RLHF in Large Language Models

Fanyu Fang

School of Mathematics and Statistics
Wuhan University

November 21, 2024

Human Alignment

- LLMs are likely to express unintended behaviors.

Human Alignment

- LLMs are likely to express unintended behaviors.
- It is necessary to align LLMs with human values, e.g., helpful, honest, and harmless (3H).

Human Alignment

- LLMs are likely to express unintended behaviors.
- It is necessary to align LLMs with human values, e.g., helpful, honest, and harmless (3H).
- OpenAI and Anthropic have verified that RLHF is a valid avenue for aligning language models with user intent on a wide range of tasks.

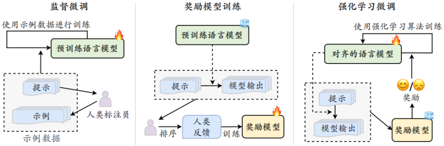


图 1: 基于人类反馈的强化学习的工作流程

The Reinforcement Learning Framework

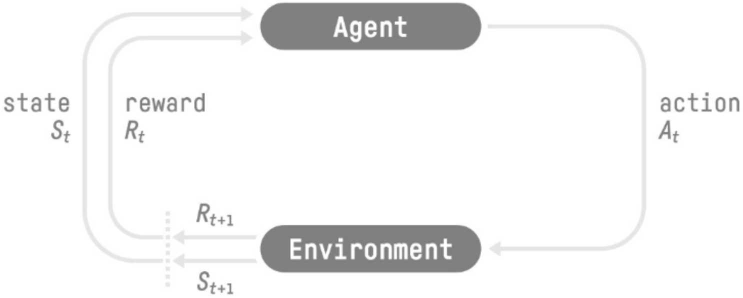
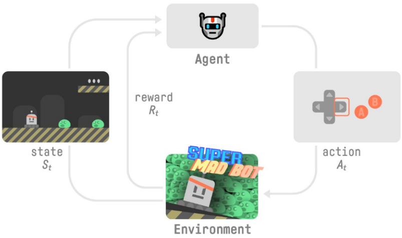
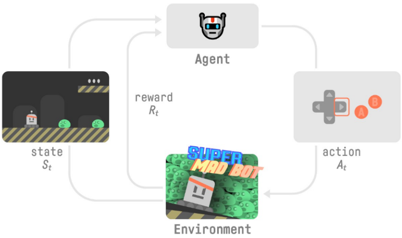


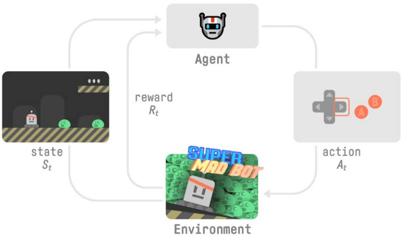
图 2: The RL Framework



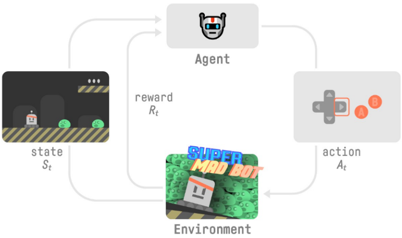
- Our Agent receives state S_0 from the Environment — we receive the first frame of our game (Environment).



- Our Agent receives state S_0 from the Environment — we receive the first frame of our game (Environment).
- Based on that state S_0, the Agent takes action A_0 — our Agent will move to the right.



- Our Agent receives state S_0 from the Environment — we receive the first frame of our game (Environment).
- Based on that state S_0, the Agent takes action A_0 — our Agent will move to the right.
- The environment goes to a new state S_1 — new frame.



- Our Agent receives state S_0 from the Environment — we receive the first frame of our game (Environment).
- Based on that state S_0, the Agent takes action A_0 — our Agent will move to the right.
- The environment goes to a new state S_1 — new frame.
- The environment gives some reward R_1 to the Agent — we're not dead (Positive Reward $+1$).

Definition

Markov Decision Processes (MDPs). An MDP is a 5-tuple, $\langle S, A, R, P, \rho_0 \rangle$, where

- S is the set of all valid states,

Definition

Markov Decision Processes (MDPs). An MDP is a 5-tuple, $\langle S, A, R, P, \rho_0 \rangle$, where

- S is the set of all valid states,
- A is the set of all valid actions,

Definition

Markov Decision Processes (MDPs). An MDP is a 5-tuple, $\langle S, A, R, P, \rho_0 \rangle$, where

- S is the set of all valid states,
- A is the set of all valid actions,
- $R : S \times A \times S \to \mathbb{R}$ is the reward function, with $r_{t+1} = R(s_t, a_t, s_{t+1})$,

Definition

Markov Decision Processes (MDPs). An MDP is a 5-tuple, $\langle S, A, R, P, \rho_0 \rangle$, where

- S is the set of all valid states,
- A is the set of all valid actions,
- $R : S \times A \times S \to \mathbb{R}$ is the reward function, with $r_{t+1} = R(s_t, a_t, s_{t+1})$,
- $P : S \times A \to \mathcal{P}(S)$ is the transition probability function, with $P(s'|s, a)$ being the probability of transitioning into state s' if you start in state s and take action a,

Definition

Markov Decision Processes (MDPs). An MDP is a 5-tuple, $\langle S, A, R, P, \rho_0 \rangle$, where

- S is the set of all valid states,
- A is the set of all valid actions,
- $R : S \times A \times S \to \mathbb{R}$ is the reward function, with $r_{t+1} = R(s_t, a_t, s_{t+1})$,
- $P : S \times A \to \mathcal{P}(S)$ is the transition probability function, with $P(s'|s, a)$ being the probability of transitioning into state s' if you start in state s and take action a,
- and ρ_0 is the starting state distribution.

Definition

- A policy can be deterministic, in which case it is denoted by μ:

$$a_t = \mu(s_t)$$

or it may be stochastic, in which case it is denoted by π:

$$a_t \sim \pi(\cdot|s_t)$$

Definition

- A policy can be deterministic, in which case it is denoted by μ:

$$a_t = \mu(s_t)$$

or it may be stochastic, in which case it is denoted by π:

$$a_t \sim \pi(\cdot|s_t)$$

-

$$G_\tau^\pi \triangleq \sum_{t=\tau}^{T} \gamma^{k-1} R_t$$

Definition

- A policy can be deterministic, in which case it is denoted by μ:

$$a_t = \mu(s_t)$$

or it may be stochastic, in which case it is denoted by π:

$$a_t \sim \pi(\cdot|s_t)$$

- $$G_\tau^\pi \triangleq \sum_{t=\tau}^{T} \gamma^{k-1} R_t$$

- $$V^\pi(x) \triangleq \mathbb{E}\left[\sum_{t=1}^{T} \gamma^{t-1} R_t \mid X_1 = x\right]$$

Definition

- A policy can be deterministic, in which case it is denoted by μ:
$$a_t = \mu(s_t)$$
or it may be stochastic, in which case it is denoted by π:
$$a_t \sim \pi(\cdot|s_t)$$
- $$G_\tau^\pi \triangleq \sum_{t=\tau}^{T} \gamma^{k-1} R_t$$
- $$V^\pi(x) \triangleq \mathbb{E}\left[\sum_{t=1}^{T} \gamma^{t-1} R_t \mid X_1 = x\right]$$
- $$Q^\pi(x,a) \triangleq \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} R_t \mid X_1 = x, A_1 = a\right]$$

- 状态 (s)：文本生成过程中的某一步的上下文。

- 状态 (s)：文本生成过程中的某一步的上下文。
- 动作 (a)：模型在当前步骤生成的输出，例如一个词或一个标记（Token）。

- 状态 (s)：文本生成过程中的某一步的上下文。
- 动作 (a)：模型在当前步骤生成的输出，例如一个词或一个标记（Token）。
- 策略 ($\pi(a|s)$)：给定当前状态（上下文），模型生成各种可能动作（标记）的概率分布。

- 状态 (s)：文本生成过程中的某一步的上下文。
- 动作 (a)：模型在当前步骤生成的输出，例如一个词或一个标记（Token）。
- 策略 ($\pi(a|s)$)：给定当前状态（上下文），模型生成各种可能动作（标记）的概率分布。
- 奖励 (r)：由奖励模型提供的标量值，用于评估生成的动作或序列的质量。

Critical Steps

- Three main stages: supervised fine-tuning (SFT), reward model (RM) training, and proximal policy optimization (PPO) on this reward model.

Critical Steps

- Three main stages: supervised fine-tuning (SFT), reward model (RM) training, and proximal policy optimization (PPO) on this reward model.
- SFT phase： the model learns to engage in general human-like dialogues by imitating human-annotated dialogue examples.

Critical Steps

- Three main stages: supervised fine-tuning (SFT), reward model (RM) training, and proximal policy optimization (PPO) on this reward model.
- SFT phase： the model learns to engage in general human-like dialogues by imitating human-annotated dialogue examples.
- Reward model： the model learns to compare the preference of different responses based on human feedback.

