策略学习

策略函数

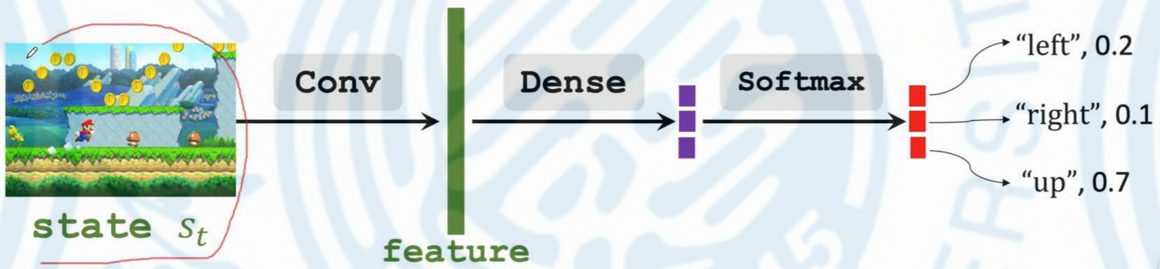
- 策略函数 $\pi(a|s)$ 的输入是状态 s
- 输出是一个概率分布，给每一个动作附上一个概率值。

策略网络

用神经网络来近似策略函数

- 用策略网络 $\pi(a|s;\theta)$ 来近似 $\pi(a|s)$
- θ 是神经网络的参数

超级玛丽设计：



状态画面经过卷积提取特征，特征经过全连接层再通过softmax层得到一个动作的概率分布，动作的概率集合全部加起来要等于1。

状态价值函数回顾与近似

折扣回报函数：

- $U_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \gamma^3 R_{t+3} + \cdots$

动作价值函数：

- $Q_\pi(s_t, a_t) = \mathbb{E}[U_t|S_t = s_t, A_t = a_t]$

状态价值函数（包含了动作价值函数）

- $V_\pi(s_t) = \mathbb{E}_A[Q_\pi(s_t, A)]$
- 消掉了动作 A，这样 V_π 只跟状态 s 与策略函数 π 有关了。可以评价当前状态的好坏

展开：

- $V_\pi(s_t) = \mathbb{E}_A[Q_\pi(s_t, A)] = \sum_a \pi(a|s_t) \cdot Q_\pi(s_t, a)$ 这里动作是离散的。
- $V_\pi(s_t) = \mathbb{E}_A[Q_\pi(s_t, A)] = \int \pi(a|s_t) \cdot Q_\pi(s_t, a)da$ 这里动作是连续的。

策略学习

💡 得到状态价值函数： $V_\pi(s_t) = \mathbb{E}_A[Q_\pi(s_t, A)] = \sum_a \pi(a|s_t) \cdot Q_\pi(s_t, a)$

近似状态价值函数：

- 用策略网络 $\pi(a|s;\theta)$ 来近似 $\pi(a|s)$
- 把 $\pi(a|s_t)$ 函数替换成 $\pi(a|s_t;\theta)$ ： $V(s_t;\theta) = \sum_a \pi(a|s_t;\theta) \cdot Q_\pi(s_t, a)$

这样，状态价值函数就可以写成： $V(s;\theta)$ ， V 可以评价策略网络的好坏，给定状态 S ，策略网络越好那么 V 的值就越大。可以改进参数 θ ，让 $V(s;\theta)$ 变大。

基于上述想法，可以把目标函数定义为 $V(s;\theta)$ 的期望： $J(\theta) = \mathbb{E}_S[V(S;\theta)]$ ，期望是关于状态 S 求的。这样目标就是改进 θ ，使得 $J(\theta)$ 越大越好。

💡 如何改进 θ ？用策略梯度算法(Policy gradient ascent)

- 观测到状态 s，这个 s 是从状态的概率分布中随机抽样出来的。
- 把 $V(s;\theta)$ 关于 s 求导可以得到一个梯度，然后用梯度上升来更新 θ ， β 是学习率。

$$\theta \leftarrow \theta + \beta \cdot \frac{\partial V(s;\theta)}{\partial \theta}$$

注意：我们这里算的是 V 关于 θ 的倒数，就是个随机梯度，随机性来源于状态 s

为什么要用梯度上升，因为我们想让目标函数 $J(\theta)$ 变得越来越大。

其中 $\frac{\partial V(s;\theta)}{\partial \theta}$ 被叫做 Policy gradient 策略梯度。

策略梯度

📌 如何计算 $\frac{\partial V(s;\theta)}{\partial \theta}$ 策略梯度?

$$\begin{aligned}\frac{\partial V(s;\theta)}{\partial \theta} &= \frac{\partial \sum_a \pi(a|s;\theta)\cdot Q_\pi(s,a)}{\partial \theta}\\ &= \sum_a \frac{\partial \pi(a|s;\theta)\cdot Q_\pi(s,a)}{\partial \theta}\end{aligned}$$

可以把求导运算推到连加里面去，连加的导数等于导数的连加

$$= \sum_a \frac{\partial \pi(a|s;\theta)}{\partial \theta}\cdot Q_\pi(s,a)$$ 把 $Q_\pi(s,a)$ 提取出来,因为假设它不依赖于 θ

。

得到策略梯度公式的一种形式：

$$\frac{\partial V(s;\theta)}{\partial \theta} = \sum_a \frac{\partial \pi(a|s;\theta)}{\partial \theta}\cdot Q_\pi(s,a)$$

📌 但实际中并不会用这个形式公式，一般是用其蒙特卡洛近似：

$$\begin{aligned}\frac{\partial V(s;\theta)}{\partial \theta} &= \sum_a \frac{\partial \pi(a|s;\theta)}{\partial \theta}\cdot Q_\pi(s,a)\\ &= \sum_a \pi(a|s;\theta)\cdot \frac{\partial\ log\ \pi(a|s;\theta)}{\partial \theta}\cdot Q_\pi(s,a)\end{aligned}$$

