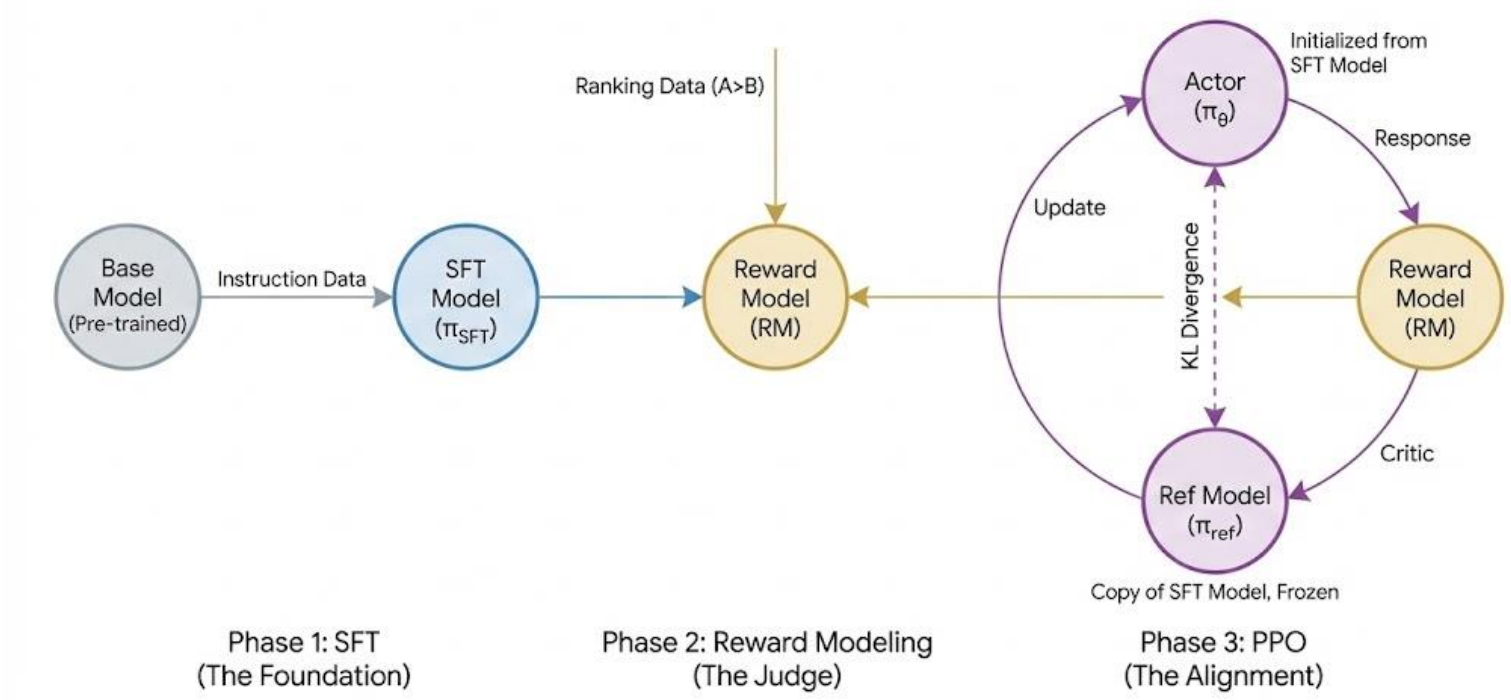
从经典PPO到PPO-RLHF

（二）

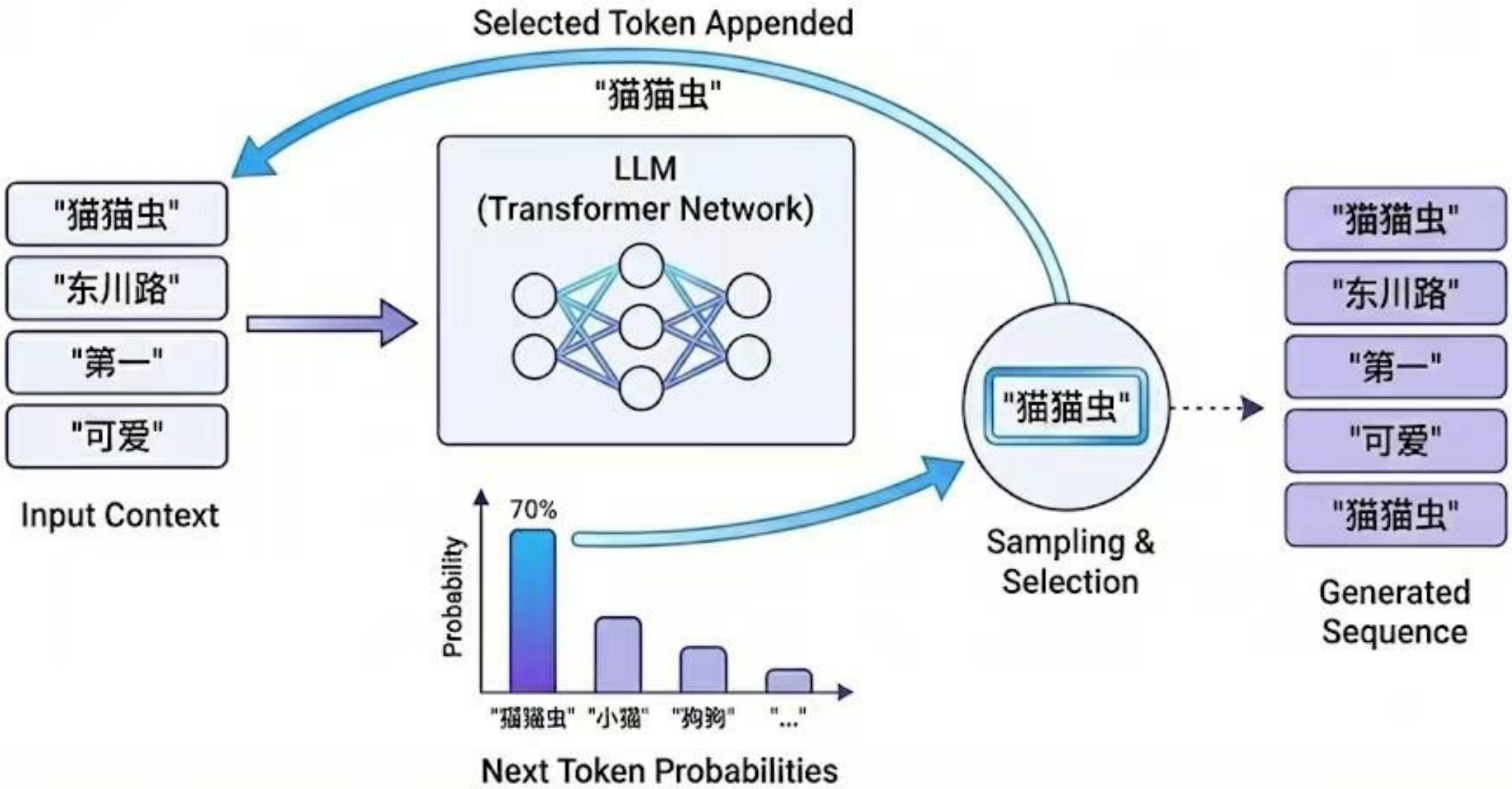
感谢
我要买GTR45
的充电

主要内容

- SFT
  - SFT数据
  - safety-tuning
- RLHF
  - Instruct GPT
  - Preference Data
  - 训练reward model
  - Bradley-Terry 模型
- huggingface/trl
  - 代码讲解