Critical Steps

- Three main stages: supervised fine-tuning (SFT), reward model (RM) training, and proximal policy optimization (PPO) on this reward model.
- SFT phase： the model learns to engage in general human-like dialogues by imitating human-annotated dialogue examples.
- Reward model： the model learns to compare the preference of different responses based on human feedback.
- PPO phase： the model is updated based on feedback from the reward model, striving to discover an optimized policy through exploration and exploitation.

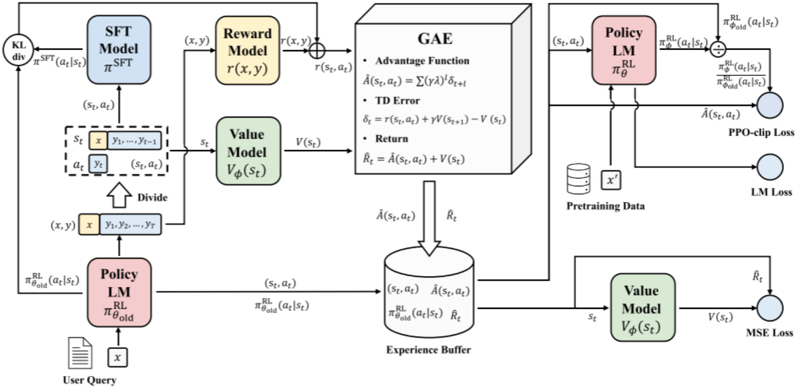


图 3: RLHF workflow

反馈数据收集

- 标注人类选择

反馈数据收集

- 标注人类选择
- 人类反馈形式：
  1、基于评分的人类反馈.

反馈数据收集

- 标注人类选择
- 人类反馈形式:
  1、基于评分的人类反馈.

反馈数据收集

- 标注人类选择
- 人类反馈形式：
  1、基于评分的人类反馈.
  2、基于排序的人类反馈.
- Reward Modeling(RM): 替代人类在 RLHF 训练过程中实时提供反馈.

反馈数据收集

- 标注人类选择
- 人类反馈形式:
  1、基于评分的人类反馈.
  2、基于排序的人类反馈.
- Reward Modeling(RM): 替代人类在 RLHF 训练过程中实时提供反馈.
- 训练方法:
  1、打分式.

反馈数据收集

- 标注人类选择
- 人类反馈形式：
  1、基于评分的人类反馈.
  2、基于排序的人类反馈.
- Reward Modeling(RM): 替代人类在 RLHF 训练过程中实时提供反馈.
- 训练方法：
  1、打分式.

反馈数据收集

- 标注人类选择
- 人类反馈形式:
  1、基于评分的人类反馈.
  2、基于排序的人类反馈.
- Reward Modeling(RM): 替代人类在 RLHF 训练过程中实时提供反馈.
- 训练方法:
  1、打分式.
  2、对比式.

反馈数据收集

- 标注人类选择
- 人类反馈形式:
  1、基于评分的人类反馈.
  2、基于排序的人类反馈.
- Reward Modeling(RM): 替代人类在 RLHF 训练过程中实时提供反馈.
- 训练方法:
  1、打分式.
  2、对比式.
  3、排序式.

Reward Modeling

- The modeling loss for each pair of preferred and dispreferred samples is:

$$\mathcal{L}(\psi) = \log \sigma \left(r(x, y_w) - r(x, y_l) \right),$$

Reward Modeling

- The modeling loss for each pair of preferred and dispreferred samples is:

$$\mathcal{L}(\psi) = \log \sigma \left(r(x, y_w) - r(x, y_l) \right),$$

- improve:

$$\begin{aligned}\mathcal{L}(\psi) = &- \lambda \mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}_{\text{rm}}} \left[\log \sigma \left(r(x, y_w) - r(x, y_l) \right) \right] \\ &+ \beta_{\text{rm}} \mathbb{E}_{(x,y_w)\sim\mathcal{D}_{\text{rm}}} \left[\log(r'(x, y_w)) \right],\end{aligned}$$

Reward Modeling

- The modeling loss for each pair of preferred and dispreferred samples is:

$$\mathcal{L}(\psi) = \log \sigma \left(r(x, y_w) - r(x, y_l) \right),$$

- improve:

$$\begin{aligned}\mathcal{L}(\psi) = & -\lambda \mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}_{\text{rm}}} \left[\log \sigma \left(r(x, y_w) - r(x, y_l) \right) \right] \\ & + \beta_{\text{rm}} \mathbb{E}_{(x,y_w)\sim\mathcal{D}_{\text{rm}}} \left[\log(r'(x, y_w)) \right],\end{aligned}$$

- Kullback-Leibler (KL) divergence:

$$r_{\text{total}} = r(x, y) - \eta \text{KL} \left(\pi_\phi^{\text{RL}}(y|x), \pi^{\text{SFT}}(y|x) \right),$$

Training Implementations for the RM

- 对于英文，从原始的 LLaMA-7B 开始，这是一种仅包含解码器的架构. 使用了 HH-RLHF 数据集中的 160k 对样本进行训练.

Training Implementations for the RM

- 对于英文，从原始的 LLaMA-7B 开始，这是一种仅包含解码器的架构. 使用了 HH-RLHF 数据集中的 160k 对样本进行训练.
- 对于中文，使用了 OpenChineseLLaMA。该模型通过在中文数据集上的增量预训练开发的，基于 LLaMA-7B 的基础，显著提高了其在中文上的理解和生成能力.

Training Implementations for the RM

- 对于英文，从原始的 LLaMA-7B 开始，这是一种仅包含解码器的架构. 使用了 HH-RLHF 数据集中的 160k 对样本进行训练.
- 对于中文，使用了 OpenChineseLLaMA。该模型通过在中文数据集上的增量预训练开发的，基于 LLaMA-7B 的基础，显著提高了其在中文上的理解和生成能力.
- 学习率设置为 5e-6，并在前 10% 的步骤中进行预热。使用动态批量方法，而不是固定值，以尽可能平衡每个批次中的标记数量，从而实现更高效和稳定的训练阶段。批量大小根据批次中的标记数量变化，最大值为 128，最小值为 4。将训练步骤固定为 1000，相当于整个训练集的约 1.06 个 epoch。设置 $\beta_{\mathrm{rm}} = 1$，这表示在整个实验中使用 LM 损失权重来训练奖励模型.

Training Performance

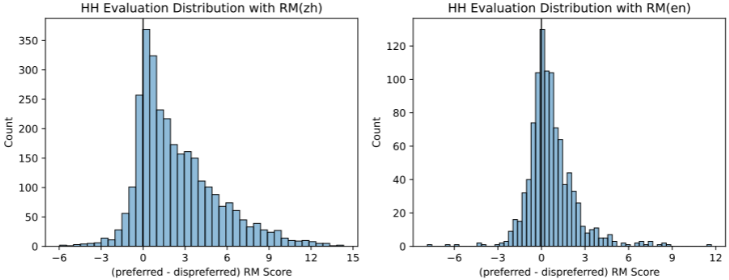


Figure 2: Histograms of the RM predictions for the HH evaluations. The left figure shows the score distribution for a PM trained on manually labeled Chinese data, while the right one shows that of HH-RLHF data. Both models roughly align with human preferences, especially the RM trained on Chinese data.

Training Performance

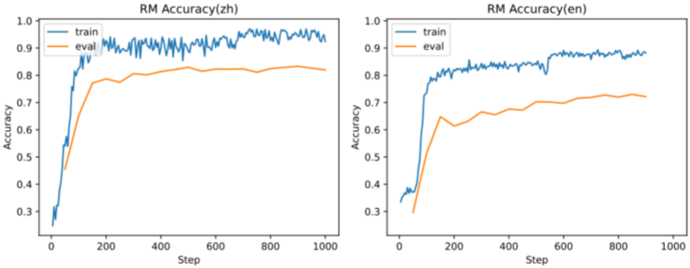


Figure 3: We show the variation of RM accuracy during training. The performance of both models steadily improves on the validation set. The RM trained on Chinese data shows a higher accuracy for the greater dissimilarity between the two responses within a pair in the Chinese data, and it becomes relatively easier for the RM to model the distinctive features between them when training and evaluating.

Policy Gradient Methods

The policy π is parameterized by θ, we denote it as $\pi(a|s,\theta)$. The update rule is:

$$\theta \leftarrow \theta + \alpha \nabla J(\theta),$$

where α is the learning rate, $J(\theta)$ represents the expected return when following policy π_θ ,$\nabla_\theta J(\theta)$ is the policy gradient.

Policy Gradient Methods

The policy π is parameterized by θ, we denote it as $\pi(a|s,\theta)$. The update rule is:

$$\theta \leftarrow \theta + \alpha \nabla J(\theta),$$

where α is the learning rate, $J(\theta)$ represents the expected return when following policy π_θ ,$\nabla_\theta J(\theta)$ is the policy gradient. A general form of policy gradient can be formulated as:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t)\Phi_t\right],$$

where Φ_t could be any of $\Phi_t = R(\tau)$ or $\Phi_t = \sum_{t'=t}^{T} R(s_{t'}, a_{t'})$ or $\Phi_t = \sum_{t'=t}^{T} R(s_{t'}, a_{t'}) - b(s_t)$ with baseline b. All of these choices lead to the same expected value for the policy gradient, despite having different variances.

