# PPO × Family 第三讲 技术问题Q&A

**Q0: 有没有第三节课内容的大白话总结?**

**A0:**

| 小节 | 要点 | 示例任务 |
| --- | --- | --- |
| 观察空间概述 | • 特征预处理可以事半功倍<br>• 高维观察信息需要有针对性地设计网络架构<br>• 复杂问题中多种模态的观察信息 | / |
| 向量观察空间 | • 统一量纲与数值大小范围调整<br>• 特征设计与变换<br>• 编码设计 | 软体机器人控制问题 |
| 图片观察空间 | • 观察信息简化<br>• 叠帧/光流<br>• 网络深度和宽度 + clip fraction<br>• Layer Normalization | 超级马里奥 |
| 复杂结构化观察空间 | • 用 Transformer 建模元素间相关关系 | 羊了个羊 |
| 通用观察空间训练方法 | • 强化学习中的编码器 encoder<br>• 共享编码特征<br>• 自动编码器的共享与学习<br>• 共享编码器模型的优劣势 | 大鱼吃小鱼 |

**Q1: 课程中提到的考虑不变性的观察编码方式是否有更详细的说明或者其他参考资料呢?**

**A1:** 关于“不变性与等变性”部分（注意这个问题并不仅限于向量观察空间），课程新增了一个详细的解释+示例补充材料：chapter3_supp_invariance.pdf

这份补充材料将主要从不变性（Invariance）与等变性（Equivariance）这两个概念的定义出发，并结合实际问题中具体的一些函数变换，探索它们在深度强化学习算法中的适用情形与处理方法。欢迎阅读和进一步讨论。

**Q2: TRPO 的 Advantage 函数也是用神经网络拟合吗?**

**A2:** TRPO/PPO 用到的 Advantage 函数，有很多种实现方法，不过目前最主流的方式都是用神经网络学习一个Value Function，然后采用 GAE (Generalized Advantage Estimation) 算出来。在 PPO × Family 系列课程第七讲中我们将会对比介绍多种计算 Advantage的方法，大家可以持续关注一下。

**Q3: Policy Gradient 和 PPO 实现中的 entropy loss 是必须的吗？它是不是相当于一个正则项**

**A3:** 不是必须的，只是一个 “bonus”，加入到原始的 policy loss 中，根据实际的决策问题和训练情况进行调整，有些环境里就是把 entropy weight 设为 0，即不启用这个 loss 。其形式类似

$$L(\theta) = L_{\text{policy}}(\theta) - cH(\pi_\theta)$$

其中， $L_{\text{policy}}$ 是原始的 policy loss， $c$ 是一个超参数，用于控制 entropy loss 的权重。

entropy loss 是一种正则化项，用于鼓励策略探索更多的动作空间，从而提高策略的探索能力。在实现中，通常使用策略函数输出的概率分布（例如，使用 softmax 函数将神经网络的输出转换为概率分布）来计算 entropy（熵）。entropy 是一个随机变量不确定性的度量，可以用公式

$$H(\pi_\theta) = -\sum_a \pi_\theta(a) \log \pi_\theta(a)$$

来计算策略的 entropy。其中， $\pi_\theta$ 是策略函数， $a$ 是可能的动作， $\theta$ 是神经网络的权重参数。entropy 越大，策略的不确定性就越高，因此，最大化 entropy 可以帮助策略探索更多的动作空间，增强其探索能力。

**Q4: 对梯度标准化（grad norm）有什么作用呢？**

**A4:** grad norm 其实属于梯度裁剪方法的一种，主要也是稳定训练，从梯度角度去除/减弱一些梯度异常情况：

- 防止梯度爆炸。在深度神经网络中，梯度通常会在反向传播过程中逐层乘上权重矩阵，导致梯度值随着层数的增加呈指数级增长，从而可能出现梯度爆炸的现象。梯度标准化可以通过限制梯度的大小，避免出现梯度爆炸，从而提高网络的稳定性。
- 提高训练效率。梯度标准化可以控制梯度的大小，使得梯度更新更加稳定和准确。有助于加快训练速度，并提高网络的收敛性和泛化性能。

在本系列课程第七讲中我们也将会对这些 tricks 进行对比分析，目前大家可以简单了解，持续关注我们课程即可。

**Q5: PPO 学习时新旧策略相差过大会有什么影响？**

**A5:** PPO 算法的核心思想就是在更新策略时，限制新策略和旧策略的差异，以避免策略更新过大，从而影响性能。一个重要方法是我们在课程中提到的 clipped optimization。通过使用 clip，PPO 算法可以限制策略更新幅度。对于 PPO-clip 版，新旧策略过大还是有副作用的，你可以理解为 clip 只是一种减缓副作用的手段，并不能完全消除副作用。毕竟 clip 后的结果还是会产生一份梯度，clip 只是让梯度不要”那么离谱"。实际上，clip 是一种更简明但有效的实现方式，在对数据多样性损害不太大的情况下显著提高策略优化的稳定性。

**Q6: 强化学习如何增加智能体的探索？**

**A6:** 对于 PPO，最简单的方式有两招：

- 使用 entropy bonus，并适当增大这一项的权重，比如从 0 改成 0.001 or 0.01

- 在收集数据阶段，挑选动作时，如果是离散动作空间，操控 softmax 采样即可 (常用的就是加个 temperature 来调)，如果是连续动作空间，可以适当把 log_sigma 初始化的大一点，或者手动来设置常数 sigma

更复杂的 PPO + 探索技巧，我们会在后续发布的第四节课中详述。

**Q7: 一般怎么判断 PPO 算法的策略已经学到收敛了呢?**

**A7:** 判断收敛，一般在足够有多样性（最简单比如多个随机种子）的测试环境上评估多个 episode，然后看 episode return 的各种统计值，或者是把这些评估结果的 replay 视频存出来，看看智能体的行为是否满足期望比如自动驾驶模拟里。可能会出现虽然能开到终点但是车开的非常抖的情况。

**Q8: 是否有关于 AlphaZero 等 MCTS 相关算法的讲解；在棋类问题上，是否 MCTS 要比 PPO 在性能上有优势呢?**

**A8:** 我们最近正在准备推出 MCTS 相关的系列博客，会讲解从 AlphaGo 开始一直到最近 MuZero 相关最新的研究工作。另外也希望大家能够更多推荐和分享我们的 PPO × Family 课程，后续反响好的话我们会计划推出 MCTS 相关的系列课程。

MCTS 是一种**基于树搜索**的算法，它将当前状态作为根节点，通过遍历状态空间，不断扩展子树，以最大化收益。它在围棋、五子棋等复杂游戏中表现出色，因为这些游戏的状态空间非常大，而 MCTS 可以通过遍历空间寻找最优解。MCTS 的优点是可以很好地处理高维、大规模、连续动作空间的问题，同时具有较好的收敛性能。

PPO 是一种**基于策略优化的无模型 (model-free)** 强化学习算法，它直接从经验中学习参数化策略。对于棋类问题，因为环境的状态转移函数是确定性的，所以很适合用 MCTS 系列算法，数据利用效率相比 PPO 这样的 model-free RL 算法更高。