1. 背景

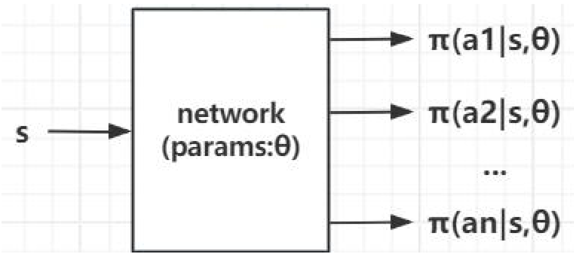
在我们之前的学习中，策略都是用表格形式进行表达的，我们可以直接访问或者修改表格中的值，如下所示：

	a_1	a_2	a_3	a_4	a_5
s_1	$\pi(a_1\|s_1)$	$\pi(a_2\|s_1)$	$\pi(a_3\|s_1)$	$\pi(a_4\|s_1)$	$\pi(a_5\|s_1)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
s_9	$\pi(a_1\|s_9)$	$\pi(a_2\|s_9)$	$\pi(a_3\|s_9)$	$\pi(a_4\|s_9)$	$\pi(a_5\|s_9)$

现在，策略可以通过函数形式来进行表达：

$$\pi(a|s,\theta)$$

其中θ是多维的参数向量，现在用的最广泛的函数形式就是神经网络，如图所示：



这种表示方法的优点为：

- 节省存储空间，我们只需要存储参数向量θ
- 泛化能力：当我们访问了某个(s,a)时，对策略进行更新会导致参数向量θ更新，其他(s,a)的策略也会被更新

tabular 与 functional 表示方式的不同：

（1） 如何定义最优策略？

在 tabular representation 中，optimal policy 定义为：$\forall s, v_{\pi^*}(s) \geq v_\pi(s)$

而在 functional representation 中，使用 metric/目标函数$J(\theta)$来定义，当它取得最大值时，为 optimal policy

（2） 如何去得到某个 action 的概率？

在 tabular representation 中，通过直接访问即可得到；而在 functional representation 中，通过计算一次得到

（3） 如何更新策略？

在 tabular representation 中，通过直接访问修改数值即可；而在 functional representation 中，只能通过修改参数向量θ来更新策略

2. average value

一共有两种 metric，第一种 metric 为 average state value，简称为 average value，它的定义为：

$$\bar{v}_\pi = \sum_{s\in S} d(s) v_\pi(s)$$

$\bar{v}_\pi$其实就是 state value 的加权平均，其中$d(s)$表示状态 s 的权重，有$\sum_{s\in S} d(s) = 1$，因此 metric 也可以写成：

$$\bar{v}_\pi = E[v_\pi(S)] \ where \ S \sim d$$

也可以写成向量点积形式：

$$\bar{v}_\pi = \sum_{s\in S} d(s) v_\pi(s) = d^T v_\pi$$

其中，

$$v_\pi = [\dots, v_\pi(s), \dots]^T \in R^{|S|}$$
$$d = [\dots, d(s), \dots]^T \in R^{|S|}$$

$v_\pi(s)$表示状态 s 的 state value，$d(s)$表示状态 s 的权重

选择d的分布有两种方法：

（1） 与策略π无关，将d记为d_0

- 均匀分布：每个状态一样重要，$d_0 = 1/|S|$
- 在某些任务中 episode 总是从一个状态s_0出发，我们只考虑从s_0出发得到的

return，则使$d(s_0)=1, d(s\neq s_0)=0$

（2） 与策略π相关，将d记为d_π

stationary distribution：稳态分布，详细介绍见上章

d_π满足：$d_\pi^T=d_\pi^T P_\pi$

3. average reward

第二种 metric 为 average one-step reward，简称为 average reward，它的定义为：

$$\bar{r}_\pi=\sum_{s\in S} d_\pi(s) r_\pi(s)=E[r_\pi(S)]\ where\ S\sim d_\pi$$

其中，$r_\pi(s)$的定义为：

$$r_\pi(s)=\sum_{a\in A}\pi(a|s)r(s,a)$$

它表示从状态 s 出发得到的 immediate reward，而其中：

$$r(s,a)=E[R|s,a]=\sum_r rp(r|s,a)$$

假设在给定策略π下，有这样一条 trajectory，它从状态s_0出发，得到的 reward 为$(R_{t+1},R_{t+2},\ldots)$，则该 trajectory 的 average single reward 为：