Advantage Function

The advantage function $A(s_t, a_t)$ represents how much better it is to take a specific action a_t at state s_t, compared to the average quality of actions at that state under the same policy. Thus,

$$\Phi_t = A(s_t, a_t) = Q(s_t, a_t) - V(s_t).$$

Advantage Function

The advantage function $A(s_t, a_t)$ represents how much better it is to take a specific action a_t at state s_t, compared to the average quality of actions at that state under the same policy. Thus,

$$\Phi_t = A(s_t, a_t) = Q(s_t, a_t) - V(s_t).$$

The estimation methods for the advantage function vary significantly across different algorithms, we introduce Generalized Advantage Estimation (GAE).

Advantage Function

The advantage function $A(s_t, a_t)$ represents how much better it is to take a specific action a_t at state s_t, compared to the average quality of actions at that state under the same policy. Thus,

$$\Phi_t = A(s_t, a_t) = Q(s_t, a_t) - V(s_t).$$

The estimation methods for the advantage function vary significantly across different algorithms, we introduce Generalized Advantage Estimation (GAE).

The GAE algorithm uses Temporal Difference (TD) returns and full Monte Carlo returns to balance bias and variance.

The GAE algorithm

The TD-k return $\hat{R}_t^k$ is a combination of actual rewards and estimated returns:

The GAE algorithm

The TD-k return $\hat{R}_t^k$ is a combination of actual rewards and estimated returns:

$$\hat{R}_t^k = r_t + \gamma r_{t+1} + \cdots + \gamma^{(k-1)} r_{t+k-1} + \gamma^k V(s_{t+k}),$$

The GAE algorithm

The TD-k return $\hat{R}_t^k$ is a combination of actual rewards and estimated returns:

$$\hat{R}_t^k = r_t + \gamma r_{t+1} + \cdots + \gamma^{(k-1)} r_{t+k-1} + \gamma^k V(s_{t+k}),$$

where γ is the discount factor. The advantage estimate using TD-k returns is called the k-step advantage, defined as:

The GAE algorithm

The TD-k return $\hat{R}_t^k$ is a combination of actual rewards and estimated returns:

$$\hat{R}_t^k = r_t + \gamma r_{t+1} + \cdots + \gamma^{(k-1)} r_{t+k-1} + \gamma^k V(s_{t+k}),$$

where γ is the discount factor. The advantage estimate using TD-k returns is called the k-step advantage, defined as:

$$\begin{aligned} \hat{A}_t^k &= \hat{R}_t^k - V(s_t) = \sum_{l=1}^{k} \gamma^{l-1} \delta_{t+l} \\ &= -V(s_t) + r_t + \gamma r_{t+1} + \cdots + \gamma^{k-1} r_{t+k-1} + \gamma^k V(s_{t+k}), \end{aligned}$$

The GAE algorithm

The TD-k return $\hat{R}_t^k$ is a combination of actual rewards and estimated returns:

$$\hat{R}_t^k = r_t + \gamma r_{t+1} + \cdots + \gamma^{(k-1)} r_{t+k-1} + \gamma^k V(s_{t+k}),$$

where γ is the discount factor. The advantage estimate using TD-k returns is called the k-step advantage, defined as:

$$\begin{aligned}\hat{A}_t^k &= \hat{R}_t^k - V(s_t) = \sum_{l=1}^{k} \gamma^{l-1} \delta_{t+l} \\ &= -V(s_t) + r_t + \gamma r_{t+1} + \cdots + \gamma^{k-1} r_{t+k-1} + \gamma^k V(s_{t+k}),\end{aligned}$$

where $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$ is the TD error.

$$
\begin{aligned}
\hat{A}_t^{\mathsf{GAE}(\gamma,\lambda)} &= (1-\lambda)\left(\hat{A}_t^{(1)} + \lambda\hat{A}_t^{(2)} + \lambda^2\hat{A}_t^{(3)} + \dots\right) \\
&= (1-\lambda)\left(\delta_t + \lambda(\delta_t + \gamma\delta_{t+1}) + \lambda^2(\delta_t + \gamma\delta_{t+1} + \gamma^2\delta_{t+2}) + \dots\right) \\
&= (1-\lambda)\big(\delta_t(1+\lambda+\lambda^2+\dots) + \gamma\delta_{t+1}(\lambda+\lambda^2+\lambda^3+\dots) \\
&\quad + \gamma^2\delta_{t+2}(\lambda^2+\lambda^3+\lambda^4+\dots) + \dots\big) \\
&= (1-\lambda)\left(\delta_t\left(\frac{1}{1-\lambda}\right) + \gamma\delta_{t+1}\left(\frac{\lambda}{1-\lambda}\right) + \gamma^2\delta_{t+2}\left(\frac{\lambda^2}{1-\lambda}\right) + \dots\right) \\
&= \sum_{l=0}^{\infty}(\gamma\lambda)^l\delta_{t+l}.
\end{aligned}
$$

$$
\begin{aligned}
\hat{A}_t^{\mathsf{GAE}(\gamma,\lambda)} &= (1-\lambda)\left(\hat{A}_t^{(1)} + \lambda\hat{A}_t^{(2)} + \lambda^2\hat{A}_t^{(3)} + \dots\right) \\
&= (1-\lambda)\left(\delta_t + \lambda(\delta_t + \gamma\delta_{t+1}) + \lambda^2(\delta_t + \gamma\delta_{t+1} + \gamma^2\delta_{t+2}) + \dots\right) \\
&= (1-\lambda)\big(\delta_t(1+\lambda+\lambda^2+\dots) + \gamma\delta_{t+1}(\lambda+\lambda^2+\lambda^3+\dots) \\
&\quad + \gamma^2\delta_{t+2}(\lambda^2+\lambda^3+\lambda^4+\dots) + \dots\big) \\
&= (1-\lambda)\left(\delta_t\left(\frac{1}{1-\lambda}\right) + \gamma\delta_{t+1}\left(\frac{\lambda}{1-\lambda}\right) + \gamma^2\delta_{t+2}\left(\frac{\lambda^2}{1-\lambda}\right) + \dots\right) \\
&= \sum_{l=0}^{\infty}(\gamma\lambda)^l\delta_{t+l}.
\end{aligned}
$$

$$
\mathsf{GAE}(\gamma,0): \hat{A}_t = \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t).
$$

$$
\begin{aligned}
\hat{A}_t^{\mathsf{GAE}(\gamma,\lambda)} &= (1-\lambda)\left(\hat{A}_t^{(1)} + \lambda\hat{A}_t^{(2)} + \lambda^2\hat{A}_t^{(3)} + \dots\right) \\
&= (1-\lambda)\left(\delta_t + \lambda(\delta_t + \gamma\delta_{t+1}) + \lambda^2(\delta_t + \gamma\delta_{t+1} + \gamma^2\delta_{t+2}) + \dots\right) \\
&= (1-\lambda)\big(\delta_t(1+\lambda+\lambda^2+\dots) + \gamma\delta_{t+1}(\lambda+\lambda^2+\lambda^3+\dots) \\
&\quad + \gamma^2\delta_{t+2}(\lambda^2+\lambda^3+\lambda^4+\dots) + \dots\big) \\
&= (1-\lambda)\left(\delta_t\left(\frac{1}{1-\lambda}\right) + \gamma\delta_{t+1}\left(\frac{\lambda}{1-\lambda}\right) + \gamma^2\delta_{t+2}\left(\frac{\lambda^2}{1-\lambda}\right) + \dots\right) \\
&= \sum_{l=0}^{\infty}(\gamma\lambda)^l\delta_{t+l}.
\end{aligned}
$$

$$
\mathsf{GAE}(\gamma,0): \hat{A}_t = \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t).
$$

$$
\mathsf{GAE}(\gamma,1): \hat{A}_t = \sum_{l=0}^{\infty}\gamma^l\delta_{t+l} = \sum_{l=0}^{\infty}\gamma^l r_{t+l+1} - V(s_t).
$$

Through GAE, we can estimate $\hat{A}_t$ of the advantage function $A(s_t, a_t)$ accurately. This estimate will play a crucial role in constructing a policy gradient estimator:

Through GAE, we can estimate $\hat{A}_t$ of the advantage function $A(s_t, a_t)$ accurately. This estimate will play a crucial role in constructing a policy gradient estimator:

$$\nabla_\theta \hat{J}(\theta) = \frac{1}{|\mathcal{D}|} \sum_{\tau \in \mathcal{D}} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \hat{A}_t,$$

Through GAE, we can estimate $\hat{A}_t$ of the advantage function $A(s_t, a_t)$ accurately. This estimate will play a crucial role in constructing a policy gradient estimator:

$$\nabla_\theta \hat{J}(\theta) = \frac{1}{|\mathcal{D}|} \sum_{\tau \in \mathcal{D}} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \hat{A}_t,$$

where $\mathcal{D}$ is a finite batch of samples, we will use $\hat{\mathbb{E}}_t$ to represent the aforementioned $\frac{1}{|\mathcal{D}|} \sum_{\tau \in \mathcal{D}} \sum_{t=1}^{T}$.

