Actor-Critic

Actor 是策略网络，用来控制agent运动，可以看做是运动员。critic 是价值网络，用来给动作打分，看作是裁判。Actor Critic 是把价值学习和策略学习结合起来 （value based和policy based）

价值网络与策略网络构建

状态价值函数：

$$V_\pi(s) = \sum_a \pi(a|s) \cdot Q_\pi(s,a) \text{ (离散)}$$

它是动作价值函数 Q_π 的期望， $\pi(s|a)$ 策略函数控制agent做运动， $Q_\pi(s,a)$ 价值函数评价动作好坏。但是上述这两个函数我们都不知道，但是可以分别用一个神经网络来近似这两个函数，然后用Actor&Critic方法来同时学习这两个网络。

策略网络(actor)：用网络 $\pi(s|a;\theta)$ 来近似 $\pi(s|a)$ ， θ 是网络参数

价值网络(critic)：用网络 $q(s,a;w)$ 来近似 $Q_\pi(s,a)$ ， w 是网络参数

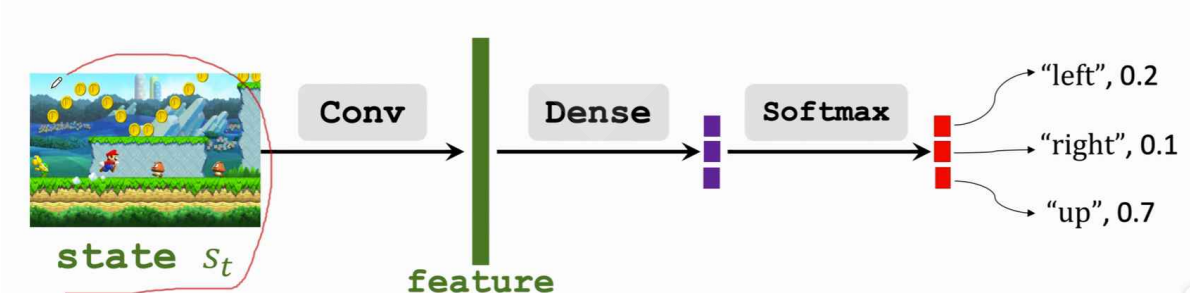
actor是一个体操运动员，她可以自己做动作，她想要做的更好，但是不知道怎么改进，这就需要裁判给她打分，这样运动员就知道什么样动作的分数高，什么样动作的分数低，这样就能改进自己，让分数越来越高。

这样： $V_\pi(s) = \sum_a \pi(a|s) \cdot Q_\pi(s,a) \approx \sum_a \pi(a|s;\theta) \cdot q(s,a;w)$