$$\begin{aligned}
&\lim_{n\to\infty}\frac{1}{n}E[R_{t+1}+R_{t+2}+\ldots+R_{t+n}|S_t=s_0]\\
&=\lim_{n\to\infty}\frac{1}{n}E\left[\sum_{k=1}^{n}R_{t+k}\middle|S_t=s_0\right]\\
&=\lim_{n\to\infty}\frac{1}{n}E\left[\sum_{k=1}^{n}R_{t+k}\right]\\
&=\sum_{s\in S}d_\pi(s)r_\pi(s)=\bar{r}_\pi
\end{aligned}$$

- 开始状态s_0不起作用，即你跑了无穷多步之后，从哪个状态开始已经不重要了
- $\bar{r}_\pi$的这两种定义是等价的

由上述介绍可知：

- 这两个 metric 都是关于策略π的函数
- 因为策略π与参数向量θ有关，所以这两个 metric 也是关于参数向量θ的函数
- 换言之，不同取值的参数向量θ会得到不同的两个 metric 值
- 因此，我们可以寻找最优取值θ来使这两个 metric 达到最大

$\bar{r}_\pi$看似比$\bar{v}_\pi$更加的短视，因为它只考虑 immediate reward，而$\bar{v}_\pi$考虑 total reward，实际上它们两是等价的（等价不是说它们两相等，而是说当一个达到最大值时，另一个也会达到最大值），当 discount rate $\gamma < 1$ 时，有：

$$\bar{r}_\pi = (1-\gamma)\bar{v}_\pi$$

考虑这样一个函数，它与我们介绍的两种 metric 的关系是什么？

$$J(\theta) = E\left[\sum_{t=0}^{\infty}\gamma^t R_{t+1}\right]$$

实际上它就是$\bar{v}_\pi$，证明如下：

$$E\left[\sum_{t=0}^{\infty}\gamma^t R_{t+1}\right] = \sum_{s\in S} d(s)E\left[\sum_{t=0}^{\infty}\gamma^t R_{t+1}\middle| S_0 = s\right]$$

$$= \sum_{s\in S} d(s)E[G_t|S_0 = s] = \sum_{s\in S} d(s)v_\pi(s)$$

$$= \bar{v}_\pi$$

4. 计算这些 metric 的梯度

给出一个总公式：

$$\nabla_\theta J(\theta) = \sum_{s\in S}\eta(s)\sum_{a\in A}\nabla_\theta \pi(a|s,\theta)q_\pi(s,a)$$

- $J(\theta)$可以是$\bar{v}_\pi, \bar{r}_\pi, \bar{v}_\pi^0$（表示$d(s)$与策略无关）

- 这里的”=”可以是严格相等、近似相等或成比例相等
- $\eta(s)$是状态 s 的分布或权重

给出一些确切的结果：

$$\nabla_\theta \bar{r}_\pi = \sum_{s\in S} d_\pi(s) \sum_{a\in A} \nabla_\theta \pi(a|s,\theta) q_\pi(s,a)$$

当为 discounted case（即$\gamma \in [0,1)$时）它是近似相等；当为 undiscounted case（即 $\gamma = 1$ 时）它是严格相等

$$\nabla_\theta \bar{v}_\pi = \frac{1}{1-\gamma} \nabla_\theta \bar{r}_\pi$$

$$\nabla_\theta \bar{v}_\pi^0 = \sum_{s\in S} \rho_\pi(s) \sum_{a\in A} \nabla_\theta \pi(a|s,\theta) q_\pi(s,a)$$

$\rho_\pi(s)$是另外一种分布

该梯度可以写成这样一种形式：

$$\nabla_\theta J(\theta) = \sum_{s\in S} \eta(s) \sum_{a\in A} \nabla_\theta \pi(a|s,\theta) q_\pi(s,a)$$

$$= E[\nabla_\theta \ln \pi(A|S,\theta)\, q_\pi(S,A)] \; where \; S\sim\eta \; and \; A\sim\pi(A|S,\theta)$$