Through GAE, we can estimate $\hat{A}_t$ of the advantage function $A(s_t, a_t)$ accurately. This estimate will play a crucial role in constructing a policy gradient estimator:

$$\nabla_\theta \hat{J}(\theta) = \frac{1}{|\mathcal{D}|} \sum_{\tau \in \mathcal{D}} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \hat{A}_t,$$

where $\mathcal{D}$ is a finite batch of samples, we will use $\hat{\mathbb{E}}_t$ to represent the aforementioned $\frac{1}{|\mathcal{D}|} \sum_{\tau \in \mathcal{D}} \sum_{t=1}^{T}$.
Value Function Estimation：

$$\mathcal{L}_{\text{critic}}(\phi) = \hat{\mathbb{E}}_t \left[\left\| V_\phi(s_t) - \hat{R}_t \right\|^2 \right].$$

Here, $V_\phi(s_t)$ represents the critic model's predicted value for state s_t with parameters ϕ, and $\hat{R}_t$ represents the actual return value for state s_t, which can be estimated as:

$$\hat{R}_t = \sum_{l=0}^{\infty} \gamma^l r_{t+l},$$

where γ is the discount factor.

Proximal Policy Optimization

- TRPO:KL divergence

$$\max_{\theta} \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t \right],$$

$$\text{subject to } \hat{\mathbb{E}}_t \left[\mathrm{KL}(\pi_{\theta_{\text{old}}}(\cdot|s_t), \pi_\theta(\cdot|s_t)) \right] \leq \delta,$$

Proximal Policy Optimization

- TRPO:KL divergence

$$\max_{\theta} \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t \right],$$

$$\text{subject to } \hat{\mathbb{E}}_t \left[\mathrm{KL}(\pi_{\theta_{\text{old}}}(\cdot|s_t), \pi_\theta(\cdot|s_t)) \right] \leq \delta,$$

- PPO-Penalty:

$$\mathcal{L}_{\text{ppo-penalty}}(\theta) = \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t \right] - \beta\, \mathrm{KL}(\pi_{\theta_{\text{old}}}(\cdot|s_t), \pi_\theta(\cdot|s_t)),$$

Proximal Policy Optimization

- TRPO:KL divergence

$$\max_{\theta} \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t \right],$$

$$\text{subject to } \hat{\mathbb{E}}_t \left[\text{KL}(\pi_{\theta_{\text{old}}}(\cdot|s_t), \pi_\theta(\cdot|s_t)) \right] \leq \delta,$$

- PPO-Penalty:

$$\mathcal{L}_{\text{ppo-penalty}}(\theta) = \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t \right] - \beta \, \text{KL}(\pi_{\theta_{\text{old}}}(\cdot|s_t), \pi_\theta(\cdot|s_t)),$$

- PPO-Clip:

$$\mathcal{L}_{\text{ppo-clip}}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t, \ \text{clip} \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon \right) \hat{A}_t \right) \right]$$

Algorithm 1 PPO

Input: initial policy parameters θ_0, initial value function parameters ϕ_0.

1: **for** $n = 0, 1, 2, \ldots$ **do**

2: Collect a set of trajectories $\mathcal{D}_n = \{\tau_i\}$ by executing policy $\pi(\theta_n)$ within the environment.

3: Compute rewards-to-go $\hat{R}_t$.

4: Compute advantage estimates, $\hat{A}_t$ (using any advantage estimation method) based on the current value function V_{ϕ_n}.

5: Update the policy by maximizing the PPO-penalty/clip objective:

$$\theta_{n+1} = \arg\max_{\theta} \mathcal{L}_{\text{ppo-clip}}(\theta_n).$$

6: Update the value function by regression on mean-squared error:

$$\phi_{n+1} = \arg\min_{\phi} \mathcal{L}_{\text{critic}}(\phi_n).$$

7: **end for**

Output: Optimized policy parameters θ

Models and Training Setup

- Reference Model and Policy Model：用 7B SFT(监督微调) 模型初始化，SFT 模型基于 OpenChineseLLaMA 对 100 万条筛选后的指令数据进行了 2 个 epoch 的监督微调，这些数据包括 40 万单轮指令样本和 60 万多轮指令样本。学习率设置为 9.5×10^{-6}，并采用余弦学习率调度，最终学习率会衰减至峰值学习率的 10%。全局批量大小设置为 1024。

Models and Training Setup

- Reference Model and Policy Model：用 7B SFT(监督微调) 模型初始化，SFT 模型基于 OpenChineseLLaMA 对 100 万条筛选后的指令数据进行了 2 个 epoch 的监督微调，这些数据包括 40 万单轮指令样本和 60 万多轮指令样本。学习率设置为 9.5×10^{-6}，并采用余弦学习率调度，最终学习率会衰减至峰值学习率的 10%。全局批量大小设置为 1024。
- Critic Model and Reward Model: 用奖励模型初始化.

Models and Training Setup

- Reference Model and Policy Model：用 7B SFT(监督微调) 模型初始化，SFT 模型基于 OpenChineseLLaMA 对 100 万条筛选后的指令数据进行了 2 个 epoch 的监督微调，这些数据包括 40 万单轮指令样本和 60 万多轮指令样本。学习率设置为 9.5×10^{-6}，并采用余弦学习率调度，最终学习率会衰减至峰值学习率的 10%。全局批量大小设置为 1024。
- Critic Model and Reward Model: 用奖励模型初始化.
- 一个手动构建的 HH 数据集上训练模型，该数据集包含 8k 无害查询和 20k 有帮助的查询，并且固定训练步数。在实验中，将从环境中采样的批量大小设置为 128，用于训练策略模型和评价模型的批量大小为 32。策略模型和评价模型的学习率分别设置为 5×10^{-7} 和 1.65×10^{-6}，并在前 10% 的训练步数中进行预热。

Models and Training Setup

- Reference Model and Policy Model：用 7B SFT(监督微调) 模型初始化，SFT 模型基于 OpenChineseLLaMA 对 100 万条筛选后的指令数据进行了 2 个 epoch 的监督微调，这些数据包括 40 万单轮指令样本和 60 万多轮指令样本。学习率设置为 9.5×10^{-6}，并采用余弦学习率调度，最终学习率会衰减至峰值学习率的 10%。全局批量大小设置为 1024。
- Critic Model and Reward Model: 用奖励模型初始化.
- 一个手动构建的 HH 数据集上训练模型，该数据集包含 8k 无害查询和 20k 有帮助的查询，并且固定训练步数。在实验中，将从环境中采样的批量大小设置为 128，用于训练策略模型和评价模型的批量大小为 32。策略模型和评价模型的学习率分别设置为 5×10^{-7} 和 1.65×10^{-6}，并在前 10% 的训练步数中进行预热。
- 所有实验均在相同配置的机器上进行。每台机器包含 8 个 80G A100 GPU、1TB 内存和 128 个 CPU。在训练阶段，使用 ZERO2 和梯度检查点（gradient checkpoint）以减少 GPU 内存开销。

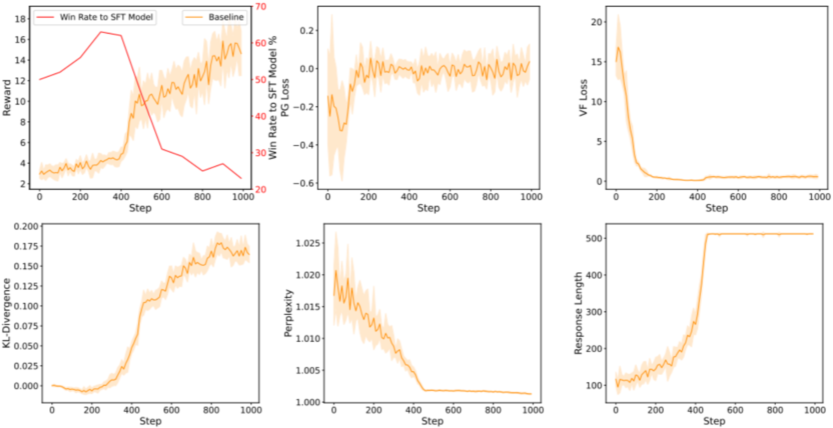


Figure 4: **(Top)** We show the response reward and training loss under vanilla PPO implementation. The red line in the first sub-figure shows the win rate of policy model response compared to SFT model response. **(Bottom)** Informative metrics for the collapse problem in PPO training, we observe significant variation in these metrics when there was a misalign between the human evaluation results and reward scores.

Score Reparameterization

- $r_n(x, y)$ denote the results of per-batch reward;

Score Reparameterization

- $r_n(x, y)$ denote the results of per-batch reward;
- $\{r(x, y)\} \triangleq \{r_n(x, y)\}_{n=1}^{B}$ denote a reward sequence in training;

Score Reparameterization

- $r_n(x, y)$ denote the results of per-batch reward;
- $\{r(x, y)\} \triangleq \{r_n(x, y)\}_{n=1}^{B}$ denote a reward sequence in training;
- $\sigma(A)$and $\bar{A}$ denote the mean and standard deviation of variable A.

