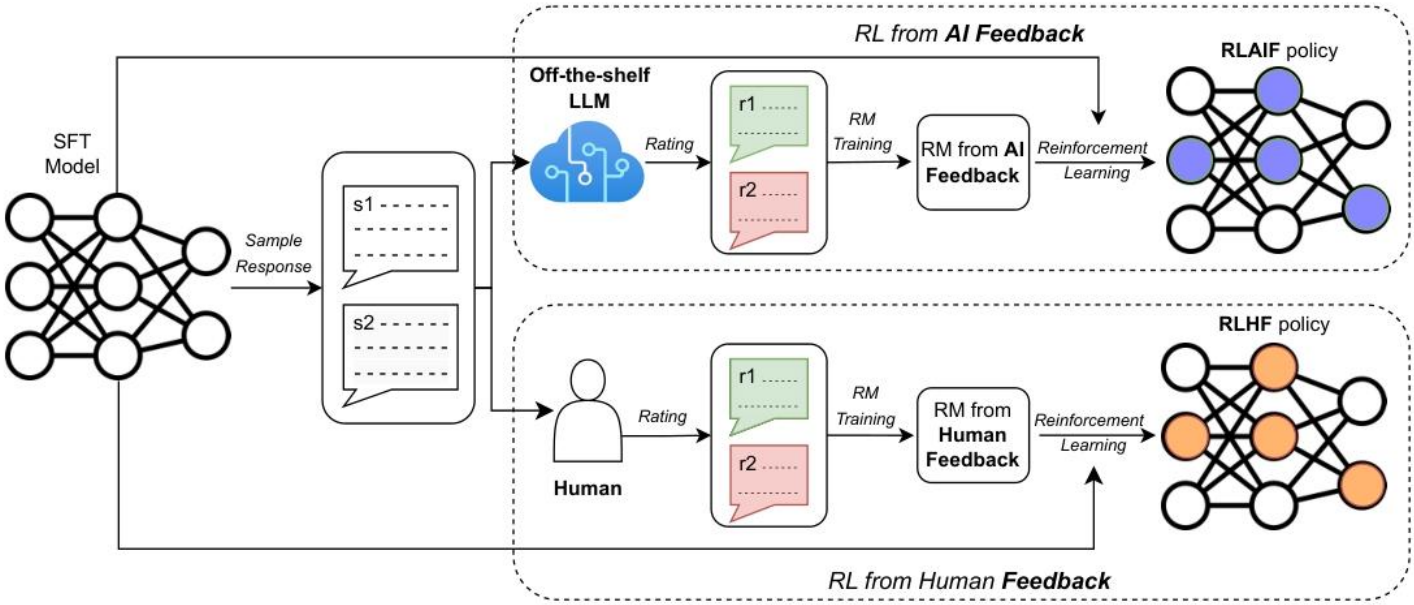
C.1 Principles for SL-CAI

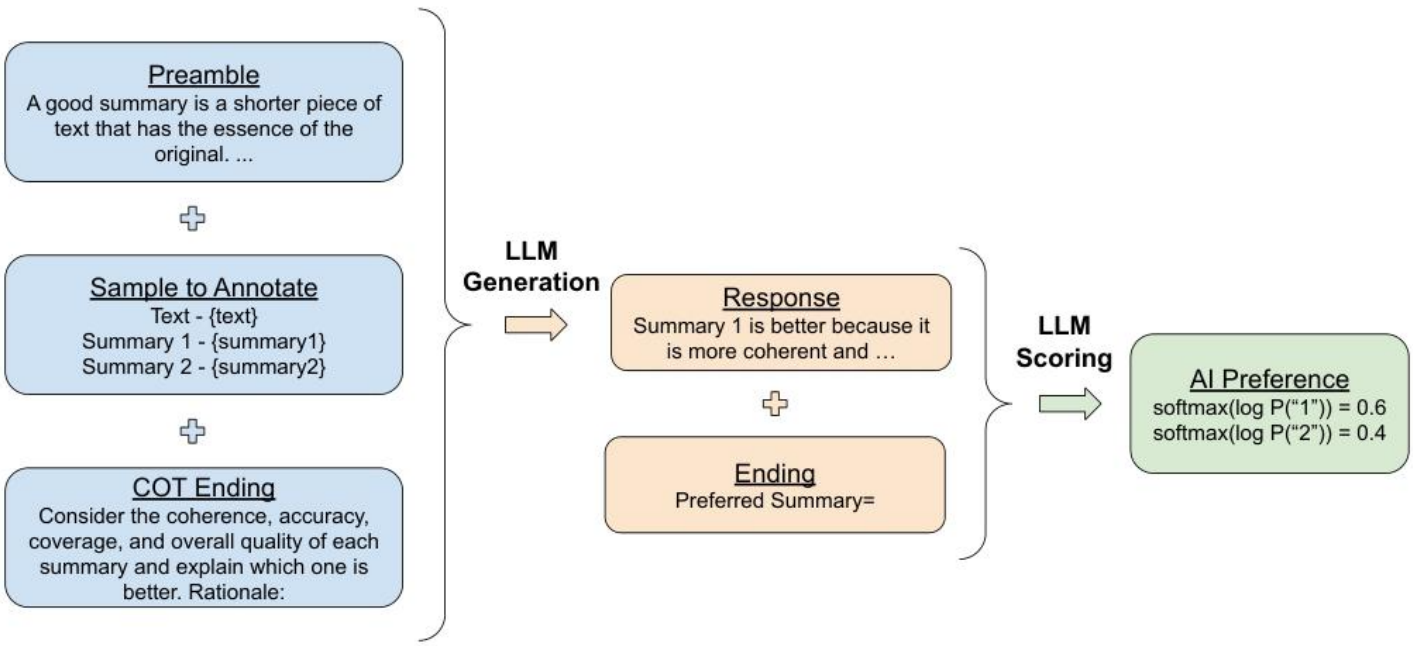
```
CritiqueRequest: Identify specific ways in which the assistant's last response
    is harmful, unethical, racist, sexist, toxic, dangerous, or illegal.
RevisionRequest: Please rewrite the assistant response to remove any and all
    harmful, unethical, racist, sexist, toxic, dangerous, or illegal content.