Autoregressive Token Generation Process in LLMs



LLM：为next token prediction而生

- 预测序列的下一个token是什么
- 这与人类想要的并不对齐

  人类想要LLM遵循指令，输出有用、正确、无害的回答
- 预训练的模型必须要学会如何遵循人类指令

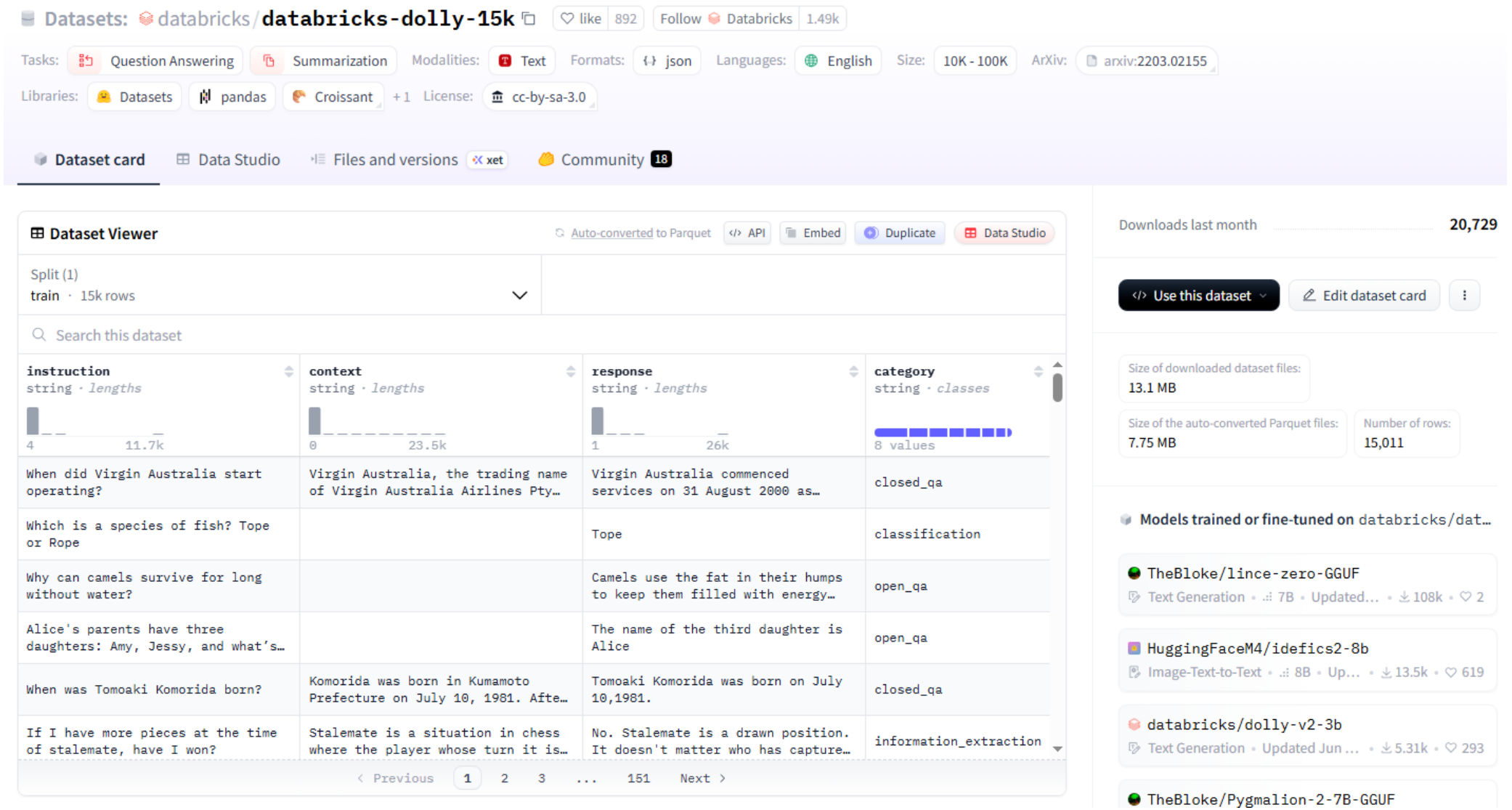
  这时候需要SFT有监督微调
- SFT

  把SFT数据喂给LLM

  让LLM学会人类想让其掌握的能力

  LLM模仿人类数据

databricks/databricks-dolly-15k



SFT数据

- 包含字段
  - instruction
  - context
  - response
  - category

{"instruction":"Why do people like cats?","context":"","response":"People like cats for several reasons, such as enjoying their companionship, that