Score Reparameterization

- $r_n(x, y)$ denote the results of per-batch reward;
- $\{r(x, y)\} \triangleq \{r_n(x, y)\}_{n=1}^{B}$ denote a reward sequence in training;
- $\sigma(A)$and $\bar{A}$ denote the mean and standard deviation of variable A.
- Reward Scaling:

$$\frac{r_n(x, y)}{\sigma(r(x, y))}.$$

Score Reparameterization

- $r_n(x, y)$ denote the results of per-batch reward;
- $\{r(x, y)\} \triangleq \{r_n(x, y)\}_{n=1}^{B}$ denote a reward sequence in training;
- $\sigma(A)$and $\bar{A}$ denote the mean and standard deviation of variable A.
- Reward Scaling:

$$\frac{r_n(x, y)}{\sigma(r(x, y))}.$$

- Reward Normalization and Clipping:

$$\tilde{r}(x, y) = \text{clip}\left(\frac{r_n(x, y) - \bar{r}(x, y)}{\sigma(r(x, y))}, -\delta, \delta\right).$$

Score Reparameterization

- $r_n(x, y)$ denote the results of per-batch reward;
- $\{r(x, y)\} \triangleq \{r_n(x, y)\}_{n=1}^{B}$ denote a reward sequence in training;
- $\sigma(A)$and $\bar{A}$ denote the mean and standard deviation of variable A.
- Reward Scaling:

$$\frac{r_n(x, y)}{\sigma(r(x, y))}.$$

- Reward Normalization and Clipping:

$$\tilde{r}(x, y) = \text{clip}\left(\frac{r_n(x, y) - \bar{r}(x, y)}{\sigma(r(x, y))}, -\delta, \delta\right).$$

- Advantages Normalization and Clipping: subtracting its mean and dividing it by its standard deviation.

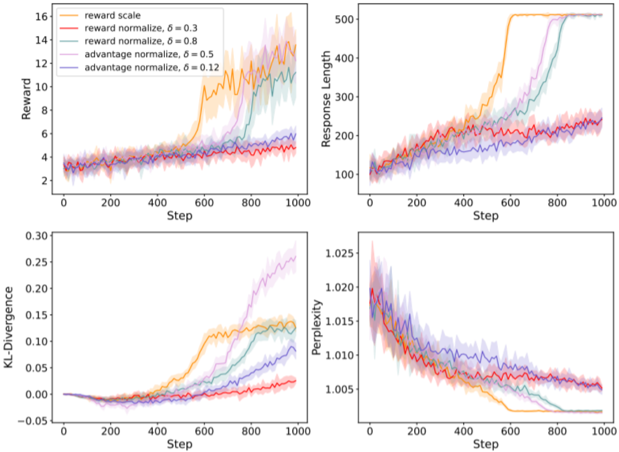


Figure 6: We show the variation of training metrics when constraining the fluctuations of intermediate variables. δ indicates the clipped range, the KL divergence indicates the optimization magnitude of policy model, and the perplexity indicates the uncertainty of policy model for current response. Scaling or clipping strategy for reward and advantage contributes to the training stability compared to vanilla PPO. Temporarily stable settings, such as reward normalize with $\delta = 0.3$, also exhibit consistent upward trends across metrics, which implies that pattern collapse problems likewise occur when training longer.

Policy Constraints

- Token Level KL-Penalty:

$$r_{\text{total}}(x, y_i) = r(x, y_i) - \eta \mathrm{KL}(\pi_\theta^{\mathrm{RL}}(y_i|x), \pi^{\mathrm{SFT}}(y_i|x)).$$

Policy Constraints

- Token Level KL-Penalty:

$$r_{\text{total}}(x, y_i) = r(x, y_i) - \eta \text{KL}(\pi_\theta^{\text{RL}}(y_i|x), \pi^{\text{SFT}}(y_i|x)).$$

- Importance Sampling： rectify the policy divergence between the historical generative model and current model when optimizing policy model with responses in the experience buffer.

Policy Constraints

- Token Level KL-Penalty:

$$r_{\text{total}}(x, y_i) = r(x, y_i) - \eta \mathrm{KL}(\pi_\theta^{\mathrm{RL}}(y_i|x), \pi^{\mathrm{SFT}}(y_i|x)).$$

- Importance Sampling: rectify the policy divergence between the historical generative model and current model when optimizing policy model with responses in the experience buffer.
-

$$\begin{aligned}\mathbb{E}_{x\sim q}[f(x)] &= \int q(x)\cdot f(x)\,dx = \int \frac{p(x)}{p(x)}\cdot q(x)\cdot f(x)\,dx \\ &= \int p(x)\cdot\left[\frac{q(x)}{p(x)}\cdot f(x)\right]dx \\ &= \mathbb{E}_{x\sim p}\left[\frac{q(x)}{p(x)}\cdot f(x)\right],\end{aligned}$$

Policy Constraints

- Token Level KL-Penalty:

$$r_{\text{total}}(x, y_i) = r(x, y_i) - \eta \text{KL}(\pi_\theta^{\text{RL}}(y_i|x), \pi^{\text{SFT}}(y_i|x)).$$

- Importance Sampling： rectify the policy divergence between the historical generative model and current model when optimizing policy model with responses in the experience buffer.
-

$$\begin{aligned}\mathbb{E}_{x\sim q}[f(x)] &= \int q(x)\cdot f(x)\,dx = \int \frac{p(x)}{p(x)}\cdot q(x)\cdot f(x)\,dx \\ &= \int p(x)\cdot\left[\frac{q(x)}{p(x)}\cdot f(x)\right]\,dx \\ &= \mathbb{E}_{x\sim p}\left[\frac{q(x)}{p(x)}\cdot f(x)\right],\end{aligned}$$

- Entropy Bonus：

$$H(\pi) = -\sum_a \pi(a|s)\log\pi(a|s).$$

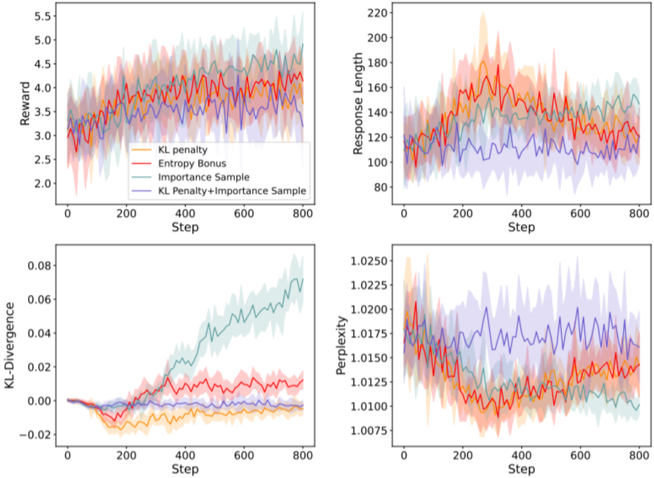


Figure 7: Training dynamics when using different methods to constrain the policy optimization. We show that all modifications can induce convergence, but only a penalty of the policy entropy or KL divergence can provide a long-lasting stable optimization. It is worth noting that all methods (including those shown in Sec 5.3.1) exhibit consistent variation in response length and perplexity in the early training period, which may imply some bias in the reward model preference.

Pretrained Initialization

- Critic Model Initialization： SFT

Pretrained Initialization

- Critic Model Initialization： SFT
- Policy Model Initialization： pre-train

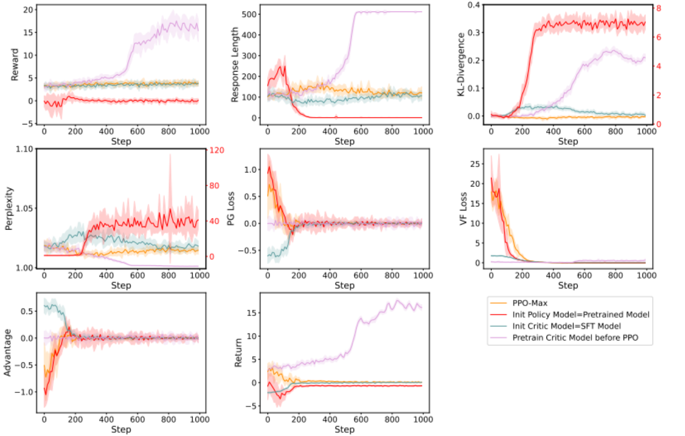


Figure 8: We show the necessity regarding supervised fine-tuning (SFT) of the policy model and the non-necessity regarding specific initialization of critic model. In the subfigure about KL-divergence and perplexity, the right axis represents the result under initiating policy model without SFT. It's a reduction to RLHF process when initializing the critic model with SFT model or omitting the fine-tuning process on policy model, so we experiment with these changes on the basis of PPO-max. Pre-training the critic model introduced additional processing to PPO, so this experiment was conducted on the basis of vanilla PPO setup.