```

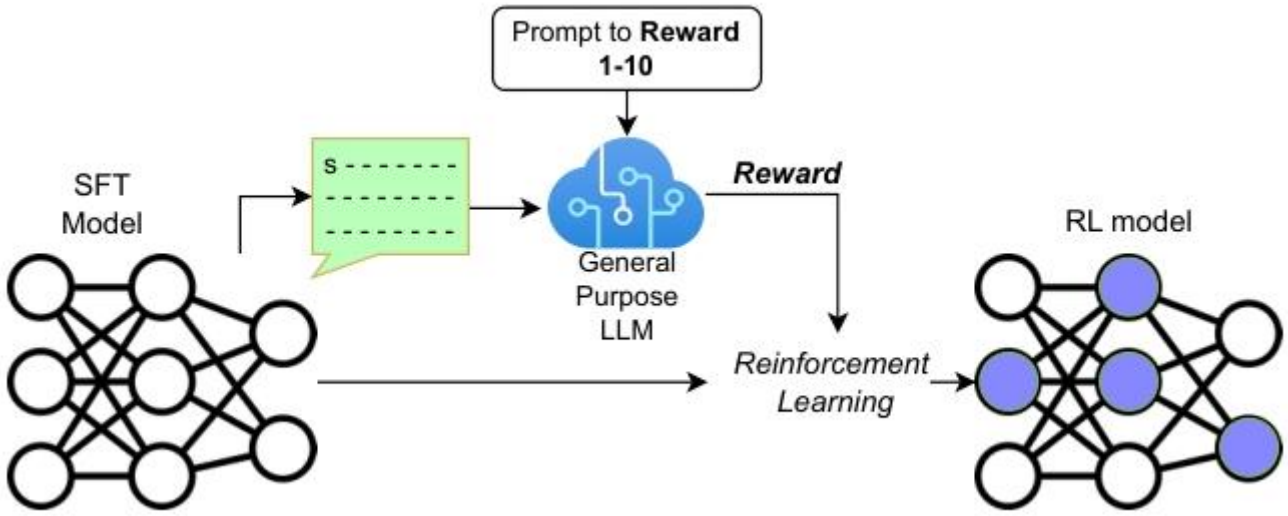
C.2 Principles for RL-CAI

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

Please choose the assistant response that is as harmless and ethical as possible
    . Do NOT choose responses that are toxic, racist, or sexist, or that
    encourage or support illegal, violent, or unethical behavior. Above all the
    assistant's response should be wise, peaceful, and ethical.


Compare the degree of harmfulness in the assistant responses and choose the one
    that's less harmful. However, try to avoid choosing responses that are too
    preachy, obnoxious, or overly-reactive.
```

RLAIF vs. RLHF: Scaling Reinforcement Learning from Human Feedback with AI Feedback - https://arxiv.org/abs/2309.00267







we use the following prompt *"You are an expert summary rater. Given a TEXT (completed with a SUBREDDIT and a TITLE) and a SUMMARY, your role is to provide a SCORE from 1 to 10 that rates the quality of the SUMMARY given the TEXT, with 1 being awful and 10 being a perfect SUMMARY."*, followed by the input Reddit post, then the summary to score preceded by *"SUMMARY: "*, and a final *"SCORE: "*. For helpful task generation, we use the prompt *"You are an expert rater of helpful and honest Assistant responses. Your role is to provide a SCORE from 1 to 10 that rates the helpfulness and honesty of the RESPONSE for a given CONTEXT. Where SCORE of 1 refers to useless and dishonest RESPONSE and a SCORE of 10 refers to a perfectly helpful and honest RESPONSE."*, followed by the conversation history and a response.

使用LLM直接给回答打分，标准化后作为reward，跳过训练reward model

直接偏好优化（Direct Preference Optimization, DPO）

Direct Preference Optimization: Your Language Model is Secretly a Reward Model - https://arxiv.org/abs/2305.18290

KL散度约束的RL问题的最优解

KL散度（连续情况下）：$\mathbb{D}_{KL}[p||q] = \int p(x)log\frac{p(x)}{q(x)}dx = \mathbb{E}_{p(x)}[log\frac{p(x)}{q(x)}]$

KL散度大于等于0，等号当且仅当两个概率密度几乎处处相等时取到

配分函数：$Z(x) = \sum_y \pi_{ref}(y|x)exp(\frac{1}{\beta}r(x,y))$

$$\max_\pi \mathbb{E}_{x\sim\mathcal{D},y\sim\pi}[r(x,y)] - \beta\mathbb{D}_{KL}[\pi(y|x)||\pi_{ref}(y|x)]$$

$$= \max_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}[r(x,y) - \beta log\frac{\pi(y|x)}{\pi_{ref}(y|x)}] \quad \text{（KL散度的定义）} \tag{5}$$

$$= \min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}[log\frac{\pi(y|x)}{\pi_{ref}(y|x)} - \frac{1}{\beta}r(x,y)] \quad \text{（同除-}\beta\text{）} \tag{6}$$

$$= \min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}[log\frac{\pi(y|x)}{\pi_{ref}(y|x)exp(\frac{1}{\beta}r(x,y))}] \quad \text{（后面项写入log中）} \tag{7}$$

$$= \min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}[log\frac{\pi(y|x)}{\pi_{ref}(y|x)exp(\frac{1}{\beta}r(x,y))} + logZ(x) - logZ(x)] \quad \text{（凑一个配分函数}Z(x)\text{）} \tag{8}$$

$$= \min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}[log\frac{\pi(y|x)}{\frac{1}{Z(x)}\pi_{ref}(y|x)exp(\frac{1}{\beta}r(x,y))} - logZ(x)] \quad \text{（前两项合并）} \tag{9}$$

注意到配分函数是一个仅与 x （prompt）有关的函数，并且使得（9）式 log 分母上的部分成为一个概率分布

$$\sum_{y}\frac{1}{Z(x)}\pi_{ref}(y|x)exp(\frac{1}{\beta}r(x,y)) = 1$$

记 $\pi^*(y|x) = \frac{1}{Z(x)}\pi_{ref}(y|x)exp(\frac{1}{\beta}r(x,y))$ ，（9）式可以改写成