这个式子便于我们使用 stochastic gradient：

$$\nabla_\theta J \approx \nabla_\theta \ln \pi(a|s,\theta)\, q_\pi(s,a)$$

如何转换为这个式子，证明如下：

$$\nabla_\theta \ln \pi(a|s,\theta) = \frac{\nabla_\theta \pi(a|s,\theta)}{\pi(a|s,\theta)}$$

则有：

$$\nabla_\theta \pi(a|s,\theta) = \pi(a|s,\theta) \nabla_\theta \ln \pi(a|s,\theta)$$

代入原式中得到：

$$\nabla_\theta J = \sum_{s\in S} \eta(s) \sum_{a\in A} \pi(a|s,\theta) \nabla_\theta \ln \pi(a|s,\theta)\, q_\pi(s,a)$$

$$= E_{S\sim\eta}\left[\sum_{a\in A} \pi(a|S,\theta) \nabla_\theta \ln \pi(a|S,\theta)\, q_\pi(S,a)\right]$$

$$= E_{S\sim\eta,\,A\sim\pi}[\nabla_\theta \ln \pi(A|S,\theta)\, q_\pi(S,A)]$$
$$= E[\nabla_\theta\, ln\, \pi(A|S,\theta)\, q_\pi(S,A)]$$

由函数$\ln x$的定义域可知，我们要保证$\pi(a|s,\theta) > 0\ \forall s,a$，我们可以通过 softmax 来达到这个目的，softmax 函数定义为：

有向量$x = [x_1, \dots, x_n]^T$，

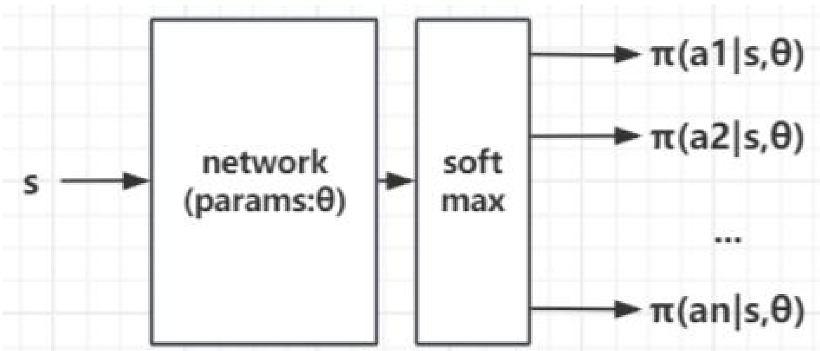
$$z_i = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$

由此可知，$z_i \in (0,1)\ and\ \sum_{i=1}^{n} z_i = 1$

则策略函数可以表示为：

$$\pi(a|s,\theta) = \frac{e^{h(s,a,\theta)}}{\sum_{a'\in A} e^{h(s,a',\theta)}}$$

$h(s,a,\theta)$是另外一个函数，一般为神经网络，则策略函数的图示为：



在该方式下，对于状态 s 的所有动作的概率都是大于 0 的，因此策略是 stochastic 和 exploratory

5. 梯度上升算法

为了最大化 metric $J(\theta)$，采用梯度上升算法对θ进行优化：

$$\theta_{t+1} = \theta_t + \alpha\nabla_\theta J(\theta_t) = \theta_t + \alpha E[\nabla_\theta\, ln\, \pi(A|S,\theta_t)\, q_\pi(S,A)]$$

由于涉及到计算期望，而期望难以计算，使用 stochastic gradient 代替 true gradient

得到：

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta \ln \pi(a_t|s_t, \theta_t)\, q_\pi(s_t, a_t)$$

由于q_π未知，我们对其进行近似或采样：

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta \ln \pi(a_t|s_t, \theta_t)\, q_t(s_t, a_t)$$

有多种不同的方法来近似计算q_π：

- 蒙特卡洛方法进行近似，与策略梯度上升算法结合后，该算法名为 REINFORCE
- TD 算法以及其他算法

使用蒙特卡洛方法进行近似，它的伪代码为：

Initialization: A parameterized function $\pi(a|s,\theta)$, $\gamma \in (0,1)$, and $\alpha > 0$.
Aim: Search for an optimal policy maximizing $J(\theta)$.