状态价值函数由策略网络和价值网络共同决定。

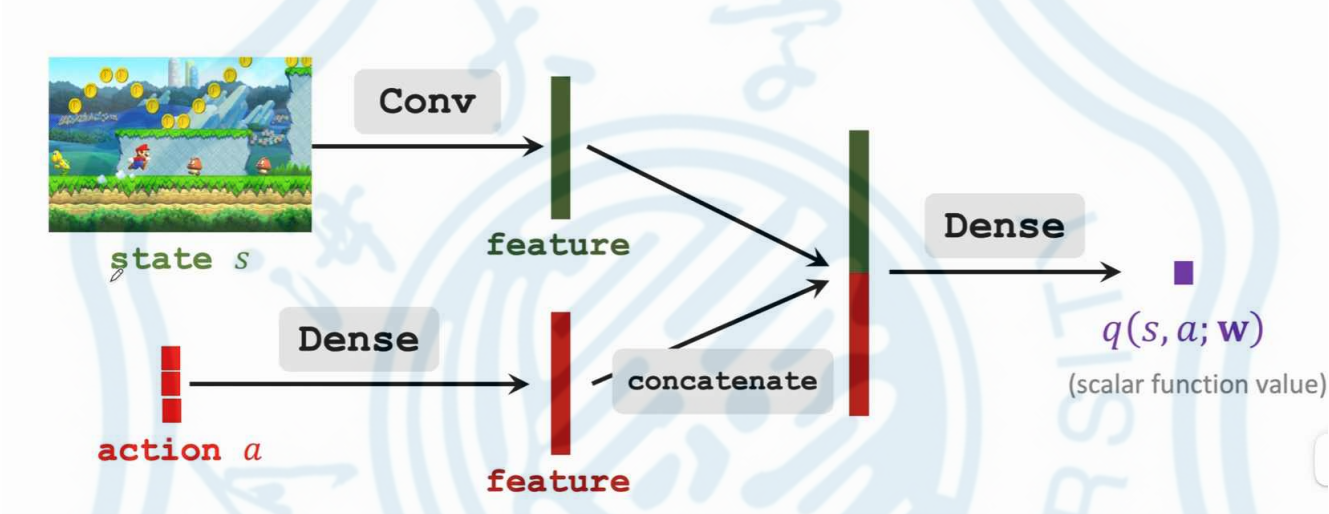
Actor搭建

1. 输入：状态 s
2. 输出：可能的动作分布
3. $\mathcal{A}$ 是动作集，如 $\mathcal{A} = \{"left","right","up"\}$
4. $\sum_{a\in\mathcal{A}} \pi(a|s,\theta) = 1$



Critic搭建

1. 输入：状态 s 和动作 a
2. 输出：近似的动作价值函数(scalar)



动作如果是离散的，可以用one-hot coding来表示，比如向左为[1,0,0]，向右为[0,1,0] · · · · · · 分别用卷积层与全连接层从输入中提取特征，得到两个特征向量，然后把这两个特征向量拼接起来，得到一个更高的特征向量，最后用一个全连接层输出一个实数，这个实数就是裁判给运动员打的分数。这个动作说明，处在状态 s 的情况下，做出动作 a 是好还是坏。这个价值网络可以与策略网络共享卷积层参数，亦可以跟策略网络完全独立。

Actor-Critic Method

同时训练策略网络与动作网络就称为Actor-Critic Method。

定义：使用神经网络（策略网络和价值网络）共同来近似状态价值函数

$$V(s;\theta,w)=\sum_{a}\pi(s|a;\theta)\cdot q(s,a;w)$$

训练：更新参数 θ、w

- 更新策略网络 $\pi(s|a;\theta)$ 是为了让 $V(s;\theta,w)$ 的值增加
  a. 监督信号单纯是由价值网络提供的
  b. actor根据裁判critic的打分来不断提高自己的水平

- 更新价值网络 $q(s,a;w)$ 是为了让critic打分更精准
  a. 监督信号纯粹来自奖励（奖励就像监督委员会给裁判的判罚水平一个矫正）
  b. 一开始裁判是随意乱打分，但是会根据环境给的奖励提高打分水平，使其接近真实打分。

步骤总结：

1. 观测状态 s_t
2. s_t 作为输入，根据策略网络 $\pi(\cdot|s_t;\theta_t)$ 随机采样一个动作 a_t
3. 实施动作 a_t 并观测新状态 s_{t+1} 以及奖励 r_t
4. 用奖励 r_t 通过 TD 算法在价值网络q中更新 w，让裁判变得更准确
5. 使用策略梯度在策略网络中更新 θ ，让运动员技术更好

两个神经网络参数的更新细节如下：

TD 更新价值网络，让裁判打分更准确

- 给动作 a_t、a_{t+1} 打分：计算 $q(s_t,a_t;w_t)$与$q(s_{t+1},a_{t+1};w_t)$
- TD target： $y_t=r_t+\gamma\cdot q(s_{t+1},a_{t+1};w_t)$ ，相对于 $q(s_t,a_t;w_t)$ 更真实一些
- 损失函数是预测值与部分真实值之间的差。

$$\text{Loss: } L(w)=\frac{1}{2}[q(s_t,a_t;w)-y_t]^2$$

- 梯度下降： $W_{t+1}=w_t-\alpha\cdot\frac{\partial L(w)}{\partial w}|w=w_t$

策略梯度更新策略网络，让状态价值增加

$$V(s;\theta,w)=\sum_a \pi(s|a;\theta)\cdot q(s,a;w)$$

状态价值函数 V 相当于运动员所有动作的平均分

策略梯度：函数 $V(s;\theta,w)$ 关于参数 θ 的导数

- $g(a,\theta)=\frac{\partial\ log\ \pi(a|s;\theta)}{\partial\theta}\cdot q(s,a;w)$,这里 q 相当于裁判的打分
- $\frac{\partial V(s;\theta;w_t)}{\partial\theta}=\mathbb{E}_A[g(A,\theta)]$ 策略梯度相当于对函数 g 求期望，把 A 给消掉，但是很难求期望，所以通常用蒙特卡洛近似取样。

