PPO × Family 第一讲技术问题 QA

课程组整合梳理了自第一节课发布以来的相关技术问题答疑，希望能启发更多对于决策智能和强化学习的思考，也欢迎进一步贡献问题和参与讨论~

Q0：有没有第一节课包含的强化学习算法的大白话总结？

A0：PPO 系列算法的进化至少分为5个阶段：

- 最初， 策略梯度（REINFORCE）方法采用 episode return，理解起来很直观，但方差大；
- 然后，为了减小方差，设计了 Actor-Critic 方法，用一个 Q 函数来作估计 return，但这样方差是小，偏差又大了；
- 接着，又有人用 ” return 减去基线函数 “ 的方式来解决偏差大的问题，这就是 A2C；
- 后来，聪明人又想在可靠的方向用合适的步长来更新策略增加稳定性，引入信赖域的概念，从而有了 TRPO；
- 最后，又有大聪明说可以来个悲观约束截断一下优化目标，就有了简洁高效的 PPO，解决了前面一箩筐问题。

而在实际应用中，强化学习并不是换一个最新的算法，最强的神经网络就能获得明显的提升，影响最终结果的因素很多，其中最重要的事应该是把原本的决策问题定义成合理的 MDP，然后灵活运用经典的 RL 算法即可。虽说可能学术研究上还有一些 PPO 后续的衍生工作，但对同一个问题，一个使用者很熟悉各种算法技巧和细节的 PPO，肯定强于一个使用者很陌生的后续衍生算法。

Q1：PPO 在多智能体强化学习（multi-agent RL）的拓展和应用？

A1：课程的第 6 讲将会围绕多智能体专题展开，涉及两种核心算法 MAPPO 和 HAPPO。

Q2：PPO 算法常使用的小技巧有哪些？

A2：第一节课文字稿的附录1.4.2部分列出了一些专门研究 PPO 实际使用技巧的学术工作+链接。本课程将会在第7讲专门详细分析下这些小技巧中最重要最有用的部分。

Q3：PPO 后续有什么更先进的衍生算法吗？

A3：狭义上，PPO 这个思路走下去，由 OpenAI 官方给出的后续版本就只有PPG。但是 PPG 要引入额外一个辅助训练阶段，所以实际使用中并没有那么方便。PPO是最经久耐用的了。如果做决策智能相关应用，把PPO用熟就足够了。RL不像 CV、NLP 这些领域的一些问题，换一个最新最强的神经网络

backbone 就能获得明显的提升，影响最终结果的因素很多，其中最重要的事把原本的决策问题定义成合理的MDP，然后灵活运用经典的 RL 算法即可。虽说可能学术研究上还有一些 PPO 后续的衍生工作，但对同一个问题，用一个你很熟悉各种算法技巧和细节的 PPO，肯定强于一个你很陌生的后续衍生算法。其他之后的改进，也不是说发现了 PPO 的局限性，而是采用结合其他方法成果的方式（比如 model based RL 和 inverse RL）来让整个算法整体效果更强，所以主要放在后面的课程中展开讲。

Q4：请问有什么基于 language grounded 思想，使用例如 PPO 这样的 RL 算法去控制机械臂的工作?

A4：OpenDILab 这边 Language grounded的 RL 研究工作正在研发中，可以保持关注。主要思路可能结合 Decision Transformer 最新的一些研究进展，有兴趣的话可以参考下这里的资料总结 awesome DT 。

Q5：PPO 如何和模仿学习（BC）融合在一起?

A5：PPO 和模仿学习，尤其专指 BC 的结合方法，将在本课程第 4 讲（奖励空间部分）有所介绍。有关这个问题，怎么样既通过模仿学到专家的知识，同时又保持探索性变得越来越强，是 PPO + BC 方法在实践中最大的难题。

Q6：有什么方式可以让强化学习智能体学到尽可能多样的 solution ?

A6：强化学习探索问题中的好奇心机制正是解决这个问题，可以参考 OpenDILab 做的这个RL探索方法大合集 。

Q7：课程中讲到初始分布的随机性，对于决策算法的性能也会产生影响，这个如何理解?

A7：初始分布具体产生的影响程度要情况而定。可以回忆第一节课中轨迹生成的概率公式，初始分布虽然只有一项但是其中很重要的一部分。很多强化学习环境会需要设置随机种子来改变初始状态，提高数据多样性，避免策略过拟合到某些初态。比如 lunarlander 环境，初始的随机种子就会改变飞船的初始速度和方向，你需要让智能体能够处理各种不同的初始情况，具体信息可以参考这个文档 。

Q8：课程第一讲40分钟时讲到的“on-policy算法，each trajectory only used once”，如果使用 replay buffer是不是就变成了 off-policy 算法？（从 buffer 中采样的数据可能会被采样多次）

A8：使不使用 buffer 并不是确定 on policy 还是 off policy的关键，重要的是看训练所用数据对应的收集策略，和训练本身策略是否一致。有些 PPO 算法实现也用 buffer 来存一下数据，但用完就扔了，这并不影响 PPO 是个 on-policy 算法。关于更多 RL 基础概念定义可以参考这里的文档 。

Q9：因为 PPO 的代码实现中有两重训练循环，所以迭代一次之后参数已经更新了，这是否可以认为学习和采样的策略不是同一个?

A9：这个细节问题确实存在，虽然大家一般称 PPO 为 on policy 算法，但因为具体实现中常用两层训练循环来提高数据利用效率，所以严格来讲，只有这两层训练循环中的第一个迭代是标准的on policy，不过因为 PPO 的算法设计中有重要性采样（IS）又有 clip，所以算法的更新幅度并不会太跳脱，实用中效果不错，权衡效果和效率就成了大家现在常用的这种实现。当然，也有人比较暴力把 PPO 完全当 off policy 来用，其实效果也可以，但是超参数设置上和 on policy 版本就相差非常多了，这个课程组之后会出补充材料来讲。

Q10：PPO 可以变成确定性策略梯度算法吗？

A10：PPO 在设计上肯定是一个随机性策略算法，去学习一个策略分布。但是最终决策使用的时候可以根据场景来调，这本质上就是采样的温度系数怎么调的问题，极端情况（温度系数无穷大）下也可以当 argmax 来用。

Q11：PPO 中策略和价值网络是否要共享部分网络结构？

A11：这个问题针对不同环境和数据会有相应的分析方法，可以保持关注第三节课。共享的底层网络是用于处理输入状态的，用于特征降维，比如对于高维图片数据常常是一些卷积层。

Q12：为什么会出现作业题第一题中“状态的表征信息”这种设定呢？有什么实际应用场景吗？

A12：这一点可以理解为，智能体并不能获得最原始的状态，能拿到的输入就是转换后的表征信息。比如说，实践中的传感器误差和通信误差，多智能体局部观察视野，神经网络本身的近似误差，都可能引发这样的情况。作业题举了一个最极端的情况，就是所有状态的表征信息都变成相同的，那么第一题的题解也正说明了这种情况带来的问题。