PPO-max Setup

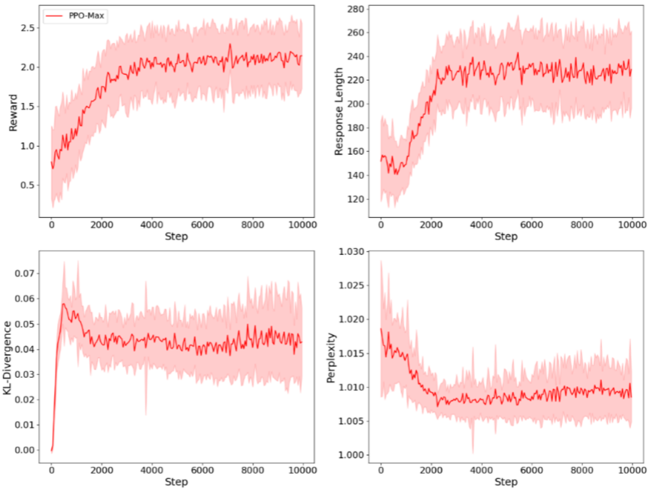


Figure 9: 10K steps training dynamics of PPO-max. PPO-max ensures long-term stable policy optimization for the model.

基于过程监督的 RLHF

- 监督信号：结果监督信号和过程监督信号。

基于过程监督的 RLHF

- 监督信号：结果监督信号和过程监督信号。
- 数据集：带有细粒度标注信息的数据集合 PRM800K。

基于过程监督的 RLHF

- 监督信号：结果监督信号和过程监督信号。
- 数据集：带有细粒度标注信息的数据集合 PRM800K。
- RLHF 训练方法：专家迭代方法：通过向专家策略学习进而改进基础策略的强化学习方法。包含两个主要阶段：策略改进和蒸馏。在策略改进阶段，专家策略进行广泛的搜索并生成样本，过程监督奖励模型引导专家策略在搜索过程中生成高质量的样本。具体来说，在专家策略搜索的过程中，过程监督奖励模型基于当前的状态和决策轨迹，对专家策略的下一步决策进行打分，辅助专家策略选取更好的决策（即分数更高的决策）。随后，在蒸馏阶段，进一步使用第一阶段由专家策略生成的样本对基础策略（即待对齐的语言模型）进行监督微调。

基于过程监督的 RLHF

- 监督信号：结果监督信号和过程监督信号。
- 数据集：带有细粒度标注信息的数据集合 PRM800K。
- RLHF 训练方法：专家迭代方法：通过向专家策略学习进而改进基础策略的强化学习方法。包含两个主要阶段：策略改进和蒸馏。在策略改进阶段，专家策略进行广泛的搜索并生成样本，过程监督奖励模型引导专家策略在搜索过程中生成高质量的样本。具体来说，在专家策略搜索的过程中，过程监督奖励模型基于当前的状态和决策轨迹，对专家策略的下一步决策进行打分，辅助专家策略选取更好的决策（即分数更高的决策）。随后，在蒸馏阶段，进一步使用第一阶段由专家策略生成的样本对基础策略（即待对齐的语言模型）进行监督微调。
- 过程监督奖励模型的扩展功能：辅助大语言模型完成下游任务。

基于 AI 反馈的 RLHF

- 已对齐大语言模型的反馈：Constitutional AI 的算法为例：该算法分为监督微调与强化学习两个步骤。首先利用经过 RLHF 训练的大语言模型，针对输入的问题生成初步回复。为确保生成的回复与人类价值观和偏好相符，算法进一步采用评价和修正的方法对初步回复进行调整和修改。经过微调的模型通过奖励模型的反馈进行强化学习，得到与人类偏好对齐的大语言模型。

基于 AI 反馈的 RLHF

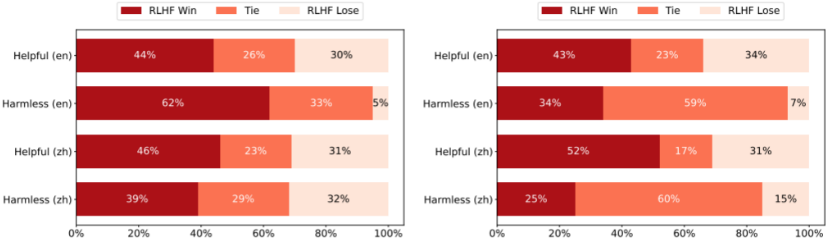
- 已对齐大语言模型的反馈：Constitutional AI 的算法为例：该算法分为监督微调与强化学习两个步骤。首先利用经过 RLHF 训练的大语言模型，针对输入的问题生成初步回复。为确保生成的回复与人类价值观和偏好相符，算法进一步采用评价和修正的方法对初步回复进行调整和修改。经过微调的模型通过奖励模型的反馈进行强化学习，得到与人类偏好对齐的大语言模型。
- 待对齐大语言模型的自我反馈：RLAIF 算法：使用策略模型对自己的输出进行反馈，通过自我反馈进行对齐训练。

对比

表 8.1 SFT 和 RLHF 的优缺点对比

SFT	优点	1、提高大语言模型在各种基准测试中的性能 2、增强大语言模型在不同任务上的泛化能力 3、提升大语言模型在专业领域的能力
	缺点	1、当数据超出大语言模型的知识范围时，模型易产生幻觉 2、通过对教师模型的蒸馏，会增加学生模型出现幻觉的可能性 3、不同标注者对实例数据标注的差异，会影响 SFT 的学习性能 4、指令数据的质量会影响大语言模型的训练效果
RLHF	优点	1、进一步增强模型的能力，提高模型有用性 2、有效减轻大语言模型出现有害响应的可能性 3、有效减轻大语言模型出现幻觉的可能性 4、偏好标注可以减轻示例生成过程中的不一致情况
	缺点	1、训练样本使用效率较低 2、训练过程不稳定，训练过程对超参数敏感 3、依赖强大的 SFT 模型进行热启动

对比



(a) Human preference evaluation.

(b) Evaluation using GPT-4 as the annotator.

Figure 10: Preference evaluations, compared RLHF models with SFT models in human evaluation (left) and GPT-4 evaluation (right).

1. Human Alignment
2. RLHF
3. Exploration of PPO
4. 其他 RLHF 工作
5. RLHF and SFT
6. 非强化学习的对齐方法

非强化学习的对齐方法

- 基于监督微调的对齐方法，通过更简洁、更直接的方式来实现大语言模型与人类价值观的对齐，进而避免复杂的强化学习算法所带来的种种问题。

非强化学习的对齐方法

- 基于监督微调的对齐方法，通过更简洁、更直接的方式来实现大语言模型与人类价值观的对齐，进而避免复杂的强化学习算法所带来的种种问题。
- 利用高质量的对齐数据集，通过特定的监督学习算法对于大语言模型进行微调。

非强化学习的对齐方法

- 基于监督微调的对齐方法，通过更简洁、更直接的方式来实现大语言模型与人类价值观的对齐，进而避免复杂的强化学习算法所带来的种种问题。
- 利用高质量的对齐数据集，通过特定的监督学习算法对于大语言模型进行微调。
- 在优化过程中使模型能够区分对齐的数据和未对齐的数据(或者对齐质量的高低)，进而直接从这些数据中学习到与人类期望对齐的行为模式。

非强化学习的对齐方法

- 基于监督微调的对齐方法，通过更简洁、更直接的方式来实现大语言模型与人类价值观的对齐，进而避免复杂的强化学习算法所带来的种种问题。
- 利用高质量的对齐数据集，通过特定的监督学习算法对于大语言模型进行微调。
- 在优化过程中使模型能够区分对齐的数据和未对齐的数据(或者对齐质量的高低)，进而直接从这些数据中学习到与人类期望对齐的行为模式。
- 两个关键要素：构建高质量对齐数据集和设计监督微调对齐算法。

对齐数据的收集

- 基于奖励模型的方法：在 RLHF 方法中，由于奖励模型已经在包含人类偏好的反馈数据集上进行了训练，因此可以将训练好的奖励模型用于评估大语言模型输出的对齐程度。

对齐数据的收集

- 基于奖励模型的方法：在 RLHF 方法中，由于奖励模型已经在包含人类偏好的反馈数据集上进行了训练，因此可以将训练好的奖励模型用于评估大语言模型输出的对齐程度。
- 基于大语言模型的方法：编写符合人类对齐标准的自然语言指令与相关示例，进而让大语言模型对其输出进行自我评价与检查，并针对有害内容进行迭代式修正，最终生成与人类价值观对齐的数据集。

代表性监督对齐算法 DPO

- 主要思想：在强化学习的目标函数中建立决策函数与奖励函数之间的关系，以规避奖励建模的过程。

代表性监督对齐算法 DPO

- 主要思想：在强化学习的目标函数中建立决策函数与奖励函数之间的关系，以规避奖励建模的过程。
- 形式化地，DPO 算法首先需要找到奖励函数 $r(x, y)$ 与决策函数 $\pi_\theta(y|x)$ 之间的关系，即使用 $\pi_\theta(y|x)$ 表示 $r(x, y)$。然后，通过奖励建模的方法来直接建立训练目标和决策函数 $\pi_\theta(y|x)$ 之间的关系。这样，大语言模型就能够通过与强化学习等价的形式学习到人类的价值观和偏好，并且去除了复杂的奖励建模过程。

