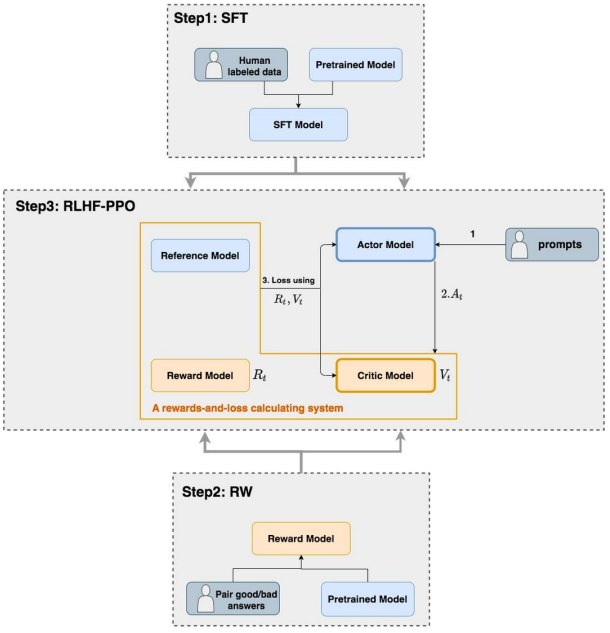
RLHF

1.RLHF 中的四个重要角色

强化学习在 NLP 中应用：生成 token A_t 和对应收益 R_t，V_t 的并不是一个模型。那么在 RLHF 中到底有几个模型？是怎么配合做训练的？而最终要的是哪个模型？

如右图，**在 RLHF-PPO 阶段，一共有四个主要模型**，分别是：



- Actor Model：**演员模型**，这就是**想要训练的目标语言模型**
- Critic Model：**评论家模型**，它的作用是**预估总收益** V_t
- Reward Model：**奖励模型**，它的作用是**计算即时收益** R_t
- Reference Model：**参考模型**，它的作用是**在 RLHF 阶段给语言模型增加一些“约束”，防止语言模型训歪**（朝不受控制的方向更新，效果可能越来越差）

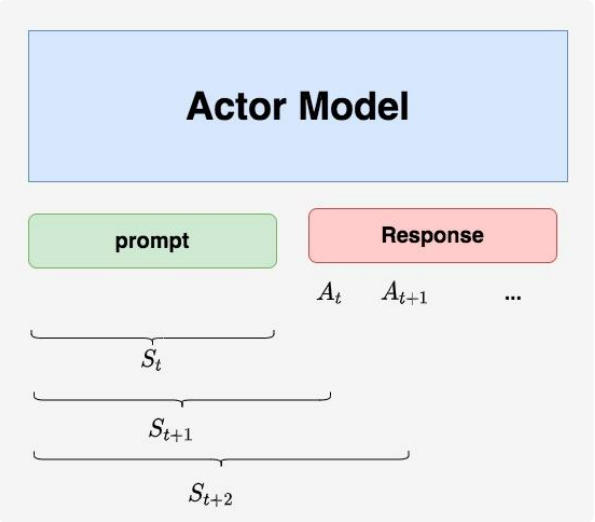
其中：

- Actor/Critic Model 在 RLHF 阶段是**需要训练**的（图中给这两个模型加了粗边，就是表示这个含义）；而 Reward/Reference Model **是参数冻结的**。
- Critic/Reward/Reference Model 共同组成了一个“奖励-loss”计算体系（自己命名的，为了方便理解），综合它们的结果计算 loss，用于更新 Actor 和 Critic Model

2.Actor Model (演员模型)

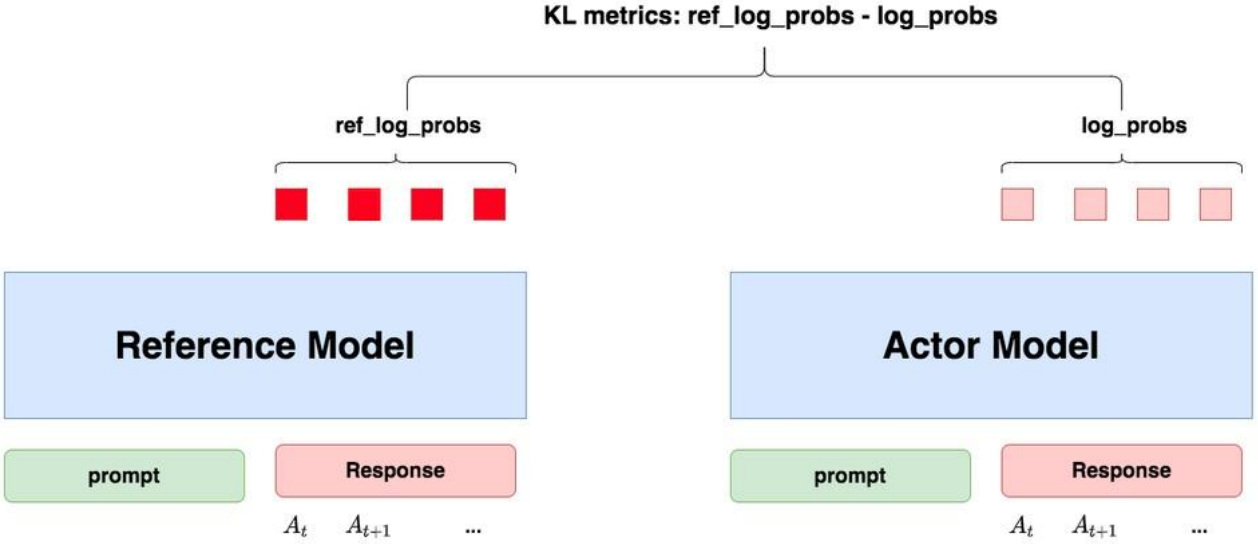
正如前文所说，**Actor 就是想要训练的目标语言模型。一般用 SFT 阶段产出的 SFT 模型来对它做初始化。**