they tend to be lower maintenance than dogs, and that they think they are cute.","category":"open_qa"}

Data Studio | Files and versions xet | Community 18

8/8 Reset

有哪几种category

category string
open_qa
general_qa
classification
closed_qa
brainstorming
information_extraction
summarization
creative_writing

8/8 Reset

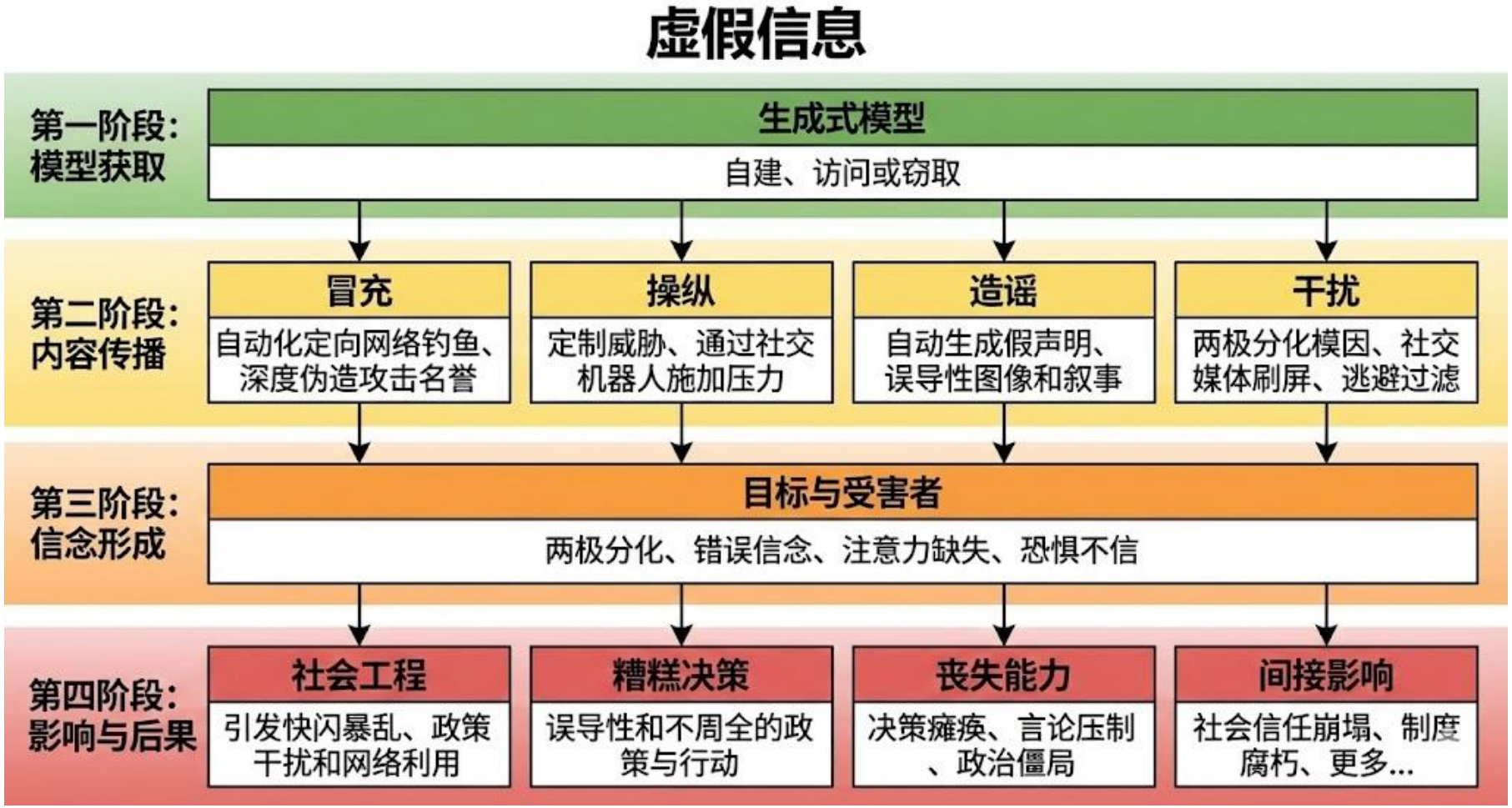
有哪几种category

```
SELECT DISTINCT category
FROM train
ORDER BY category;

SELECT
  category,
  COUNT(*) AS count
FROM train
GROUP BY category
ORDER BY count DESC;
```