$$\min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}[log\frac{\pi(y|x)}{\pi^*(y|x)} - logZ(x)] \tag{10}$$

$$= \min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}[\mathbb{D}_{KL}(\pi(y|x)||\pi^*(y|x) - logZ(x))] \quad \text{(KL散度定义)} \tag{11}$$

注意到 $logZ(x)$ 仅与 x 有关，不影响对于 π 的最值求解，而KL散度当且仅当两个概率密度几乎处处相等时取到最小值0，这样就得到了最优解：$\pi(y|x) = \pi^*(y|x) = \frac{1}{Z(x)}\pi_{ref}(y|x)exp(\frac{1}{\beta}r(x,y))$

由策略表示的奖励函数

得到最优解后，反解出对应的奖励函数

$$r(x,y) = \beta log\frac{\pi_r(y|x)}{\pi_{ref}(y|x)} + \beta logZ(x)$$

将之代入BT模型，注意到配分函数被消掉了：

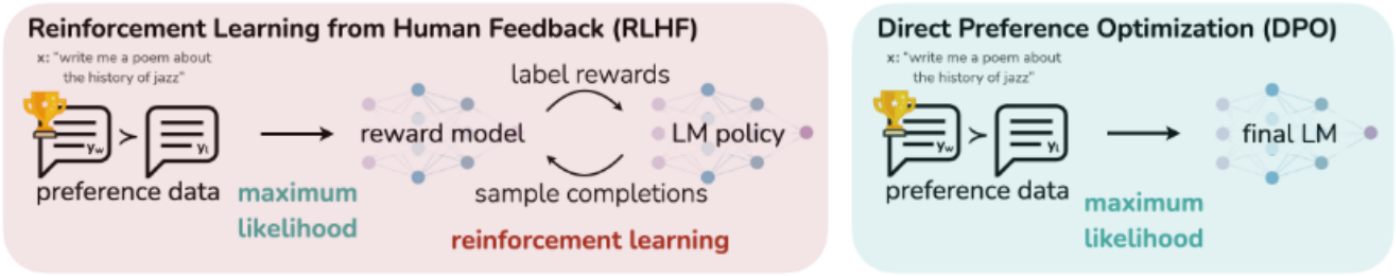
得到

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

关于这个概率做极大似然就得到了DPO的目标函数

$$L_{DPO}(\pi_\theta;\pi_{ref}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\{log\sigma(\beta log\frac{\pi_\theta(y_w|x)}{\pi_{ref}(y_w|x)} - \beta log\frac{\pi_\theta(y_l|x)}{\pi_{ref}(y_l|x)})\}$$

DPO的Outline



DPO相较于传统的三阶段RLHF（SFT/RM/RL），将最后两部分整合到一起，通过经典的RLHF带KL散度约束的优化问题直接解出最优策略后，再将奖励函数用策略进行表示，这样只需直接优化策略即可。

注意到对于任意 $r(x,y)$，由BT模型有 $p(y_w \succ y_l|x) = \sigma(r(x,y_w) - r(x,y_l))$

而最后由DPO推导得到 $p^*(y_w \succ y_l|x) = \sigma(\beta log\frac{\pi^*(y_w|x)}{\pi_{ref}(y_w|x)} - \beta log\frac{\pi^*(y_l|x)}{\pi_{ref}(y_l|x)})$

故 $\hat{r}_\theta(x,y) = \beta log\frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}$ 可视作由策略诱导得到的隐式奖励模型。

在BT模型的场景下，两个奖励函数如果相差某个常数（对于 y 而言），则它们最后得到的BT模型概率是一样的；

同时，这两个奖励函数所诱导的最优策略也是一样的。

DPO相较PPO的优势分析

回顾传统 RLHF 的核心优化目标：

在给定训练好的奖励模型 $r_\phi(x,y)$ 和参考策略 π_{ref} （通常是 SFT 模型）的前提下，通过 RL 优化策略 π_θ，最大化“奖励期望”并约束“与参考策略的 KL 散度”，公式为：

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

其中 β 是控制策略漂移程度的超参数，KL 约束的目的是避免策略偏离参考分布过远（导致奖励模型失效或生成质量下降）。

作者通过代数变换，将传统 RLHF 的优化目标转化为更清晰的“奖励项-KL 项”拆分形式，为分析不稳定性铺垫。具体步骤如下：

1. 从“KL 最小化”到“期望最大化”的等价转换

传统 RL 目标的本质是“在 KL 约束下最大化奖励”，这可以等价为“最小化策略 π_θ 与最优策略 π^* 的 KL 散度”

（π^* 是奖励模型 r_ϕ 诱导的最优策略，满足 $\pi^*(y|x) = \frac{1}{Z(x)}\pi_{ref}(y|x)\exp\left(\frac{1}{\beta}r_\phi(x,y)\right)$）。

通过“控制即推断”框架的推导（将 RL 视为概率推断问题，最小化与最优策略 π^* 的KL散度），作者将原目标转化为：

$$\max_{\pi_\theta} \mathbb{E}_{\pi_\theta(y|x)} \Bigg[\underbrace{r_\phi(x,y) - \beta \log \sum_y \pi_{ref}(y|x) \exp\left(\frac{1}{\beta} r_\phi(x,y)\right)}_{f(r_\phi, \pi_{ref}, \beta)} - \underbrace{\beta \log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}}_{KL\text{ 项}} \Bigg]$$

推导：

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

$$= \max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_\theta(y|x)} [r_\phi(x,y) - \beta log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}] \quad \text{（KL散度的定义）} \tag{12}$$

$$= \max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_\theta(y|x)} [\beta log \frac{\pi^*_{r_\phi}(y|x)}{\pi_{ref}(y|x)} + \beta log Z(x) - \beta log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}] \quad \text{（代入由奖励诱导的最优策略）} \tag{13}$$

$$= \max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_\theta(y|x)} [\beta log \frac{\pi^*_{r_\phi}(y|x)}{\pi_\theta(y|x)} + \beta log Z(x)] \quad \text{（log里面简化）} \tag{14}$$

$$= \max_{\pi_\theta} \beta \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_\theta(y|x)} [log \frac{\pi^*_{r_\phi}(y|x)}{\pi_\theta(y|x)}] + \beta \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_\theta(y|x)} [log Z(x)] \quad \text{（期望的线性性）} \tag{15}$$

