Deep Q-Network

Deep Q-Network

深度神经网络和Q-learning的结合。

神经网络用于近似估计Q值（发挥其万能函数拟合器的强大能力），从而实现降维的目的。

状态空间维度灾难

在一些场景下（例如强化学习应用广泛的游戏AI领域），对于Q(s,a)中的s，其可能性实在过多，例如一个游戏的截图通常都有 几百x几百 像素，假设是210×160，并且每个像素点状态有16种，那么其状态空间就是$16^{210*160}$，那么(s,a)的组合显然会是一个天文数字。所以，我们不太可能用表格的方式来明确记录每一个状态的value值。

而且，直觉上来看，大量状态的Q（s,a）值都是极为接近或相同的，比如，位置的微小的偏移下，对于动作的决策通常非常接近。

上述内容可以通过下图的游戏来帮助联想。

价值函数近似

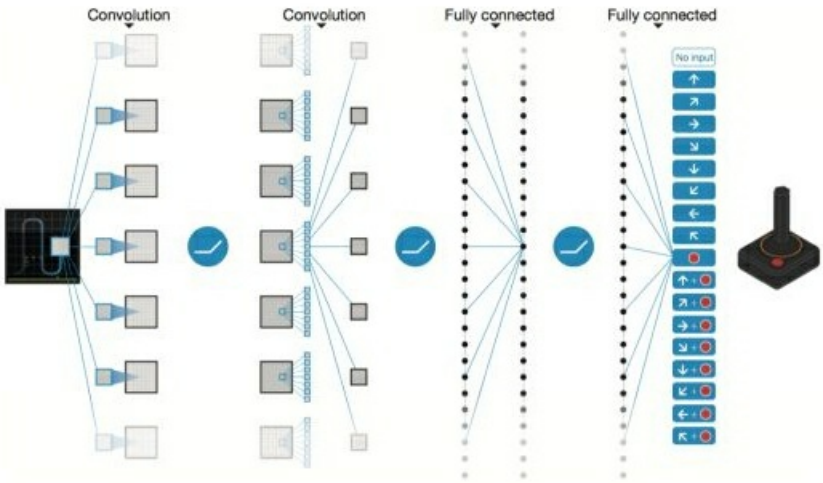
$Q(s,a) \approx f(s,a,w)$

其中，w是模型f的参数，f是对Q的近似。

不管s的维度是多大，可以通过一系列的矩阵运算降维为单值Q。

Q-network示例

Q-network就是用神经网络作为模型f，对Q函数进行近似。



以DQN为例，输入是经过处理的4个连续的84x84图像，然后经过两个卷积层，两个全连接层，最后输出包含每一个动作Q值的向量。

DQN训练

神经网络的训练需要标签和损失函数，用于反向传播调整模型f中的参数w。

对于之前介绍的Q-learning：

> $R_{t+1} + \lambda max_a Q_k(S_{t+1}, a) - Q_k(S_t, A_t)$ 可以认为是根据新采样的Reward对Q值估计误差的评估。

这样就可以不断通过采样获得标签和损失，用于训练w了，形式化的损失函数就是：

$$L(w) = E[(r + \lambda max_{a'} f(s', a', w) - f(s, a, w))^2]$$

训练算法描述

Algorithm 1 Deep Q-learning with Experience Replay

Initialize replay memory $\mathcal{D}$ to capacity N
Initialize action-value function Q with random weights
for episode $= 1, M$ **do**
  Initialise sequence $s_1 = \{x_1\}$ and preprocessed sequenced $\phi_1 = \phi(s_1)$
  for $t = 1, T$ **do**
    With probability ϵ select a random action a_t
    otherwise select $a_t = \max_a Q^*(\phi(s_t), a; \theta)$
    Execute action a_t in emulator and observe reward r_t and image x_{t+1}
    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $\mathcal{D}$
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from $\mathcal{D}$
    Set $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma \max_{a'} Q(\phi_{j+1}, a'; \theta) & \text{for non-terminal } \phi_{j+1} \end{cases}$
    Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ according to equation 3
  end for
end for