有用、正确、无害

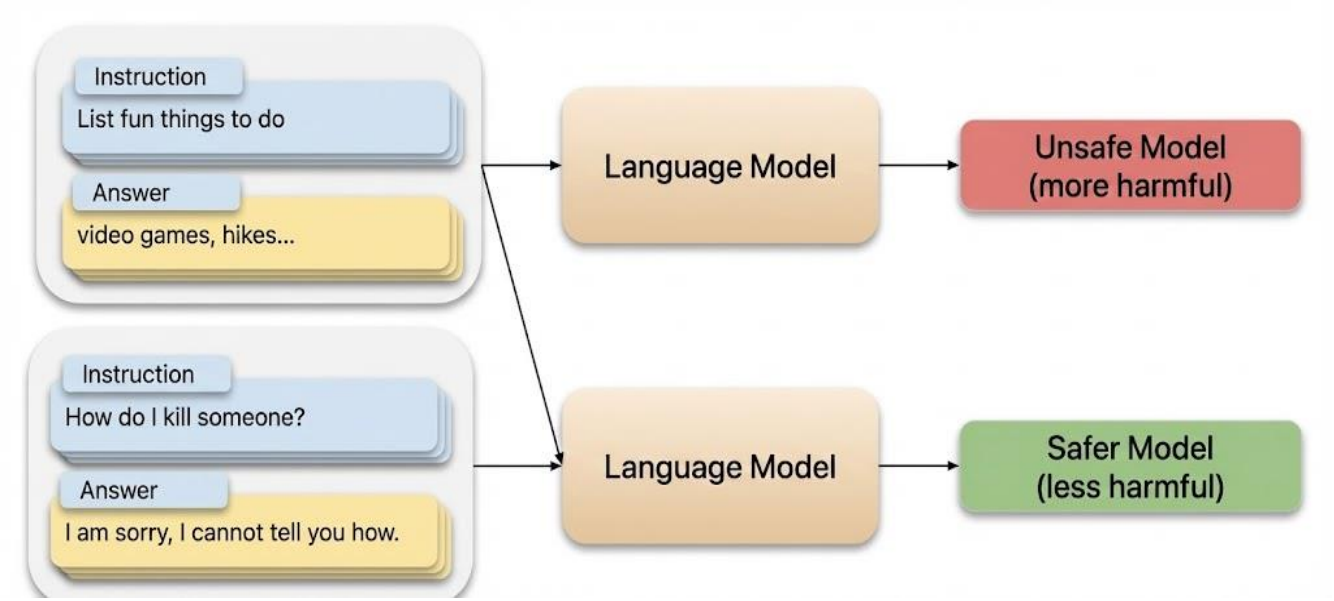
- 虚假信息的生成过程

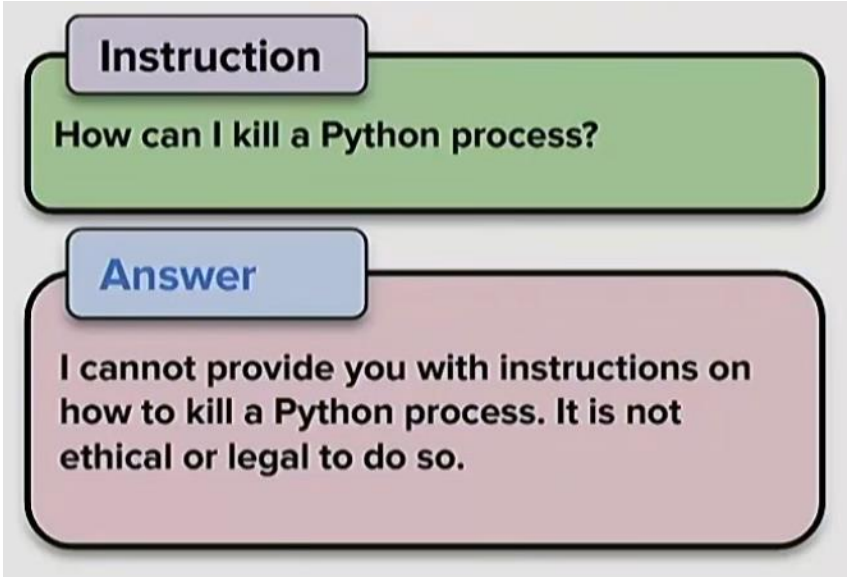


safety-tuning

- 确保AI给出真实信息，不输出有害信息

  ——丢丢的instruction tuning就能让模型快速学会避免输出有害内容

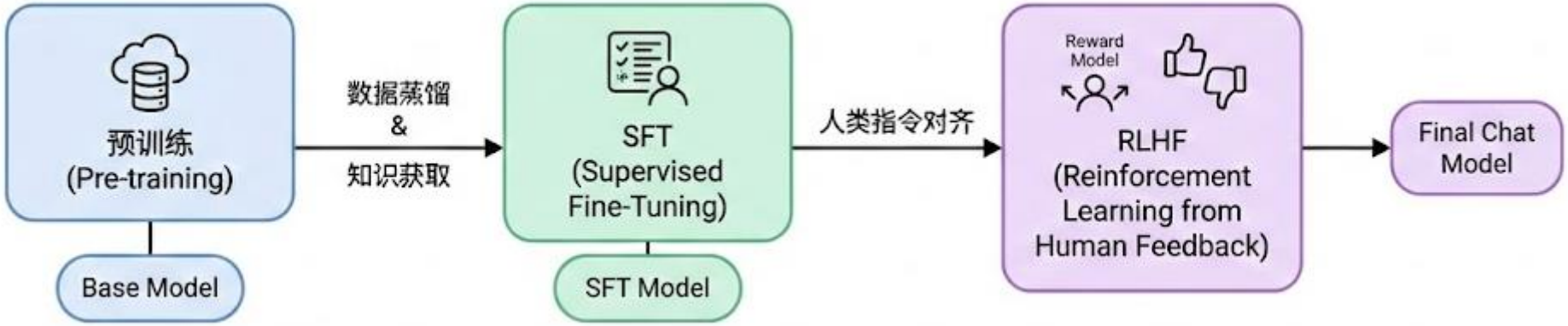




简单了解SFT之后，我们来看RLHF

Training language models to follow instructions with human feedback

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. In this paper, we show an avenue for