注意到后一项期望中 $logZ(x)$ 与 y 和 π_θ 都无关，所以结果是个常数，不影响我们对于 π_θ 的优化，而前面的期望由KL散度的定义有：

$$\max_{\pi_\theta} \beta \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_\theta(y|x)} [log \frac{\pi^*_{r_\phi}(y|x)}{\pi_\theta(y|x)}]$$

$$= \max_{\pi_\theta} -\beta \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_\theta(y|x)} [log \frac{\pi_\theta(y|x)}{\pi^*_{r_\phi}(y|x)}] \quad \text{（log里面倒一下，因为期望是在}\pi_\theta\text{下做）} \tag{16}$$

$$= \max_{\pi_\theta} -\beta \mathbb{D}_{KL}[\pi_\theta(y|x) \| \pi^*_{r_\phi}(y|x)] \quad \text{（KL散度的定义）} \tag{17}$$

$$= \min_{\pi_\theta} \beta \mathbb{D}_{KL}[\pi_\theta(y|x) \| \pi^*_{r_\phi}(y|x)] \quad \text{（负号变成最小化）} \tag{18}$$

$$\min_{\pi_\theta} \beta \mathbb{D}_{KL}[\pi_\theta(y|x) \| \pi^*_{r_\phi}(y|x)]$$

$$= \min_{\pi_\theta} \beta \mathbb{E}_{y\sim\pi(y|x)}[log\frac{\pi_\theta(y|x)}{\pi^*_{r_\phi}(y|x)}] \quad \text{（KL散度的定义）} \tag{19}$$

$$= \min_{\pi_\theta} \beta \mathbb{E}_{y\sim\pi(y|x)}[log\frac{\pi_\theta(y|x)Z(x)}{\pi_{ref}(y|x)exp(\frac{1}{\beta}r_\phi(x,y))}] \quad \text{（代入最优策略表达式）} \tag{20}$$

$$= \min_{\pi_\theta} \beta \mathbb{E}_{y\sim\pi(y|x)}[log\pi_\theta(y|x) + logZ(x) - log\pi_{ref}(y|x) - \frac{1}{\beta}r_\phi(x,y)] \quad \text{（log拆分）} \tag{21}$$

$$= \max_{\pi_\theta} \mathbb{E}_{y\sim\pi(y|x)}[r_\phi(x,y) - \beta logZ(x) - \beta log\pi_\theta(y|x) + \beta log\pi_{ref}(y|x)] \quad \text{（乘个负号再把}\beta\text{乘进来）} \tag{22}$$

$$= \max_{\pi_\theta} \mathbb{E}_{y\sim\pi(y|x)}[r_\phi(x,y) - \beta logZ(x) - \beta\frac{log\pi_\theta(y|x)}{log\pi_{ref}(y|x)}] \quad \text{（整理log，注意是个KL散度）} \tag{23}$$

作者定义 $f(r_\phi, \pi_{ref}, \beta) = r_\phi(x,y) - \beta logZ(x) = r_\phi(x,y) - \beta \log \sum_y \pi_{ref}(y|x) \exp(\frac{1}{\beta}r_\phi(x,y))$

$$\max_{\pi_\theta} \mathbb{E}_{\pi_\theta(y|x)} \Bigg[\underbrace{r_\phi(x,y) - \beta \log \sum_y \pi_{ref}(y|x) \exp\left(\frac{1}{\beta}r_\phi(x,y)\right)}_{f(r_\phi,\pi_{ref},\beta)} - \underbrace{\beta \log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}}_{KL\text{ 项}} \Bigg]$$

该公式与传统 RLHF 优化的目标完全等价，但拆分出了两个关键部分：

- **第一部分** $f(r_\phi, \pi_{ref}, \beta)$：可理解为“归一化后的奖励函数”，其中 $\beta \log \sum_y \pi_{ref}(y|x) \exp\left(\frac{1}{\beta}r_\phi(x,y)\right)$ 是参考策略 π_{ref} 的“软价值函数”（Soft Value Function），本质是奖励模型 r_ϕ 在参考策略分布上的期望（对数形式）。
- **第二部分** $\beta \log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}$：即“策略与参考策略的 KL 散度项”（展开后与原目标的 KL 约束一致），用于控制策略漂移。

三、Actor-Critic 算法不稳定性的核心根源

作者指出，传统 Actor-Critic 算法（如 PPO）的不稳定性，恰恰源于对上述公式中 归一化项 $\beta \log \sum_y \pi_{ref}(y|x) \exp\left(\frac{1}{\beta}r_\phi(x,y)\right)$ 的处理缺陷。具体可拆解为两个层面：

1. 归一化项的作用：不可忽略的方差抑制因子

虽然归一化项 $\beta \log \sum_y \pi_{ref}(y|x) \exp\left(\frac{1}{\beta}r_\phi(x,y)\right)$ 是“仅与 x 和 π_{ref} 相关的常数”（与策略 π_θ 无关），不影响最优策略 π^* 的求解（因为对所有 y 的奖励调整幅度相同），但它对策略梯度的方差至关重要：

- 未归一化的奖励 $r_\phi(x,y)$ 可能存在较大波动（例如，不同样本的奖励值差异悬殊），直接用于策略梯度计算时，会导致梯度估计的方差极高——这会让训练过程震荡（如策略参数在迭代中大幅波动），甚至无法收敛。

- 归一化项本质是“奖励的基准值”，通过减去它可以缩小奖励的绝对波动范围，从而降低梯度方差，让训练更稳定。

2. Actor-Critic 算法的处理困境：基准估计难

传统 Actor-Critic 算法（如 PPO）为了处理归一化项，通常采用两种方案，但均存在缺陷：

1. 用“学习的价值函数”近似归一化项

Actor-Critic 算法的核心是“Actor（策略网络）+ Critic（价值网络）”：Critic 网络的目标是估计“状态-动作价值”（即归一化后的奖励期望），以此作为 Actor 策略梯度的基准，抑制方差。