上面转换可以链式法则： $\frac{\partial log[\pi(\theta)]}{\partial \theta} = \frac{1}{\pi(\theta)}\cdot\frac{\partial \pi(\theta)}{\partial \theta}$ 反推

$$= \mathbb{E}_A[\frac{\partial\ log\ \pi(A|s;\theta)}{\partial \theta}\cdot Q_\pi(s,A)]$$

把 a 当作随机变量 A， $\pi(a|s;\theta)$ 是 a 的概率密度函数。

上述推导不够严谨，但是可以使用。

综上，推导出了策略梯度两种形式

$$\text{Form1:}\quad \frac{\partial V(s;\theta)}{\partial \theta} = \sum_a \frac{\partial \pi(a|s;\theta)}{\partial \theta}\cdot Q_\pi(s,a)$$

$$\text{Form2:}\quad \frac{\partial V(s;\theta)}{\partial \theta} = \mathbb{E}_{A\sim\pi(\cdot|s;\theta)}[\frac{\partial\ log\ \pi(A|s;\theta)}{\partial \theta}\cdot Q_\pi(s,A)]$$

计算随机梯度

📌 对于离散的动作，动作集是 $\mathcal{A}=\{"left","right","up"\}$

使用第一种形式的公式: $\frac{\partial V(s;\theta)}{\partial\theta}=\sum_a\frac{\partial\pi(a|s;\theta)}{\partial\theta}\cdot Q_\pi(s,a)$

- 对于每个动作 a，都把 $f(a,\theta)=\frac{\partial\pi(a|s;\theta)}{\partial\theta}\cdot Q_\pi(s,a)$ 计算出来
- 根据上面公式，策略梯度就是把这些所有的动作 $f(a,\theta)$ 都加起来，
  $\frac{\partial V(s;\theta)}{\partial\theta}=f("left",\theta)+f("right",\theta)+f("up",\theta)$ （这是离散的情况）

📌 对于连续的动作，如 $\mathcal{A}=[0,1]$

使用第二种形式的公式: $\frac{\partial V(s;\theta)}{\partial\theta}=\mathbb{E}_{A\sim\pi(\cdot|s;\theta)}[\frac{\partial\ log\ \pi(A|s;\theta)}{\partial\theta}\cdot Q_\pi(s,A)]$

A 是连续变量，所以想要直接求期望需要做定积分。但是 π 函数是个神经网络，没法用数学公式求积分，可以用蒙特卡洛近似求取。

蒙特卡洛近似

蒙特卡洛就是抽一个或多个随机样本，用随机样本来近似期望。

1. 根据概率密度函数 $\pi(\cdot|s;\theta)$ 随机抽样一个动作 $\hat{a}$ ，比如[0,1]这个集合，抽样得到0.2
2. 计算 $g(\hat{a},\theta)=\frac{\partial\ log\ \pi(\hat{a}|s;\theta)}{\partial\theta}\cdot Q_\pi(s,\hat{a})$ （g就是gradient，计算策略梯度）

- 根据定义， $\mathbb{E}_A[g(A,\theta)]=\frac{\partial V(s;\theta)}{\partial\theta}$
- 因为 $\hat{a}$ 是概率密度函数 $\pi(\cdot|s;\theta)$ 随机抽样得来的，所以 $g(\hat{a},\theta)$ 是策略密度函数 $\frac{\partial V(s;\theta)}{\partial\theta}$ 的一个无偏估计
- 由于是无偏估计，所以可以用 $g(\hat{a},\theta)$ 来对 $\frac{\partial V(s;\theta)}{\partial\theta}$ 做一个近似，更新模型梯度的时候用 $g(\hat{a},\theta)$ 来更新就行了。

这种方式对离散的动作也是可以使用的。

策略梯度算法过程

1. 观测到状态 s_t ，接下来用蒙特卡洛近似来计算策略梯度
2. 把策略网络 $\pi(\cdot|s;\theta)$ 作为概率密度函数随机采样动作 a_t 。
3. 计算价值函数的值，记作 $q_t\approx Q_\pi(s_t,a_t)$

4. 对策略网络求导，得到向量矩阵或者张量： $d_{\theta,t}=\frac{\partial log\pi(a_t|s_t,\theta)}{\partial\theta}|\theta=\theta_t$
5. 近似计算策略梯度： $g(a_t,\theta_t)=q_t\cdot d_{\theta,t}$
6. 更新策略网络： $\theta_{t+1}=\theta_t+\beta\cdot g(a_t,\theta_t)$

问题：第三步动作价值函数 $Q_\pi(s_t,a_t)$ 是啥，如何计算?

方法1：reinforce 算法

用策略网络 π 来控制 agent 运动，从一开始玩到游戏结束，把整个游戏轨迹都记录下来

$s_1,a_1,r_1,s_2,a_2,r_2,\cdots,s_t,a_t,r_t$

观测到所有奖励 r ,就可以算出折扣回报 $u_t=\sum_{k=t}^{T}\gamma^{k-t}r_k$ 。

由于 $Q_\pi(s_t,a_t)=\mathbb{E}[U_t]$ ，所以可以使用 u_t 来近似 $Q_\pi(s_t,a_t)$

使用 $q_t=u_t$

就是用观测到的 u_t 来代替 $Q_\pi(s_t,a_t)$ 函数

方法2：用一个神经网络来近似 Q_π

原本是拿神经网络来近似一个策略函数 π ，对于两个神经网络就涉及到了Actor-Critic。