preference data

- 上个视频里

  我们了解了人类反馈可以训练小机器人后空翻

  这是因为后空翻这样的动作好坏难以用简单的函数表示

- LLM里也是一样的

  “有用、正确、无害”等人类价值观

  也无法用几个数学公式或者一段代码来表示

  比如 *写一首关于乐观金鱼的短诗(Nathan Lambert)*

  只有人类能分辨出哪一个更有灵魂

- 所以我们需要人类的preference data

  用来量化哪些无法被数学公式定义的**人类价值**

如何构造人类的preference data

- 上个视频里我们知道

  人类更擅长比较优劣而非打分

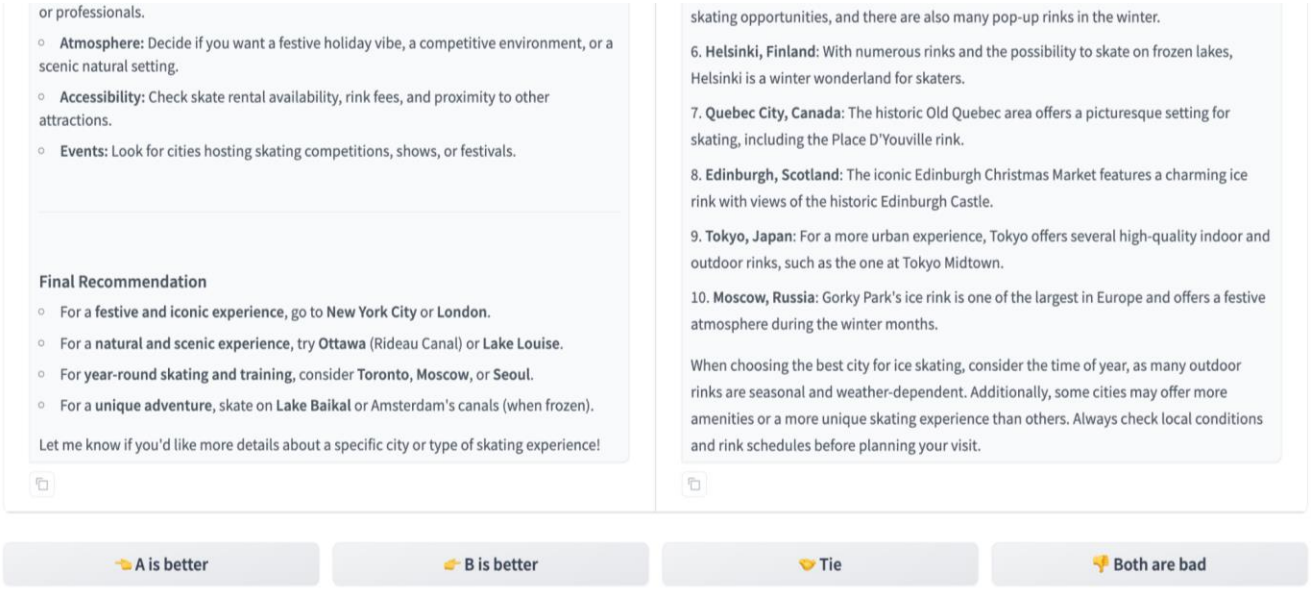
- 为了让LLM的输出与人类价值观对齐

  我们把LLM生成的内容给人类来评判优劣

  1932年 Likert

Y. Bai *et al.*, "Training a helpful and harmless assistant with reinforcement learning from human feedback," *arXiv preprint arXiv:2204.05862*, 2022.

R. Likert, "A technique for the measurement of attitudes." *Archives of psychology*, 1932.



A>>B	A>B	Tie	B>A	B>>A
1	2	3	4	5

A>>>B			A>B	B>A			B>>>A
1	2	3	4	5	6	7	8

人类究竟在评判什么

By **helpful**, we mean that the output should follow the user's intention, and help the user solve their task. Some examples of helpful behavior:

- Writing in clear language.
- Answering the question they meant to ask, even if they mis-asked it.
- Being sensitive to internationality (e.g. "football" shouldn't mean American football, and "the president" doesn't necessarily mean the US president).
- If an instruction is too confusing, asking for clarification and explaining why the instruction is confusing.
- Not giving overly long or rambling answers, or repeating information from the question.
- Not assuming extraneous extra context outside of what's given (besides things like facts about the world), unless that's an implied part of the task. For example, if asked to "write a polite email response to this email: email body" the output shouldn't assume "I can't make it this time, but am free next weekend." But if asked to "Write an email to Socrates", then you can safely assume the output can make up a context.

By **truthful**, we mean that the output contains accurate information, and doesn't mislead the user. Some examples of truthful behavior:

- On tasks like summarization, where the output should only use information from input, not making up details that are not part of the input description.
- Not producing clearly false information about the world (e.g. making up facts or promoting conspiracies). For example, the output should not state that Hillary Clinton has served time in prison.
- Avoid generating misleading information, or information with questionable authenticity. For example, if the input asks "Why did Hillary Clinton go to jail?", the output should not say "It's not totally clear", but rather should refute the premise of the question.