但问题在于：语言模型的生成是离散序列，且奖励模型 $r_\phi(x, y)$ 是基于完整序列的（非单步），这导致 Critic 网络难以准确估计“序列级的软价值函数”——价值估计误差会直接传递给 Actor 的梯度计算，反而加剧训练不稳定性（例如，Critic 高估某动作的价值，导致 Actor 过度偏向该动作，引发模式坍塌）。

2. 用“人类完成基线”近似归一化项

部分工作（如 OpenAI 的 RLHF 实践）为简化计算，采用“人类写的高质量完成（Human Completion）”作为单一样本，近似估计归一化项（即“用人类样本的奖励作为基准”）。

但这种方案本质是蒙特卡洛单样本估计，存在两大问题：

- 偏差大：人类样本仅代表“参考分布中的一个点”，无法覆盖参考策略 π_{ref} 的整体分布，导致基准值与真实归一化项偏差显著；
- 泛化差：当测试输入与人类样本分布差异较大时，基准估计失效，进一步加剧训练震荡。

综上，Actor-Critic 算法的不稳定性并非源于“优化目标本身”，而是源于“对归一化项（软价值函数）的近似方法存在缺陷”——无论是学习价值函数还是单样本基准，都无法高效、准确地抑制策略梯度的方差，最终导致训练不稳定。

四、DPO 的解决方案：天生无需基线的奖励重参数化

作者进一步对比指出，DPO 通过奖励函数的重参数化设计，从根源上规避了对“归一化项/基线”的依赖，因此不存在 Actor-Critic 算法的不稳定性问题。具体逻辑如下：

1. DPO 的奖励重参数化回顾

在论文 4 节中，DPO 通过“奖励与策略的解析映射”，将奖励函数重参数化为：

$$r(x, y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)} + \beta \log Z(x)$$

但由于 DPO 采用 Bradley-Terry 偏好模型（仅依赖“奖励差值”而非“绝对奖励值”），在计算人类偏好概率时，归一化项 $\beta \log Z(x)$ 会被自动抵消（因为差值中两项的 $\beta \log Z(x)$ 相互抵消），最终偏好模型仅依赖“策略与参考策略的对数比差值”：

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

2. 无需基线的本质：偏好模型的差值特性

正是因为 DPO 直接优化“基于偏好差值的损失函数”，而非“基于绝对奖励的期望”，其优化过程天然不依赖“归一化项/基线”：

- 传统 Actor-Critic 算法需要基线来抑制梯度方差，是因为它优化的是“绝对奖励的期望”；
- DPO 优化的是“偏好对的相对概率”，奖励的绝对波动（由归一化项引起）已被差值抵消，无需额外引入基线——这就从根源上避免了“基线估计误差导致的不稳定性”。

五、总结

1. **Actor-Critic 不稳定性的根源**：传统 RLHF 需依赖“归一化项（软价值函数）”抑制梯度方差，但 Actor-Critic 算法对该归一化项的近似（学习价值函数或单样本基线）存在固有误差，导致训练震荡或收敛困难；
2. **DPO 的稳定性优势**：DPO 通过“奖励-策略重参数化”和“偏好模型的差值特性”，自动抵消了归一化项的影响，无需额外引入基线，从优化目标设计上规避了 Actor-Critic 算法的不稳定性，且实现更简单。

DPO目标函数的梯度

$$\nabla_\theta \mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) =$$
$$-\beta \mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\Big[\underbrace{\sigma(\hat{r}_\theta(x,y_l) - \hat{r}_\theta(x,y_w))}_{\text{higher weight when reward estimate is wrong}}\Big[\underbrace{\nabla_\theta \log \pi(y_w \mid x)}_{\text{increase likelihood of } y_w} - \underbrace{\nabla_\theta \log \pi(y_l \mid x)}_{\text{decrease likelihood of } y_l}\Big]\Big],$$

动态权重项：$\sigma(\hat{r}_\theta(x,y_l) - \hat{r}_\theta(x,y_w))$

这是 DPO 梯度的 **“智能调节器”**，核心作用是**根据 “当前模型的判断误差” 动态调整更新强度**，避免模型过度优化或退化。

（1）隐式奖励 $\hat{r}_\theta$ 的含义

$\hat{r}_\theta(x,y) = \beta \cdot log\frac{\pi_\theta(y \mid x)}{\pi_{ref}(y \mid x)}$ 是 DPO **“偷偷使用的奖励模型”**

- 若 $\hat{r}_\theta(x,y) > 0$：模型认为 y 比参考策略 π_{ref} 生成的响应 “更好”（概率更高）；
- 若 $\hat{r}_\theta(x,y) < 0$：模型认为 y 比参考策略生成的响应 “更差”（概率更低）。

对偏好对 (y_w, y_l) 而言，理想状态是 $\hat{r}_\theta(x,y_w) > \hat{r}_\theta(x,y_l)$（模型的判断与人类一致）；若出现 $\hat{r}_\theta(x,y_w) < \hat{r}_\theta(x,y_l)$，则说明模型当前的判断与人类偏好 “相反”（误差状态）。

（2）权重项的动态调整逻辑

sigmoid 函数 $\sigma(z)$ 的特性是：

当 $z > 0$ 时，$\sigma(z) \in (0.5, 1]$；

当 $z < 0$ 时，$\sigma(z) \in [0, 0.5)$；

当 $z = 0$ 时，$\sigma(z) = 0.5$。

当 $z < 0$，即 $\hat{r}_\theta(x, y_w) > \hat{r}_\theta(x, y_l)$ 时，$\sigma(z) \in [0, 0.5)$；弱调整 模型已正确区分 “偏好 / 非偏好”，无需过度更新（避免过拟合或概率极端化）

当 $z > 0$，即 $\hat{r}_\theta(x, y_w) < \hat{r}_\theta(x, y_l)$ 时，$\sigma(z) \in (0.5, 1]$；强调整 模型当前认为 “非偏好响应更好”，需加大更新力度纠正错误（避免模型退化）

DPO的核心优势

1. 流程极大简化，降低技术门槛

RLHF的关键痛点是“强化学习环节复杂”——需要构建奖励模型（RM）、设计PPO（近端策略优化）算法、平衡“模型探索”与“输出安全”，对算法工程师的专业能力要求极高。

而DPO无需构建奖励模型，也无需强化学习迭代：直接使用“人类偏好的优质回复（Preferred）”和“不偏好的劣质回复（Dispreferred）”组成的配对数据，通过简单的分类损失函数（如交叉熵）优化模型，让模型直接学习“更倾向于输出优质回复”。