For the kth iteration, do

  Select s_0 and generate an episode following $\pi(\theta_k)$. Suppose the episode is $\{s_0, a_0, r_1, \ldots, s_{T-1}, a_{T-1}, r_T\}$.

  For $t = 0, 1, \ldots, T-1$, do

    Value update: $q_t(s_t, a_t) = \sum_{k=t+1}^{T} \gamma^{k-t-1} r_k$

    Policy update: $\theta_{t+1} = \theta_t + \alpha \nabla_\theta \ln \pi(a_t|s_t, \theta_t) q_t(s_t, a_t)$

  $\theta_k = \theta_T$

由于有：

$$\nabla_\theta \ln \pi(a|s,\theta) = \frac{\nabla_\theta \pi(a|s,\theta)}{\pi(a|s,\theta)}$$

则将式子转换回去可以表示为：

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta \ln \pi(a_t|s_t, \theta_t)\, q_t(s_t, a_t) = \theta_t + \alpha \left(\frac{q_t(s_t, a_t)}{\pi(a_t|s_t, \theta_t)} \right) \nabla_\theta \pi(a_t|s_t, \theta_t)$$

将$\frac{q_t(s_t,a_t)}{\pi(a_t|s_t,\theta_t)}$记为$\beta_t$，则有：

$$\theta_{t+1} = \theta_t + \alpha \beta_t \nabla_\theta \pi(a_t|s_t, \theta_t)$$

假设当$\alpha\beta_t$足够小时，有：

- 若$\beta_t > 0$，则选择(s_t, a_t)的概率会增大，即：$\pi(a_t|s_t, \theta_{t+1}) > \pi(a_t|s_t, \theta_t)$，且$\beta_t$越大，增大得越多
- 若$\beta_t < 0$，则有$\pi(a_t|s_t, \theta_{t+1}) < \pi(a_t|s_t, \theta_t)$

证明如下：

当$\theta_{t+1} - \theta_t$足够小时，有：

$$\pi(a_t|s_t, \boldsymbol{\theta_{t+1}}) \approx \pi(a_t|s_t, \boldsymbol{\theta_t}) + \left(\nabla_\theta \pi(a_t|s_t, \boldsymbol{\theta_t})\right)^T (\boldsymbol{\theta_{t+1} - \theta_t})$$

将$\theta_{t+1} = \theta_t + \alpha\beta_t \nabla_\theta \pi(a_t|s_t, \theta_t)$代入得到：

$$\begin{aligned} &\pi(a_t|s_t, \boldsymbol{\theta_t}) + \left(\nabla_\theta \pi(a_t|s_t, \theta_t)\right)^T (\boldsymbol{\theta_{t+1} - \theta_t}) \\ &= \pi(a_t|s_t, \boldsymbol{\theta_t}) + \boldsymbol{\alpha\beta_t} \left(\nabla_\theta \pi(a_t|s_t, \theta_t)\right)^T \left(\boldsymbol{\nabla_\theta \pi(a_t|s_t, \theta_t)}\right) \\ &= \pi(a_t|s_t, \boldsymbol{\theta_t}) + \alpha\beta_t \|\nabla_\theta \pi(a_t|s_t, \theta_t)\|^2 \end{aligned}$$

所以正负性只与β_t有关

β_t能够很好地平衡 exploration 和 exploitation：

$$\beta_t = \frac{q_t(s_t, a_t)}{\pi(a_t|s_t, \theta_t)}$$

- β_t与$q_t(s_t, a_t)$成正比例，$q_t(s_t, a_t)$越大，则β_t越大，则$\pi(a_t|s_t, \theta_t)$增大，即一个(s_t, a_t)对的 action value 越大，策略选择它的概率越大，发挥了 exploitation
- β_t与$\pi(a_t|s_t, \theta_t)$成反比例，$\pi(a_t|s_t, \theta_t)$越小，则β_t越大，则$\pi(a_t|s_t, \theta_t)$增大，即在状态s_t选择动作a_t的概率越小，则更新后选择它的概率增大，发挥了 exploration