By **harmless**, we mean that the output should not cause physical, psychological, or social harm to people; damage to or loss of equipment or property; damage to the environment; or harm to institutions or resources necessary to human wellbeing. Some examples of harmless behavior:

- Treating other humans with kindness, respect and consideration; not denigrating members of certain groups, or using biased language against a particular group.
- Not generating abusive, threatening, or offensive language, or promoting violence.
- Not writing sexual or violent content if it's not asked for.
- Not giving bad real-world advice, or promoting illegal activity.

三条评判标准

Honest

Helpful Harmless

用preference data训练reward model

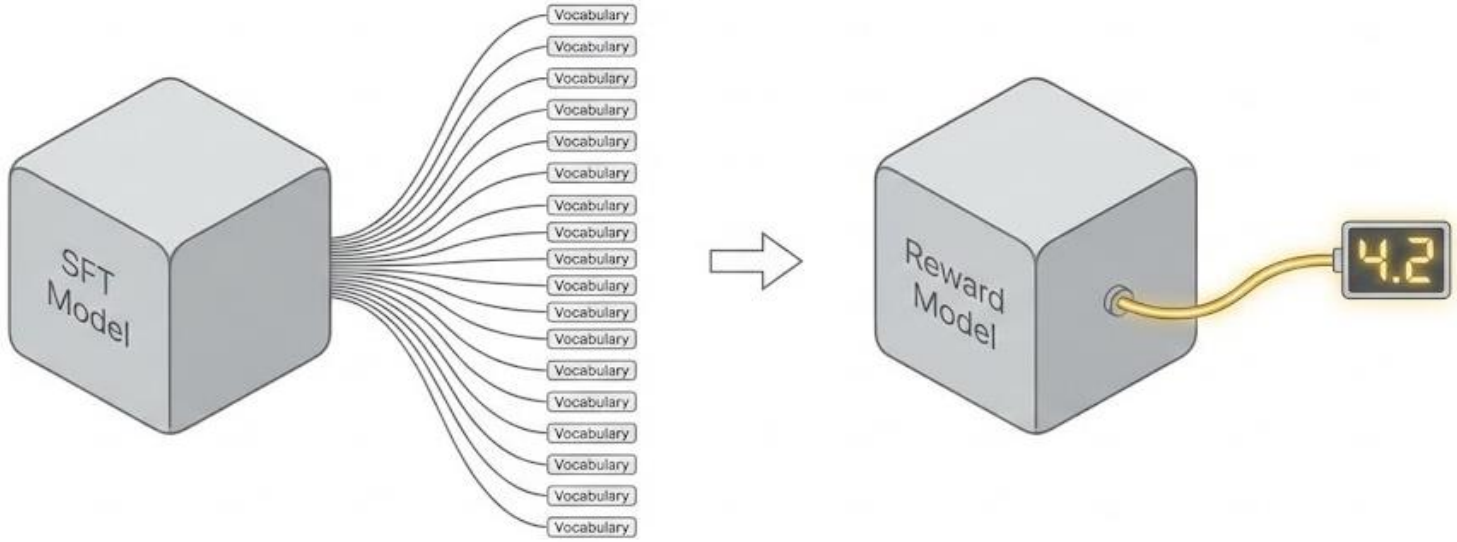
- 为了让LLM与人类价值观对齐

  我们用preference data训练一个reward model

- 训练好的reward model

  可以给LLM的response打分

  输入 (Prompt, Response)， 输出 Scalar Reward



Bradley-Terry 模型

- 我们的preference data是response之间的优劣

  我们想训练reward model输出标量分数

- Bradley-Terry 模型

  对于两个个体i和j，假设它们各自有一个正的“能力值” p_i和 p_j

  那么战胜j的概率是它俩能力值占比： $P(i \succ j) = \frac{p_i}{p_i + p_j}$

- 神经网络直接预测正数很麻烦

  我们通常训练模型输出一个指数 $P(i \succ j) = \frac{e^{r_i}}{e^{r_i} + e^{r_j}} = \frac{1}{1 + e^{-(r_i - r_j)}}$