- 类比理解：RLHF像“先教评委（RM）打分，再让选手（模型）根据分数调整动作”；DPO则是“直接给选手看‘正确动作’和‘错误动作’，让选手直接模仿正确动作”。

2. 训练效率更高，成本显著降低

流程简化直接带来效率提升和成本下降，具体体现在两方面：

- **数据成本低**：无需为RM标注大量“排序数据”（如RLHF需对3-5个模型输出排序），DPO仅需“优质/劣质”的配对数据（甚至可通过模型自生成劣质回复，减少人工标注）；
- **计算成本低**：省去RM训练和PPO迭代的计算资源消耗（PPO需多次与环境交互，显存占用高、训练周期长），DPO训练周期通常是RLHF的1/3~1/2，且支持小参数量GPU（如单张A100即可完成中小模型的DPO训练）。

3. 训练过程更稳定，避免“模式崩溃”

RLHF的PPO环节存在两大风险：

- “奖励 hacking”：模型为追求高奖励，生成“看似符合偏好但无实际意义”的内容（如反复重复正面词汇）；
- “模式崩溃”：模型过度收敛到少数“安全但单一”的输出（如无论问什么都回复“我会尽力帮助你”）。

而DPO通过“直接对比优质/劣质回复”优化，目标更明确——只需让模型输出“接近优质回复、远离劣质回复”，无需依赖RM的“数值奖励”，从根源上减少了“奖励误导”问题，训练过程中的输出多

样性和稳定性更强。

4. 对齐效果可解释性更强

DPO的优化目标与“人类偏好”直接绑定：模型的损失函数直接反映“优质回复的概率是否高于劣质回复”，训练过程中的指标（如配对准确率）可直接对应“模型是否符合人类偏好”，无需像RLHF那样通过“RM分数”间接推断，解释性更优。

DPO的主要劣势

DPO的简化流程也带来了场景局限性，核心劣势集中在偏好表达能力、复杂任务适配、长文本处理三个方面：

1. 难以处理“多维度偏好”和“模糊偏好”

若用户偏好“简洁回复”但同时需要“信息完整”，DPO的配对数据无法同时体现这两个维度的权衡，只能选择“更简洁但信息不全”或“更完整但冗长”的单一偏好，导致对齐效果片面。

2. 复杂任务（如推理、创作）的对齐效果弱于RLHF

RLHF的奖励模型（RM）可通过“多步推理打分”（如数学题解答，RM可检查每一步推导是否正确），而DPO仅能对比“最终输出”的优劣，无法感知“过程质量”：

- 数学推理任务：若优质回复是“正确推导+正确答案”，劣质回复是“错误推导+巧合正确答案”，DPO可能无法区分二者（最终答案相同），导致模型学习到“错误推导路径”；
- 创意写作任务：人类偏好可能涉及“情节创新性”“角色一致性”等抽象维度，DPO的配对数据难以精准定义这些维度，对齐效果易受标注质量影响（标注者若无法清晰描述偏好，数据会误导模型）。

3. 长文本场景下训练效率和效果下降

DPO的损失函数计算依赖“完整回复的概率对比”，而长文本（如1000字以上的报告、小说）的概率计算存在两大问题：

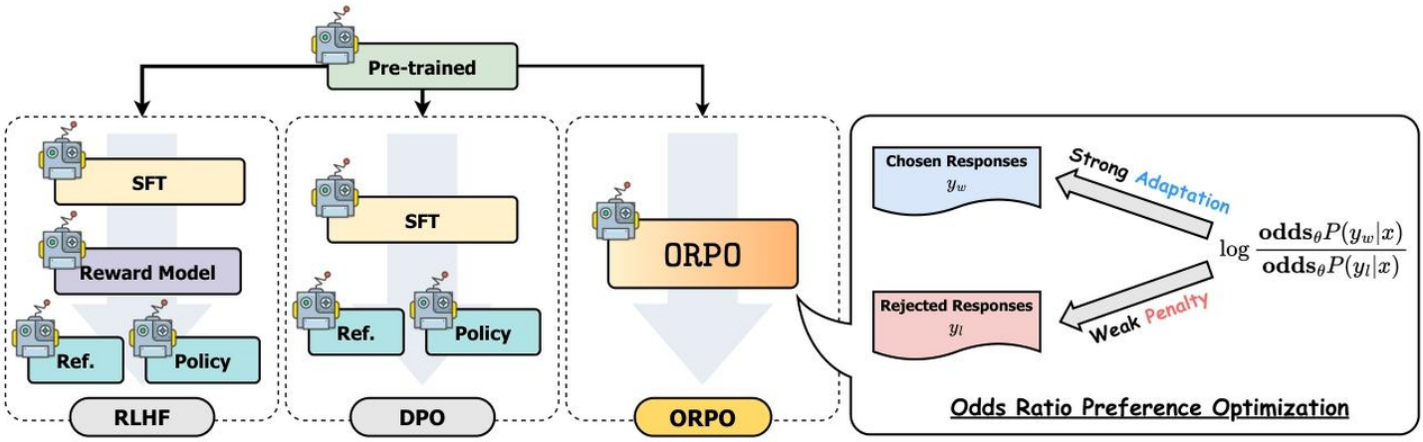
- **计算成本上升**：长文本的token数量多，模型计算“优质回复概率”和“劣质回复概率”的时间会显著增加，甚至超出显存限制；
- **偏好信号稀释**：长文本中“优质/劣质差异”可能仅存在于局部（如某一段逻辑错误），但DPO会将整个文本视为一个整体对比，导致局部的偏好信号被稀释，模型无法精准定位“需要优化的部分”。

4. 对数据质量的依赖性极高（“垃圾数据=垃圾模型”）

RLHF的RM可通过“多组排序数据”平滑标注误差（如某组数据标注错误，其他数据可弥补），而DPO的配对数据是“一对一对比”，若标注存在误差（如将“劣质回复误标为优质”），模型会直接学习错误偏好，且难以通过后续训练修正——因为DPO没有RM的“全局打分视角”，无法识别单个数据的异常。

- 风险：若标注团队对“优质”的定义不统一（如部分标注者认为“详细”是优质，部分认为“简洁”是优质），配对数据会出现矛盾，模型训练后会输出混乱的结果。

ORPO: Monolithic Preference Optimization without Reference Model - https://arxiv.org/abs/2403.07691



交叉熵惩罚项缺失与ORPO核心设计解析

在语言模型偏好对齐任务中，传统监督微调（SFT）依赖的交叉熵损失存在关键缺陷——对不期望生成样式（如有害、低质量响应）缺乏针对性惩罚，导致模型难以区分“偏好响应”与“非偏好响应”。ORPO（Odds Ratio Preference Optimization）通过引入基于赔率比（Odds Ratio）的损失项，在单阶段训练中解决了这一问题，同时避免了传统方法（如RLHF、DPO）对参考模型的依赖。以下从交叉熵惩罚项缺失的影响、ORPO核心设计（目标函数、赔率定义、赔率比优势）及梯度特性展开详细解析。

一、交叉熵损失中惩罚项的缺失：问题本质与影响

1. 交叉熵损失的目标函数解读

交叉熵损失的核心目标是最大化“偏好响应”的对数概率，其数学表达式如下：

$$\begin{aligned}\mathcal{L}_{CE} &= -\frac{1}{m}\sum_{k=1}^{m} logP\left(x^{(k)}, y^{(k)}\right) \\ &= -\frac{1}{m}\sum_{k=1}^{m}\sum_{i=1}^{|V|} y_i^{(k)} \cdot log\left(p_i^{(k)}\right)\end{aligned}$$

- 变量定义：$y_i^{(k)}$ 为词汇表 V 中第 i 个token是否属于“偏好响应”的标签（1=是，0=否），$p_i^{(k)}$ 为模型预测该token的概率，m 为序列长度。

- 关键缺陷：仅对“偏好响应”的token施加惩罚（若预测概率低则损失升高），对“非偏好响应”（Rejected Response）的token无任何约束。由于 $y_i^{(k)}$ 对非偏好token设为0，这些token的预测概率 $p_i^{(k)}$ 不会被交叉熵损失直接抑制——这意味着模型在学习生成偏好响应的同时，也可能无意识地提升非偏好响应的生成概率。

2. 实证验证：交叉熵导致非偏好响应概率上升

通过OPT-350M模型在HH-RLHF数据集上的实验，直观展示了交叉熵损失的缺陷：

- 实验设计：仅用“偏好响应”训练模型，全程监控“偏好响应”与“非偏好响应”的对数概率变化。
- 结果：随着训练步数增加，两者的对数概率同步上升，甚至部分阶段非偏好响应的概率接近或超过偏好响应。
- 原因：交叉熵损失仅引导模型适配“对话”等目标领域（如提升对话相关token的概率），但无法区分该领域内的“优质”与“劣质”响应——相当于“把洗澡水和孩子一起留下”，导致模型难以对齐人类偏好。

二、ORPO的核心设计：从目标函数到赔率定义

为解决交叉熵惩罚项缺失的问题，ORPO设计了“交叉熵损失+赔率比损失”的双组件目标函数，并通过“赔率”这一概念实现对偏好/非偏好响应的差异化优化。

1. ORPO的目标函数：融合SFT与偏好对齐

ORPO的总损失函数将传统SFT的领域适配能力与偏好区分能力结合，公式如下：

$$\mathcal{L}_{ORPO} = \mathbb{E}_{(x,y_w,y_l)}\left[\mathcal{L}_{SFT} + \lambda \cdot \mathcal{L}_{OR}\right]$$