算法：

根据策略网络随机抽样得到动作 a： $a\sim\pi(\cdot|s_t;\theta_t)$（ $g(a,\theta)$ 是无偏估计）

有了随机梯度g，可以做一次梯度上升： $\theta_{t+1}=\theta_t+\beta\cdot g(a,\theta_t)$

算法总结

1. 观测旧状态 s_t ，根据策略网络 $\pi(\cdot|s_t;\theta_t)$ 随机采样一个动作 a_t
2. 执行动作 a_t ；环境会告诉我们新的状态 s_{t+1} 和奖励 r_t
3. 拿新的状态 s_{t+1} 作为输入，用策略网络 π 计算出新的概率并采样新的动作：$\tilde{a}_{t+1} \sim \pi(\cdot|s_{t+1};\theta_t)$,这个动作只是假想的动作，agent不会执行，拿来算下 Q 值。
4. 算两次价值网络的输出： $q_t = q(s_t, a_t; w_t)$和$q_{t+1} = q(s_{t+1}, \tilde{a}_{t+1}; w_t)$ ，$\tilde{a}_{t+1}$ 用完就丢掉了，并不会真正执行
5. 计算TD error： $\delta_t = q_t - \underbrace{(r_t + \gamma \cdot q_{t+1})}_{TD\ target}$
6. 对价值网络求导： $d_{w,t} = \dfrac{\partial q(s_t, a_t; w)}{\partial w}|w = w_t$
7. 更新价值网络，让裁判打分更精准： $w_{t+1} = w_t - \alpha \cdot \delta_t \cdot d_{w,t}$
8. 对策略网络 π 求导： $d_{\theta,t} = \dfrac{\partial log\pi(a_t|s_t,\theta)}{\partial \theta}|\theta = \theta_t$
9. 用梯度上升来更新策略网络，让运动员平均成绩更高： $\theta_{t+1} = \theta_t + \beta \cdot q_t \cdot d_{\theta,t}$ ，这里 $q_t \cdot d_{\theta,t}$ 是策略梯度的蒙特卡洛近似。

- 每一轮迭代做以上9个步骤，且只做一次动作，观测一次奖励，更新一次神经网络参数。
- 根据策略梯度算法推导，算法第九步用到了 q_t ，它是裁判给动作打的分数，书和论文通常拿 δ_t 来替代 q_t 。 q_t 是标准算法， δ_t 是Policy Gradient With Baseline(效果更好)，都是对的，算出来期望也相等。
- 为什么baseline效果更好，因为可以更好的计算方差，更快的收敛。

总结

状态价值函数： $V_\pi(s) = \sum_a \pi(a|s) \cdot Q_\pi(s,a)$ ，越大越好

- 但是直接学 π 函数不容易，用神经网络-策略网络 $\pi(s|a;\theta)$ 来近似
- 算策略梯度的时候有个困难就是不知道动作价值函数 Q_π ，所以要用神经网络-价值网络 $q(s,a;w)$ 来近似

在训练时：

- agent由策略网络(actor)给出动作 $a_t \sim \pi(\cdot|s_t;\theta)$
- 价值网络 q 辅助训练 π ，给出评分

训练后：

- 还是由策略网络给出动作 $a_t \sim \pi(\cdot|s_t;\theta)$
- 价值网络 q 不再使用

如何训练：

用策略梯度更新策略网络：

- 尽可能提升状态价值： $V_\pi(s) = \sum_a \pi(a|s;\theta) \cdot q(s,a;w)$
- 计算策略梯度，蒙特卡洛算： $\frac{\partial V(s;\theta;w_t)}{\partial \theta} = \mathbb{E}_A[\frac{\partial\ log\ \pi(a|s;\theta)}{\partial \theta} \cdot q(s,a;w)]$
- 执行梯度上升。

TD 算法更新价值网络

- $q_t = q(s_t,a_t;w)$ 是价值网络是对期望回报的估计
- TD target: $y_t = r_t + \gamma \cdot \underset{a}{max}\, q(s_{t+1},a_{t+1};w)$ ，y_t 也是价值网络是对期望回报，不过它用到了真实奖励，因此更靠谱一点，所以将其作为 target，相当于机器学习中的标签。
- 把 q_t 与 y_t 差值平方作为损失函数计算梯度： $\frac{\partial (q_t - y_t)^2/2}{\partial w} = (q_t - y_t) \cdot \frac{\partial q(s_t,a_t;w)}{\partial w}$
- 梯度下降，缩小 q_t 与 y_t 差距。