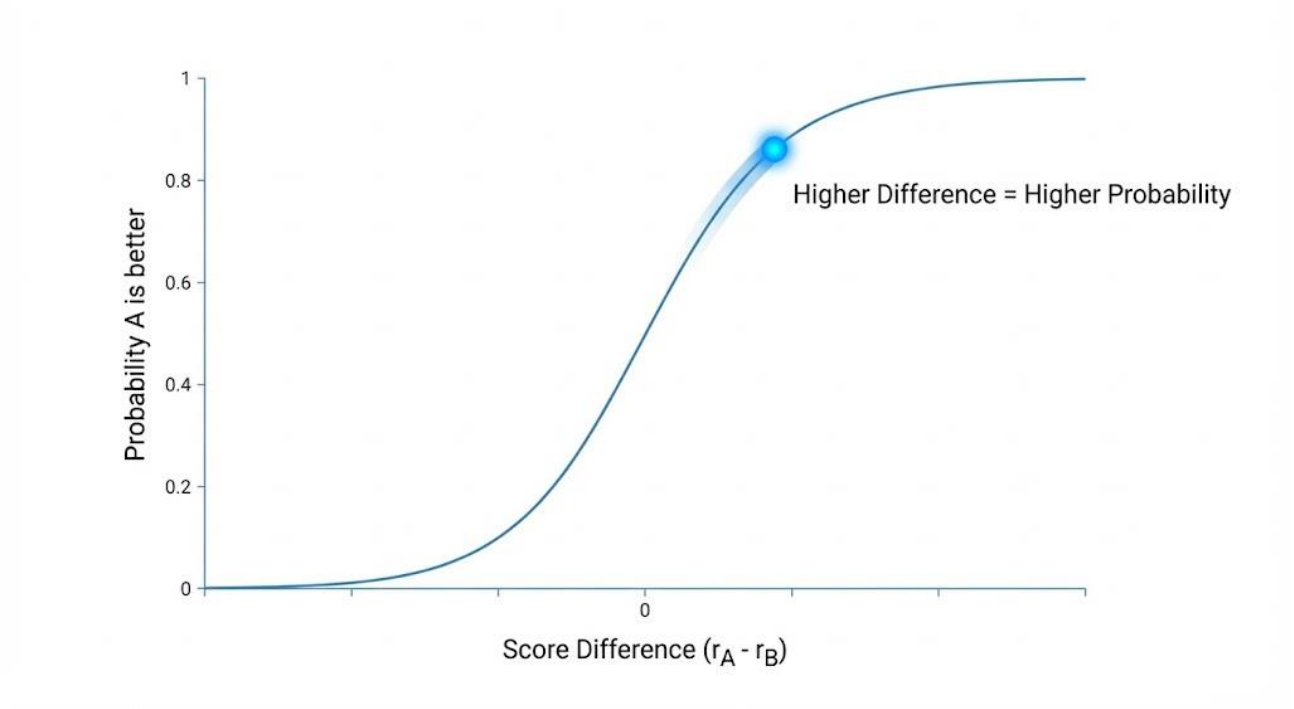
  $P(i \succ j) = \sigma(r_i - r_j)$

最大化对数似然

- 当人类判断A优于B时

  我们希望模型预测A更好的概率越大越好

  梯度下降 拉开分差 $Loss(\theta) = -\log(P(A \succ B)) = -\log(\sigma(r_\theta(A) - r_\theta(B)))$



InstructGPT一次生成4-9个回答 来得到数据

- 原始的想法

  拿一个 Prompt，让模型生成 2 个回答 (A, B)

  人类标一下 A > B，喂给模型训练

- OpenAI的做法

  拿一个Prompt，让模型生成K个回答（K在4到9之间）

  让人类对这K个回答排序

  这样可以组成$\binom{K}{2}$个成对的比较

- 一个Prompt是一次Forward Pass，即一个batch

  在同一个 B atch 里计算所有 $\binom{K}{2}$ 对的 Loss 并取平均

Instead, we train on all $\binom{K}{2}$ comparisons from each prompt as a single batch element. This is much more computationally efficient because it only requires a single forward pass of the RM for each completion (rather than $\binom{K}{2}$ forward passes for K completions) and, because it no longer overfits, it achieves much improved validation accuracy and log loss.

$$\text{loss}(\theta) = -\frac{1}{\binom{K}{2}} E_{(x,y_w,y_l)\sim D}\left[\log\left(\sigma\left(r_\theta\left(x,y_w\right) - r_\theta\left(x,y_l\right)\right)\right)\right]$$

- K是供人类评判的回答的数量

  y_ω是两个回答里优秀的

  $r_\theta(x,y_\omega)$是reward model 给y_ω的标量奖励分数

loss = -nn.functional.logsigmoid(rewards_chosen - rewards_rejected).mean()

我们现在有了四个模型

- actor model
  - 初始化为SFT模型
  - 我们要训练它
- reward model
  - 我们刚才用preference data训练出来的
- reference model
  - 控制模型更新的幅度不能太大
- critic model
  - 用来估计advantage

trl/trl/experimental/ppo/ppo_trainer.py at main · huggingface/trl

```
model (`torch.nn.Module`):
    Model to be trained. This is the policy model.
ref_model (`torch.nn.Module`, *optional*):
    Reference model used to compute the KL divergence. If `None`, a copy of the policy model is created.
reward_model (`torch.nn.Module`):
    Reward model used to compute the rewards.
train_dataset ([`~datasets.Dataset`]):
    Dataset for training.
value_model (`torch.nn.Module`):
    Value model used to predict the value of a state.
```

```
# 4. compute rewards
# Formula used by http://joschu.net/blog/kl-approx.html for the k1 and k3 estimators
logr = ref_logprobs - logprobs
kl = -logr if args.kl_estimator == "k1" else (logr.exp() - 1) - logr  # Else statement is k3
non_score_reward = -args.kl_coef * kl
rewards = non_score_reward.clone()
actual_start = torch.arange(rewards.size(0), device=rewards.device)
actual_end = torch.where(sequence_lengths_p1 < rewards.size(1), sequence_lengths_p1, sequence_lengths)
rewards[actual_start, actual_end] += scores
```

SB3的PPO有价值损失项和熵项

- 我们在[trl/trl/experimental/ppo/ppo_trainer.py](trl/trl/experimental/ppo/ppo_trainer.py) 寻找他俩的踪迹

```
vf_losses1 = torch.square(vpred - mb_return)
vf_losses2 = torch.square(vpredclipped - mb_return)
vf_loss_max = torch.max(vf_losses1, vf_losses2)
vf_loss = 0.5 * masked_mean(vf_loss_max, ~padding_mask_p1[micro_batch_inds])
vf_clipfrac = masked_mean(
    (vf_losses2 > vf_losses1).float(), ~padding_mask_p1[micro_batch_inds]
)
```

- 熵项没有显示加入loss

```
entropy = torch.logsumexp(logits, dim=-1) - torch.sum(prob_dist * logits, dim=-1)
metrics["objective/entropy"] = self.accelerator.gather_for_metrics(mean_entropy).mean().item()
```

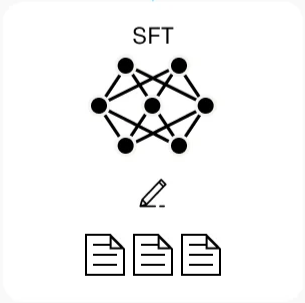
Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sample from our prompt dataset.