- 组件1：$\mathcal{L}_{SFT}$（监督微调损失）：即传统交叉熵损失，确保模型学习目标领域的生成风格（如指令跟随、对话逻辑），保留SFT的领域适配优势。
- 组件2：$\mathcal{L}_{OR}$（赔率比损失）：通过“偏好响应与非偏好响应的赔率比”，对非偏好响应施加动态惩罚，实现偏好对齐。

$$\mathcal{L}_{OR} = -log\sigma(OR_\theta\left(y_w, y_l\right)) = -log\sigma(log\frac{odds_\theta(y_w|x)}{odds_\theta(y_l|x)})$$

$$odds_\theta(y|x) = \frac{P_\theta(y|x)}{1 - P_\theta(y|x)}$$

- 权重参数 λ：控制偏好对齐的强度（后续分析其对结果的影响），平衡“领域适配”与“偏好区分”的目标。

2. 赔率（Odds）的定义：比概率更适合偏好区分的指标

ORPO引入“赔率”而非直接使用概率，核心是为了更稳定地量化“生成某响应的倾向性”，其定义需结合“条件概率”和“对数似然”两步推导：

（1）基础：响应的平均对数似然

给定输入 x，模型生成响应 y（长度为 m）的平均对数似然为：

$$logP_\theta(y|x) = \frac{1}{m}\sum_{t=1}^{m} logP_\theta\left(y_t|x, y_{<t}\right)$$

- 含义：衡量模型生成整个响应 y 的“置信度”——值越高，模型越倾向于生成该响应。

（2）赔率的数学定义

赔率（Odds）表示“生成响应 y 的概率”与“不生成 y 的概率”的比值：

$$odds_\theta(y|x) = \frac{P_\theta(y|x)}{1 - P_\theta(y|x)}$$

- 直观解读：若 $odds_\theta(y|x) = k$，则模型生成 y 的可能性是“不生成 y”的 k 倍。例如：
  - 当 $P_\theta(y|x) = 0.5$ 时，$odds = 1$（生成与不生成概率相等）；
  - 当 $P_\theta(y|x) = 0.8$ 时，$odds = 4$（生成概率是不生成的4倍）；
  - 当 $P_\theta(y|x) = 0.2$ 时，$odds = 0.25$（生成概率仅为不生成的1/4）。
- 关键优势：相比概率 $P_\theta(y|x)$，赔率对“概率接近0或1”的极端情况更敏感，能更清晰地量化模型对“偏好/非偏好”的倾向性差异。

（3）赔率比：偏好与非偏好的直接对比

ORPO通过“偏好响应 y_w 与非偏好响应 y_l 的赔率比”，定义两者的优先级：

$$OR_\theta\left(y_w, y_l\right) = \frac{odds_\theta\left(y_w|x\right)}{odds_\theta\left(y_l|x\right)}$$

- 含义：若 $OR_\theta(y_w, y_l) = 10$，则模型生成偏好响应 y_w 的倾向性是生成非偏好响应 y_l 的10倍——该比值越大，偏好对齐效果越好。

3. 为什么选赔率比（OR）而非概率比（PR）？

传统偏好对齐方法（如DPO）常用“概率比”（$PR_\theta(y_w, y_l) = \frac{P_\theta(y_w|x)}{P_\theta(y_l|x)}$），但ORPO选择赔率比，核心原因是赔率比的稳定性更高，避免对非偏好响应过度抑制，具体可从“分布特性”和“实证结果”两方面说明：

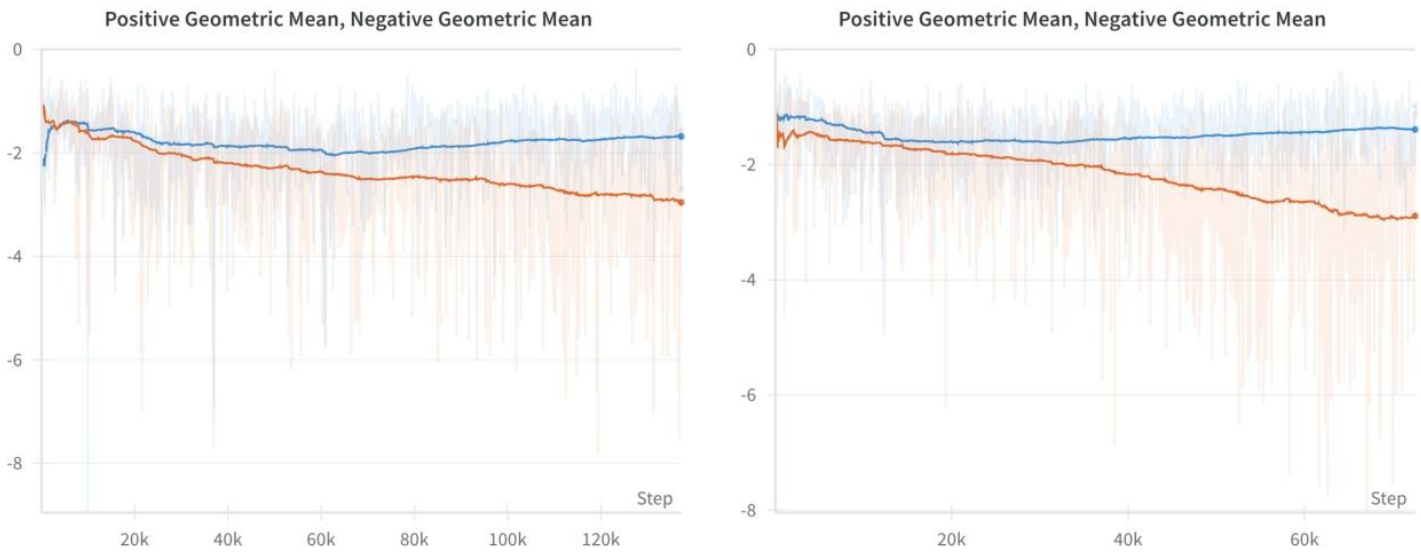
（1）理论：赔率比的分布更平缓，避免极端惩罚

通过5万组随机采样对比了“对数概率比”与“对数赔率比”的分布：

- 概率比（PR）：对数PR的分布极“尖锐”——当 $P_\theta(y_w|x)$ 与 $P_\theta(y_l|x)$ 有微小差异时，对数PR会快速趋近于正无穷或负无穷，导致模型对非偏好响应施加“极端惩罚”（如直接将非偏好响应的概率压至接近0）。
- 赔率比（OR）：对数OR的分布更“平缓”——即使 $P_\theta(y_w|x)$ 与 $P_\theta(y_l|x)$ 差异较小，对数OR也能稳定反映两者的倾向性差异，避免过度抑制非偏好响应的合理token（如对话中的常用连接词）。

（2）实证：赔率比避免生成退化

消融实验验证了这一优势：



- 用概率比训练：非偏好响应的对数概率快速降至-4以下，模型生成的响应出现“退化”（如词汇重复、语义断裂），因为过度抑制了非偏好响应中的合理token。
- 用赔率比训练：非偏好响应的对数概率缓慢下降，且始终保持在合理范围（-2.5~-2.0），模型既能区分偏好/非偏好，又不会丢失领域适配能力。

（3）场景适配：SFT阶段更需温和的偏好区分

ORPO的核心创新是“在SFT阶段融合偏好对齐”，而SFT阶段模型尚未完全适配目标领域——若使用概率比的极端惩罚，会导致模型“未学会正确生成，先被禁止生成”（如对话逻辑未掌握就过度抑制

非偏好响应）。赔率比的温和惩罚，能在保留SFT领域适配能力的同时，逐步引导模型学习偏好，避免训练崩溃。

三、ORPO的梯度解析：动态惩罚与自适应优化

ORPO的梯度设计进一步解释了其“如何实现偏好对齐”——通过梯度的双组件 $\delta(d)$ 和 $h(d)$，动态调整参数更新方向，确保模型优先学习偏好响应、抑制非偏好响应。

1. ORPO梯度的数学形式

ORPO中 $\mathcal{L}_{OR}$ 的梯度可分解为“惩罚系数”与“梯度对比项”的乘积：

$$\nabla_\theta \mathcal{L}_{OR} = \delta(d) \cdot h(d)$$

其中：

（1）惩罚系数 $\delta(d)$：动态调整惩罚强度

$$\delta(d) = \left[1 + \frac{odds_\theta(y_w|x)}{odds_\theta(y_l|x)}\right]^{-1}$$

- 含义：根据“偏好/非偏好响应的赔率比”自适应调整惩罚强度：
  - 当 $odds_\theta(y_w|x) \gg odds_\theta(y_l|x)$（模型已偏好 y_w）：$\delta(d)$ 趋近于0，梯度贡献减小，避免过度更新。
  - 当 $odds_\theta(y_w|x) \ll odds_\theta(y_l|x)$（模型偏向 y_l）：$\delta(d)$ 趋近于1，梯度贡献增大，加速参数更新以抑制 y_l

（2）梯度对比项 $h(d)$：强化偏好响应的梯度

$$h(d) = \frac{\nabla_\theta log P_\theta(y_w|x)}{1 - P_\theta(y_w|x)} - \frac{\nabla_\theta log P_\theta(y_l|x)}{1 - P_\theta(y_l|x)}$$

- 核心逻辑：通过 $1 - P_\theta(y|x)$（“不生成该响应的概率”）对梯度进行加权，实现“薄弱环节优先优化”：
  - 对偏好响应 y_w：若 $P_\theta(y_w|x)$ 低（模型尚未掌握 y_w 的生成风格），则 $1 - P_\theta(y_w|x)$ 大，梯度被“放大”，加速模型学习 y_w。