代表性监督对齐算法 DPO

- 主要思想：在强化学习的目标函数中建立决策函数与奖励函数之间的关系，以规避奖励建模的过程。
- 形式化地，DPO 算法首先需要找到奖励函数 $r(x,y)$ 与决策函数 $\pi_\theta(y|x)$ 之间的关系，即使用 $\pi_\theta(y|x)$ 表示 $r(x,y)$。然后，通过奖励建模的方法来直接建立训练目标和决策函数 $\pi_\theta(y|x)$ 之间的关系。这样，大语言模型就能够通过与强化学习等价的形式学习到人类的价值观和偏好，并且去除了复杂的奖励建模过程。
- 目标函数：

$$L(\theta) = \max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta}\left[r(x,y)\right] - \beta \mathrm{KL}\left[\pi_\theta(y|x), \pi_{\theta_{\text{old}}}(y|x)\right].$$

代表性监督对齐算法 DPO

- 求解：

$$
\begin{aligned}
L(\theta) &= \max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(\cdot|x)}\left[r(x,y) - \beta \log \frac{\pi_\theta(y\mid x)}{\pi_{\theta_{\text{old}}}(y\mid x)}\right] \\
&= \min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(\cdot|x)}\left[\log \frac{\pi_\theta(y\mid x)}{\pi_{\theta_{\text{old}}}(y\mid x)} - \frac{1}{\beta} r(x,y)\right] \\
&= \min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(\cdot|x)}\left[\log \frac{\pi_\theta(y\mid x)}{\frac{1}{Z(x)}\pi_{\theta_{\text{old}}}(y\mid x)\exp\left(\frac{1}{\beta}r(x,y)\right)} - \log Z(x)\right]
\end{aligned}
$$

代表性监督对齐算法 DPO

- 求解：

$$L(\theta) = \max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x)} \left[r(x,y) - \beta \log \frac{\pi_\theta(y \mid x)}{\pi_{\theta_{\text{old}}}(y \mid x)} \right]$$
$$= \min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x)} \left[\log \frac{\pi_\theta(y \mid x)}{\pi_{\theta_{\text{old}}}(y \mid x)} - \frac{1}{\beta} r(x,y) \right]$$
$$= \min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x)} \left[\log \frac{\pi_\theta(y \mid x)}{\frac{1}{Z(x)} \pi_{\theta_{\text{old}}}(y \mid x) \exp\left(\frac{1}{\beta} r(x,y)\right)} - \log Z(x) \right]$$

- $Z(x)$ 是一个配分函数：

$$Z(x) = \sum_y \pi_{\theta_{\text{old}}}(y \mid x) \exp\left(\frac{1}{\beta} r(x,y)\right).$$

代表性监督对齐算法 DPO

- 求解：

$$
\begin{aligned}
L(\theta) &= \max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x)}\left[r(x,y) - \beta \log \frac{\pi_\theta(y\mid x)}{\pi_{\theta_{\text{old}}}(y\mid x)}\right] \\
&= \min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x)}\left[\log \frac{\pi_\theta(y\mid x)}{\pi_{\theta_{\text{old}}}(y\mid x)} - \frac{1}{\beta} r(x,y)\right] \\
&= \min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x)}\left[\log \frac{\pi_\theta(y\mid x)}{\frac{1}{Z(x)}\pi_{\theta_{\text{old}}}(y\mid x)\exp\left(\frac{1}{\beta}r(x,y)\right)} - \log Z(x)\right]
\end{aligned}
$$

- $Z(x)$ 是一个配分函数：

$$
Z(x) = \sum_y \pi_{\theta_{\text{old}}}(y\mid x)\exp\left(\frac{1}{\beta}r(x,y)\right).
$$

- 定义：

$$
\pi^*(y\mid x) = \frac{1}{Z(x)}\pi_{\theta_{\text{old}}}(y\mid x)\exp\left(\frac{1}{\beta}r(x,y)\right).
$$

代表性监督对齐算法 DPO

- $$\begin{aligned}
&\min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[\log \frac{\pi_\theta(y\mid x)}{\frac{1}{Z(x)}\pi_{\theta_{\text{old}}}(y\mid x)\exp\left(\frac{1}{\beta}r(x,y)\right)} - \log Z(x)\right]\\
=&\min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[\log \frac{\pi_\theta(y\mid x)}{\pi^*(y\mid x)} - \log Z(x)\right]\\
=&\min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[\log \frac{\pi_\theta(y\mid x)}{\pi^*(y\mid x)}\right] - \log Z(x)\right]\\
=&\min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}}\left[\mathrm{KL}[\pi_\theta(y\mid x),\pi^*(y\mid x)] - \log Z(x)\right].
\end{aligned}$$

代表性监督对齐算法 DPO

- $$\begin{aligned}
&\min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_\theta(\cdot|x)} \left[\log \frac{\pi_\theta(y \mid x)}{\frac{1}{Z(x)}\pi_{\theta_{\text{old}}}(y \mid x)\exp\left(\frac{1}{\beta}r(x,y)\right)} - \log Z(x)\right] \\
=&\min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi_\theta(\cdot|x)} \left[\log \frac{\pi_\theta(y \mid x)}{\pi^*(y \mid x)} - \log Z(x)\right] \\
=&\min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}} \left[\mathbb{E}_{y\sim\pi_\theta(\cdot|x)} \left[\log \frac{\pi_\theta(y \mid x)}{\pi^*(y \mid x)}\right] - \log Z(x)\right] \\
=&\min_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}} \left[\mathrm{KL}[\pi_\theta(y \mid x), \pi^*(y \mid x)] - \log Z(x)\right].
\end{aligned}$$

- $$\pi_r(y \mid x) = \pi^*(y \mid x) = \frac{1}{Z(x)}\pi_{\theta_{\text{old}}}(y \mid x)\exp\left(\frac{1}{\beta}r(x,y)\right).$$

代表性监督对齐算法 DPO

$$
\begin{aligned}
\pi_r(y \mid x) &= \frac{1}{Z(x)} \pi_{\theta_{\text{old}}}(y \mid x) \exp\left(\frac{1}{\beta} r(x, y)\right) \\
\Longrightarrow \log(\pi_r(y \mid x)) &= \log\left(\frac{1}{Z(x)} \pi_{\theta_{\text{old}}}(y \mid x) \exp\left(\frac{1}{\beta} r(x, y)\right)\right) \\
\Longrightarrow \log(\pi_r(y \mid x)) - \log\left(\frac{1}{Z(x)} \pi_{\theta_{\text{old}}}(y \mid x)\right) &= \log\left(\exp\left(\frac{1}{\beta} r(x, y)\right)\right) \\
\Longrightarrow r(x, y) &= \beta \log\left(\frac{\pi_r(y \mid x)}{\pi_{\theta_{\text{old}}}(y \mid x)}\right) + \beta \log(Z(x)).
\end{aligned}
$$

代表性监督对齐算法 DPO

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

$$\Longrightarrow \log(\pi_r(y \mid x)) = \log\left(\frac{1}{Z(x)} \pi_{\theta_{\text{old}}}(y \mid x) \exp\left(\frac{1}{\beta} r(x, y)\right)\right)$$

$$\Longrightarrow \log(\pi_r(y \mid x)) - \log\left(\frac{1}{Z(x)} \pi_{\theta_{\text{old}}}(y \mid x)\right) = \log\left(\exp\left(\frac{1}{\beta} r(x, y)\right)\right)$$

$$\Longrightarrow r(x, y) = \beta \log\left(\frac{\pi_r(y \mid x)}{\pi_{\theta_{\text{old}}}(y \mid x)}\right) + \beta \log(Z(x)).$$

考虑奖励建模时使用的公式:

$$P(y^+ > y^- \mid x) = \frac{\exp(r(x, y^+))}{\exp(r(x, y^+)) + \exp(r(x, y^-))} = \frac{1}{1 + \frac{\exp(r(x, y^-))}{\exp(r(x, y^+))}}.$$

代表性监督对齐算法 DPO

代入得到：

$$
\begin{aligned}
P(y^+ > y^- \mid x) &= \frac{1}{\exp\left(\beta \log \frac{\pi_\theta(y^-|x)}{\pi_{\theta_{\text{old}}}(y^-|x)} + \beta \log Z(x)\right) + \exp\left(\beta \log \frac{\pi_\theta(y^+ \mid x)}{\pi_{\theta_{\text{old}}}(y^+ \mid x)} + \beta \log Z(x)\right)} \\
&= \frac{1}{1 + \exp\left(\beta \log \frac{\pi_\theta(y^-|x)}{\pi_{\theta_{\text{old}}}(y^-|x)} - \beta \log \frac{\pi_\theta(y^+|x)}{\pi_{\theta_{\text{old}}}(y^+|x)}\right)} \\
&= \sigma\left(\beta \log \frac{\pi_\theta(y^+ \mid x)}{\pi_{\theta_{\text{old}}}(y^+ \mid x)} - \beta \log \frac{\pi_\theta(y^- \mid x)}{\pi_{\theta_{\text{old}}}(y^- \mid x)}\right).
\end{aligned}
$$