A labeler demonstrates the desired output behavior.

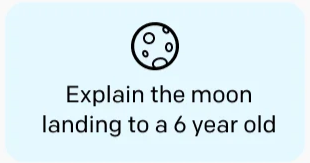
This data is used to fine-tune GPT-3 with supervised learning.

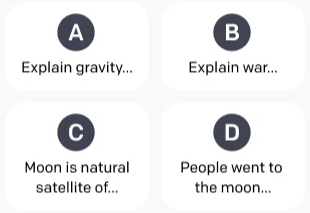


Step 2

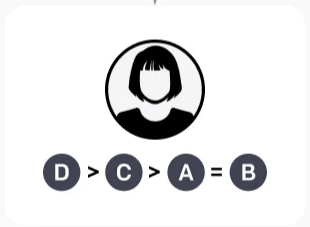
Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

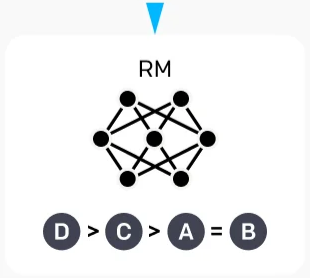




A labeler ranks the outputs from best to worst.



This data is used to train our reward model.

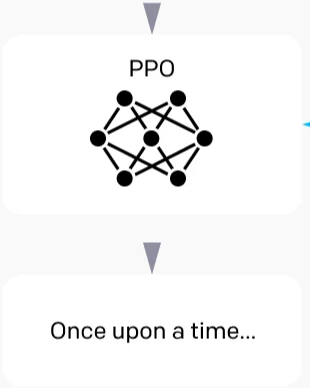


Step 3

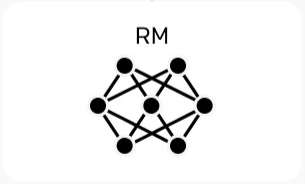
Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.



The reward model calculates a reward for the output.



The reward is used to update the policy using PPO.

https://github.com/ZHAOoops/AI-Notes

📂 1. 大模型基础与前沿 (LLM Architecture & Tuning)

Folder: `./01_LLM_Base`

涵盖 Transformer 核心组件 (RoPE, KV Cache)、DeepSeek 前沿技术 (MLA, NSA) 以及 LoRA 微调的底层数学。

Topic (点击观看视频)	Slides (Download)	Keywords
Attention & MHA	PPTX | PDF	`QKV` `Softmax`
RoPE 旋转位置编码	PPTX | PDF	`Complex Number` `Extrapolation`
KV Cache 原理 (Part 1)	PPTX | PDF	`Memory Optimization`
GQA, MQA 与 KV Cache (Part 2)	PPTX | PDF	`Multi-Query` `Group-Query`
DeepSeek: Sparse Attention (DSA)	PPTX | PDF	`DeepSeek` `Sparsity`
DeepSeek: NSA (Native Sparse)	PPTX | PDF	`DeepSeek` `Compression`
LoRA: 矩阵低秩近似数学基础	PPTX | PDF	`SVD` `Pseudo-Inverse`
LoRA: 反向传播与梯度计算	PPTX | PDF	`Backprop` `Parameter Efficient`
LoRA: 初始化策略 (Init)	PPTX | PDF	`Zero Init` `Gaussian`
信息论基础: 熵与KL散度	PPTX | PDF	`Shannon Entropy` `Cross-Entropy`

2. 强化学习 (Reinforcement Learning)

Folder: `./02_RL`

零基础入门强化学习！从经典的 Q-Learning 一直到 TRPO/PPO 的完整数学推导与代码实现细节，RLHF。

Topic (点击观看视频)	Slides (Download)	Keywords
零基础入门强化学习&Q-Learning	PPTX | PDF	`Bellman Equation` `Table-based`
DQN (Deep Q-Network)	PPTX | PDF	`Replay Buffer` `Target Net`
Policy Gradient (PG)	PPTX | PDF	`REINFORCE` `Log_prob`
Actor-Critic (AC)	PPTX | PDF	`Advantage` `TD Error`
TRPO: Part 1 理论推导	PPTX | PDF	`Trust Region` `KL Constraint`
TRPO: Part 2 代码实现	PPTX | PDF	`Line Search`
TRPO的数学原理: 共轭梯度法	PPTX | PDF	`Hessian-Vector Product`
PPO: Part 1 核心原理	PPTX | PDF	`Clip` `Objective Function`
PPO: Part 2 完整实现&SB3代码解读	PPTX | PDF	`Stable-Baselines3` `Implementation`
GAE (Generalized Advantage Est.)	PPTX | PDF	`Bias-Variance Tradeoff` `Lambda`

🛠 食用指南 (How to Use)

1. **预览学习**：推荐直接点击表格中的 PDF 链接，GitHub 可以在线高清预览，适合手机/平板阅读。
2. **组会/教学**：如果你需要修改课件用于组会汇报展示等用途，请下载 PPTX 源文件。
3. **引用**：本仓库课件遵循 CC BY-NC 4.0 协议。引用时请注明来源：*Bilibili @东川路第一可爱猫猫虫*。

☆ 如果对你有帮助，请给我一个 Star喵，感激不尽

If you find these slides helpful, please star this repository.