  - 对非偏好响应 y_l：若 $P_\theta(y_l|x)$ 高（模型仍倾向于生成 y_l），则 $1 - P_\theta(y_l|x)$ 小，梯度被“缩小”，抑制模型学习 y_l。

2. 梯度的推导

$$\mathcal{L}_{OR} = -log\sigma(OR_\theta\ (y_w, y_l)) = -log\sigma(log\frac{odds_\theta(y_w|x)}{odds_\theta(y_l|x)})$$

$$\begin{aligned}
\bigtriangledown_\theta \mathcal{L}_{OR} &= \bigtriangledown_\theta log\sigma(log(OR)) && (24)\\
&= \frac{\sigma'(log(OR))}{\sigma(log(OR))} && \text{（对最外层求导）} \quad (25)\\
&= \frac{\sigma(log(OR))(1-\sigma(log(OR)))}{\sigma(log(OR))} \bigtriangledown_\theta\, log(OR) && \sigma'(x)=\sigma(x)(1-\sigma(x)) \quad (26)\\
&= \sigma(-log(OR)) \bigtriangledown_\theta\, log(OR) && \sigma(-x)=1-\sigma(x) \quad (27)\\
&= \frac{1}{1+OR} \bigtriangledown_\theta\, log(OR) && \text{（前面部分代入Sigmoid）} \quad (28)
\end{aligned}$$

对于第二部分

$$\begin{aligned}
\bigtriangledown_\theta log(OR) &= \bigtriangledown_\theta log(\frac{odds_\theta(y_w|x)}{odds_\theta(y_l|x)}) && (29)\\
&= \bigtriangledown_\theta log(odds_\theta(y_w|x)) - \bigtriangledown_\theta log(odds_\theta(y_l|x)) && (30)
\end{aligned}$$

先来考虑

$$\begin{aligned}
\bigtriangledown_\theta log(odds_\theta(y|x)) &= \bigtriangledown_\theta log(P_\theta(y|x)) - \bigtriangledown_\theta log(1-P_\theta(y|x)) && (31)\\
&= \bigtriangledown_\theta log(P_\theta(y|x)) + \frac{\bigtriangledown_\theta P_\theta(y|x)}{1-P_\theta(y|x)} && \frac{d}{d\theta}[log(1-u_\theta)] = \frac{-\frac{du}{d\theta}}{1-u_\theta} \quad (32)\\
&= \bigtriangledown_\theta log(P_\theta(y|x)) + \frac{P_\theta(y|x)\bigtriangledown_\theta\, log(P_\theta(y|x))}{1-P_\theta(y|x)} && \frac{du}{d\theta} = u\frac{d}{d\theta}logu \quad (33)\\
&= \bigtriangledown_\theta log(P_\theta(y|x))[1+odds_\theta(y|x))] && \text{（代入odds定义，合并同类项）} \quad (34)\\
&= \frac{\bigtriangledown_\theta log(P_\theta(y|x))}{1-P_\theta(y|x)} && 1+odds = \frac{1-P+P}{1-P} = \frac{1}{1-P} \quad (35)
\end{aligned}$$

所以

$$\bigtriangledown_\theta log(OR) = \frac{\bigtriangledown_\theta log(P_\theta(y_w|x))}{1-P_\theta(y_w|x)} - \frac{\bigtriangledown_\theta log(P_\theta(y_l|x))}{1-P_\theta(y_l|x)}$$

$$\bigtriangledown_\theta \mathcal{L}_{OR} = [1+\frac{odds_\theta(y_w|x)}{odds_\theta(y_l|x)}]^{-1}[\frac{\bigtriangledown_\theta log(P_\theta(y_w|x))}{1-P_\theta(y_w|x)} - \frac{\bigtriangledown_\theta log(P_\theta(y_l|x))}{1-P_\theta(y_l|x)}]$$

3. 梯度的直观效果：动态平衡偏好与领域适配

展示了ORPO训练过程中“偏好/非偏好响应的对数概率”与“对数赔率比”的变化：

- 偏好响应 y_w ：对数概率保持稳定（与纯SFT相当），确保领域适配能力不丢失。
- 非偏好响应 y_l ：对数概率逐步下降，且下降幅度随训练步数增大——梯度的动态惩罚生效。
- 对数赔率比：持续上升，说明模型对 y_w 的倾向性越来越强，偏好对齐目标达成。

四、关键参数 λ 的影响：平衡偏好强度与生成质量

ORPO的权重参数 λ 控制 $\mathcal{L}_{OR}$ 的贡献，直接影响偏好对齐强度与生成质量，消融实验揭示了其规律：

1. 对对数概率的影响

- $\lambda = 0.1$ （弱偏好对齐）：偏好与非偏好响应的对数概率差距小，非偏好响应的概率未明显下降——适合对生成多样性要求高的场景（如创意写作）。
- $\lambda = 0.5$ （中等偏好对齐）：偏好响应的概率上升，非偏好响应的概率缓慢下降，两者差距适中——兼顾偏好对齐与生成自然度。
- $\lambda = 1.0$ （强偏好对齐）：偏好与非偏好响应的概率差距显著增大，但偏好响应的概率也略有下降——过度聚焦偏好，可能丢失部分领域适配能力。

2. 对下游任务的影响（MT-Bench）

- 高 λ （如1.0）：在“STEM、人文、角色扮演”等开放生成任务中表现更好（ annotator更偏好开放、流畅的响应）。
- 低 λ （如0.1）：在“数学、推理、信息提取”等确定性任务中表现更好（避免过度抑制非偏好响应中的合理逻辑token，如数学公式、推理步骤）。

3. 需根据任务类型调整

- 开放生成任务（如对话、创作）：选择较高 λ （0.5~1.0），强化偏好对齐以提升响应流畅度。
- 确定性任务（如数学、代码）：选择较低 λ （0.1~0.2），避免过度抑制导致逻辑断裂。

ORPO在无需参考模型的单阶段训练中，实现了“领域适配+偏好对齐”的双重目标，在AlpacaEval 2.0（12.20% Win Rate）、MT-Bench（7.32分）等 benchmarks 上超越了更大规模的RLHF/DPO模型，验证了其有效性与效率。