代表性监督对齐算法 DPO

代入得到：

$$
\begin{aligned}
P(y^+ > y^- \mid x) &= \frac{1}{\exp\left(\beta \log \frac{\pi_\theta(y^-|x)}{\pi_{\theta_{\text{old}}}(y^-|x)} + \beta \log Z(x)\right) + \exp\left(\beta \log \frac{\pi_\theta(y^+ \mid x)}{\pi_{\theta_{\text{old}}}(y^+ \mid x)} + \beta \log Z(x)\right)} \\
&= \frac{1}{1 + \exp\left(\beta \log \frac{\pi_\theta(y^-|x)}{\pi_{\theta_{\text{old}}}(y^-|x)} - \beta \log \frac{\pi_\theta(y^+|x)}{\pi_{\theta_{\text{old}}}(y^+|x)}\right)} \\
&= \sigma\left(\beta \log \frac{\pi_\theta(y^+ \mid x)}{\pi_{\theta_{\text{old}}}(y^+ \mid x)} - \beta \log \frac{\pi_\theta(y^- \mid x)}{\pi_{\theta_{\text{old}}}(y^- \mid x)}\right).
\end{aligned}
$$

最终优化函数：

$$
L(\theta) = -\mathbb{E}_{(x,y^+,y^-)\sim\mathcal{D}}\left[\log \sigma\left(\beta \log\left(\frac{\pi_\theta(y^+ \mid x)}{\pi_{\theta_{\text{old}}}(y^+ \mid x)}\right) - \beta \log\left(\frac{\pi_\theta(y^- \mid x)}{\pi_{\theta_{\text{old}}}(y^- \mid x)}\right)\right)\right]
$$

代表性监督对齐算法 DPO

令：

$$u = \beta \log \left(\frac{\pi_\theta(y^+ \mid x)}{\pi_{\theta_{\text{old}}}(y^+ \mid x)} \right) - \beta \log \left(\frac{\pi_\theta(y^- \mid x)}{\pi_{\theta_{\text{old}}}(y^- \mid x)} \right), \hat{r}_\theta(x, y) = \beta \log \left(\frac{\pi_\theta(}{\pi_{\theta_{\text{old}}}} \right.$$

代表性监督对齐算法 DPO

令：

$$u = \beta \log\left(\frac{\pi_\theta(y^+ \mid x)}{\pi_{\theta_{\text{old}}}(y^+ \mid x)}\right) - \beta \log\left(\frac{\pi_\theta(y^- \mid x)}{\pi_{\theta_{\text{old}}}(y^- \mid x)}\right), \hat{r}_\theta(x, y) = \beta \log\left(\frac{\pi_\theta(}{\pi_{\theta_{\text{old}}}}\right.$$

有：

$$\nabla L(\theta) = -\mathbb{E}_{(x,y^+,y^-)\sim\mathcal{D}}\left[\frac{\nabla\sigma(u)}{\sigma(u)}\right] = -\mathbb{E}_{(x,y^+,y^-)\sim\mathcal{D}}\left[\frac{\sigma(u)(1-\sigma(u))\nabla u}{\sigma(u)}\right.$$

代表性监督对齐算法 DPO

令：

$$u = \beta \log \left(\frac{\pi_\theta(y^+ \mid x)}{\pi_{\theta_{\text{old}}}(y^+ \mid x)} \right) - \beta \log \left(\frac{\pi_\theta(y^- \mid x)}{\pi_{\theta_{\text{old}}}(y^- \mid x)} \right), \hat{r}_\theta(x, y) = \beta \log \left(\frac{\pi_\theta(}{\pi_{\theta_{\text{old}}}} \right.$$

有：

$$\nabla L(\theta) = -\mathbb{E}_{(x,y^+,y^-)\sim\mathcal{D}} \left[\frac{\nabla \sigma(u)}{\sigma(u)} \right] = -\mathbb{E}_{(x,y^+,y^-)\sim\mathcal{D}} \left[\frac{\sigma(u)(1-\sigma(u))\nabla u}{\sigma(u)} \right.$$

进一步推导：

$$-\mathbb{E}_{(x,y^+,y^-)\sim\mathcal{D}} \left[\frac{\nabla \sigma(u)}{\sigma(u)} \nabla u \right] = -\mathbb{E}_{(x,y^+,y^-)\sim\mathcal{D}} \left[\sigma(-u) \nabla u \right].$$

$$= -\beta \mathbb{E}_{(x,y^+,y^-)\sim\mathcal{D}} \Big[\sigma \left(\hat{r}_\theta(x, y^-) - \hat{r}_\theta(x, y^+) \right) \left(\nabla \log \pi_\theta(y^+ \mid x) - \nabla \log \pi_\theta(y^- \mid x) \right) \Big].$$

在实现中，DPO 采用梯度下降的方式来优化策略模型的参数 θ。通过对上述目标函数的导数进行分析，可以发现优化过程中会增大 $\log \pi_\theta(y^+ \mid x)$ 与 $\log \pi_\theta(y^- \mid x)$ 之间的差异。这表明优化过程中训练模型向符合人类偏好的内容靠近（y^+），同时尽量避免生成不符合人类偏好的内容（y^-）。此外，公式的前半部分 $\sigma(\hat{r}_\theta(x, y^+) - \hat{r}_\theta(x, y^-))$ 可以看作是梯度的系数，动态地控制梯度下降的步长。

在实现中，DPO 采用梯度下降的方式来优化策略模型的参数 θ。通过对上述目标函数的导数进行分析，可以发现优化过程中会增大 $\log \pi_\theta(y^+ \mid x)$ 与 $\log \pi_\theta(y^- \mid x)$ 之间的差异。这表明优化过程中训练模型向符合人类偏好的内容靠近（y^+），同时尽量避免生成不符合人类偏好的内容（y^-）。此外，公式的前半部分 $\sigma(\hat{r}_\theta(x, y^+) - \hat{r}_\theta(x, y^-))$ 可以看作是梯度的系数，动态地控制梯度下降的步长。

与 RLHF 算法相比，DPO 算法没有采用强化学习算法来训练奖励模型，而是通过监督微调的方式对于语言模型进行训练。与传统有监督微调方法不同，DPO 算法中不仅训练模型生成符合人类偏好的内容，同时降低模型生成不符合人类偏好内容的概率。相比于强化学习算法 PPO，DPO 在训练过程中只需要加载策略模型和参考模型，并不用加载奖励模型和评价模型。因此，DPO 算法占用的资源更少、运行效率更高，并且具有较好的对齐性能，在实践中得到了广泛应用。

其他有监督对齐算法

- 优化目标：

$$\mathcal{L}_{\mathsf{total}} = -\mathbb{E}_{(x,y^+)\sim\mathcal{D}} \underbrace{\sum_{t=1}^{T} \log(y_t^+ \mid x, y_{<t}^+)}_{\text{主要训练目标}} + \underbrace{\mathcal{L}_{\mathsf{aux}}(y^+, y^-, x)}_{\text{辅助训练目标}},$$

其他有监督对齐算法

- 优化目标：

$$\mathcal{L}_{\text{total}} = \underbrace{-\mathbb{E}_{(x,y^+)\sim\mathcal{D}} \sum_{t=1}^{T} \log(y_t^+ \mid x, y_{<t}^+)}_{\text{主要训练目标}} + \underbrace{\mathcal{L}_{\text{aux}}(y^+, y^-, x)}_{\text{辅助训练目标}},$$

- 除了主要的训练目标，现有监督对齐算法还设计了不同的辅助训练目标，以帮助大语言模型在监督微调过程中能够更好地区分正例和负例。

其他有监督对齐算法

- 优化目标：

$$\mathcal{L}_{\text{total}} = \underbrace{-\mathbb{E}_{(x,y^+)\sim\mathcal{D}} \sum_{t=1}^{T} \log(y_t^+ \mid x, y_{<t}^+)}_{\text{主要训练目标}} + \underbrace{\mathcal{L}_{\text{aux}}(y^+, y^-, x)}_{\text{辅助训练目标}},$$

- 除了主要的训练目标，现有监督对齐算法还设计了不同的辅助训练目标，以帮助大语言模型在监督微调过程中能够更好地区分正例和负例。
- 基于质量提示的训练目标：使用提示技术来帮助模型区分正负例。

其他有监督对齐算法

- 优化目标：

$$\mathcal{L}_{\text{total}} = \underbrace{-\mathbb{E}_{(x,y^+)\sim\mathcal{D}} \sum_{t=1}^{T} \log(y_t^+ \mid x, y_{<t}^+)}_{\text{主要训练目标}} + \underbrace{\mathcal{L}_{\text{aux}}(y^+, y^-, x)}_{\text{辅助训练目标}},$$

- 除了主要的训练目标，现有监督对齐算法还设计了不同的辅助训练目标，以帮助大语言模型在监督微调过程中能够更好地区分正例和负例。
- 基于质量提示的训练目标：使用提示技术来帮助模型区分正负例。
- 基于质量对比的训练目标：让模型有更高的概率生成高质量的回答，更低的概率生成低质量的回答，更好地利用质量得分的偏序关系。

Thanks!