最终目的是**让 Actor 模型能产生符合人类喜好的 response**。所以策略是，先喂给 Actor 一条 prompt （这里假设 `batch_size = 1`，所以是 1 条 prompt），让它生成对应的 response。然后，再将“prompt + response"送入我们的“奖励–loss”计算体系中去算得最后的 loss，用于更新 actor。



3.Reference Model（参考模型）

Reference Model（以下简称 Ref 模型）一般也用 SFT 阶段得到的 SFT 模型做初始化，在训练过程中，它的参数是冻结的。 Ref 模型的主要作用是防止 Actor“训歪”，那么它具体是怎么做到这一点的呢?



“防止模型训歪”换一个更详细的解释是：**希望训练出来的 Actor 模型既能达到符合人类喜好的目的，又尽量让它和 SFT 模型不要差异太大**。简言之，**希望两个模型的输出分布尽量相似**。那什么指标能用来衡量输出分布的相似度呢？自然而然想到了 **KL 散度**。

如图所示：

- **对 Actor 模型**，喂给它一个 `prompt`，它正常输出对应的 response。那么 response 中每一个 token 肯定有它对应的 log_prob 结果，把这样的结果记为 **`log_probs`**
- **对 Ref 模型**，把 Actor 生成的 `"prompt + response"` 喂给它，那么它同样能给出每个 token 的 log_prob 结果，我们记其为 **`ref_log_probs`**
- 那么这两个模型的输出分布相似度就可以用 `ref_log_probs - log_probs` 来衡量，可以从两个方面来理解这个公式：
  - **从直觉上理解**，`ref_log_probs` 越高，说明 Ref 模型对 Actor 模型输出的肯定性越大。即 Ref 模型也认为，对于某个 S_t ，输出某个 A_t 的概率也很高 $P(A_t|S_t)$ ）。这时可以认为 Actor 模型较 Ref 模型没有训歪。
  - **从 KL 散度上理解**，$KL[Actor(X)||Ref(X)] = E_{x\sim Actor(x)}[log\frac{Actor(x)}{Ref(x)}] = log\_probs - ref\_log\_probs$ （当然这里不是严格的等于，只是 KL 散度的近似），这个值越小意味着两个分布的相似性越高。

注：可能已经注意到，按照 KL 散度的定义，这里写成 `log_probs - ref_log_probs` 更合适一些。但是如果你看过一些 RLHF 相关的论文的话，可能记得在计算损失函数时，有一项 $R_t - KL$ 散度 （对这个有疑惑不要紧，我们马上在后文细说），即 KL 散度前带了负号，所以这里我写成 `ref_log_probs - log_probs` 这样的形式，更方便大家从直觉上理解这个公式。

现在，已经知道**怎么利用 Ref 模型和 KL 散度来防止 Actor 训歪了**。**KL 散度将在后续被用于 loss 的计算**。

4.Critic Model（评论家模型）

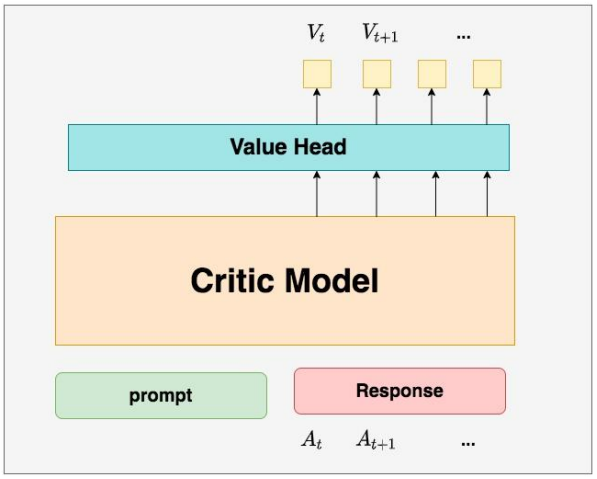
Critic Model 用于预测期望总收益 V_t ，和 Actor 模型一样，它需要做参数更新。实践中，Critic Model 的设计和初始化方式也有很多种，例如和 Actor 共享部分参数、从 RW 阶段的 Reward Model 初始化而来等等。我们讲解时，和 deepspeed-chat 的实现保持一致：从 RW 阶段的 Reward Model 初始化而来。

你可能想问：训练 Actor 模型我能理解，但我还是不明白，为什么要单独训练一个 Critic 模型用于预测收益呢?

> 这是因为，当我们在前文讨论总收益 V_t （即时 + 未来）时，我们是站在上帝视角的，也就是这个 V_t 就是客观存在的、真正的总收益。但是在训练模型时，就

> 没有这个上帝视角加成了，**也就是在 `t` 时刻，给不出客观存在的总收益 V_t ，只能训练一个模型去预测它。**

所以总结来说，在 RLHF 中，不仅要训练模型生成符合人类喜好的内容的能力（Actor），也要提升模型对人类喜好量化判断的能力（Critic）。这就是 Critic 模型存在的意义。来看看它的大致架构：



deepspeed-chat 采用了 **Reward 模型作为它的初始化**，所以这里也按 Reward 模型的架构来简单画画它。你可以简单理解成，Reward/Critic 模型和 Actor 模型的架构是很相似的（毕竟输入都一样），同时，它在最后一层增加了一个 Value Head 层，该层是个简单的线形层，用于将原始输出结果映射成单一的 V_t 值。

在图中， V_t 表示 Critic 模型对 `t` 时刻及未来（response 完成）的收益预估。

5.Reward Model（奖励模型）

Reward Model 用于**计算生成 token A_t 的即时收益**，它就是 RW 阶段所训练的奖励模型，在 RLHF 过程中，它的**参数是冻结的**。

你可能想问：为什么 Critic 模型要参与训练，而同样是和收益相关的 Reward 模型的参数就可以冻结呢？这是因为，Reward 模型是站在上帝视角的。这个上帝视角有两层含义：

- 第一点，Reward 模型是经过和“估算收益”相关的训练的，因此在 RLHF 阶段它可以直接被当作一个能产生客观值的模型。
- 第二点，Reward 模型代表的含义就是“**即时收益**”，你的 token A_t 已经产生，因此即时收益自然可以立刻算出。

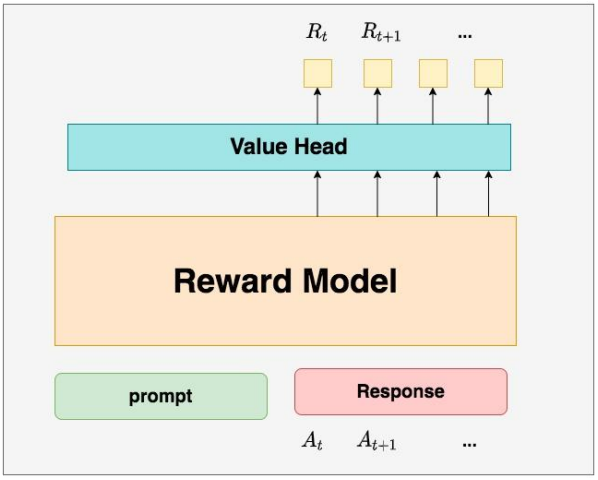
你还可能想问：已经用 Critic 预测出 V_t 了，而这个 V_t 包含了“即时”和“未来”的概念，那还需要代表“即时”的 R_t 做什么呢？直接用 V_t 不就好了吗？

为了解答这个问题，先回顾下 1.2 部分中给出的价值函数： $V_t = R_t + \gamma V_{t+1}$

这个函数告诉我们，当前可以用两个结果来表示 `t` 时刻的总收益：

- 结果 1：Critic 模型预测的 V_t
- 结果 2：Reward 模型预测的 R_t 和 critic 模型预测的 V_{t+1}

那么哪一个结果更靠近上帝视角给出的客观值呢？当然是结果 2，因为结果 1 全靠预测，而结果 2 中的 R_t 是事实数据。我们知道 Critic 模型也是参与参数更新的，可以用 `MSE(上帝视角的客观收益-Critic模型预测的收益)` 来衡量它的 loss。**但是上帝视角的客观收益是不知道的，只能用已知事实数据去逼近它，所以我们就用 $R_t + \gamma * V_{t+1}$ 来做近似。** 这就是 R_t, V_t 同时存在的意义。



Reward 模型和 critic 模型非常相似，这里就只给出架构图，不再做过多的说明。

5.RLHF 中的 loss 计算

到目前为止，已经基本了解了 RLHF 的训练框架，以及其中的四个重要角色（训练一个 RLHF，有 4 个模型在硬件上跑，可想而知对存储的压力）。在本节中，一起来解读 RLHF 的 loss 计算方式。在解读中，会再一次理一遍 RLHF 的整体训练过程，填补相关细节。在这之后，就可以来看代码解析了。

在第三部分的讲解中，我们知道 **Actor 和 Critic 模型都会做参数更新**，所以 loss 也分成 2 个：

- Actor loss：用于**评估 Actor 是否产生了符合人类喜好的结果**，将作用于 Actor 的 BWD 上。
- Critic loss：用于**评估 Critic 是否正确预测了人类的喜好**，将作用于 Critic 的 BWD 上。