**强化学习理论学习与代码实现**

# 深度强化学习-Double DQN

目前流行的Q-learning算法会过高的估计在特定条件下的动作值。实际上，在实践中，这种过高的估计是否常见，是否会损害性能，以及是否可以预防，这些以前都不知道。于是Hado van Hasselt在2015年发表论文《Deep Reinforcement Learning with Double Q-learning》[1]回答了上述的问题，特别的，文章中指出最近的DQN算法，的确存在在玩Atari 2600时会存在严重的过高估计问题，并提出了Double Q-learning algorithm，可以很好的降低观测到的过高估计问题，而且在多个游戏上取得了更好的效果。

## 回顾

为了解决序列决策问题，可以学习每个动作最优值的估计值，大多数问题的规模比较大，从而导致无法分别学习所有状态下的所有动作，相反，可以通过参数化的方式来近似拟合值函数，在状态St执行At，返回即时奖励Rt+1，并进入下一状态St+1，标准的Q学习参数更新如下：

 （1）

$$

\boldsymbol{\theta}\_{t+1}=\boldsymbol{\theta}\_{t}+\alpha\left(Y\_{t}^{\mathrm{Q}}-Q\left(S\_{t}, A\_{t} ; \boldsymbol{\theta}\_{t}\right)\right) \nabla\_{\boldsymbol{\theta}\_{t}} Q\left(S\_{t}, A\_{t} ; \boldsymbol{\theta}\_{t}\right)

$$

其中α为标量步长，目标*YtQ*定义如下：

 （2）

$$

Y\_{t}^{\mathrm{Q}} \equiv R\_{t+1}+\gamma \max \_{a} Q\left(S\_{t+1}, a ; \boldsymbol{\theta}\_{t}\right)

$$

该更新就类似于随机梯度下降，将当前值*Q*(*St*,*At*;*θt*)朝目标值*YtQ*更新。

在上一章中我们介绍了Nature DQN，为了保证算法具有更高的稳定性，Nature DQN中引入了带有参数θ-的目标网络，该网络与在线更新网络结构相同，只是目标网络的参数更新存在延迟，即每隔C步使用在线网络更新目标网络的参数，更新的目标为：

 （3）

$$

Y\_{t}^{\mathrm{DQN}} \equiv R\_{t+1}+\gamma \max \_{a} Q\left(S\_{t+1}, a ; \boldsymbol{\theta}\_{t}^{-}\right)

$$

## 过高估计

其实早在1993年Thrun和Schwartz就已经对Q学习的过高估计进行了研究，他们认为如果动作值中包含在[-ε, ε]上均匀分布的随机误差，那么每个目标可以过高估计达 $\gamma \epsilon \frac{m-1}{m+1}$，其中m为动作数量。之后 van Hasselt在2010年就指出使用表格型方法环境中的噪声也会导致过高估计，并提出了Double Q学习。

但实际上任何形式的估计误差都可能引起上偏差，而不轮这是误差是由环境噪声、函数近似、不平稳性或其他误差源导致的。Thrun和Schwartz给出了特定设置下过高估计的上界，van Hasselt试图寻找下界。

定理1：考虑一个状态s，在该状态下所有的最优动作值函数 $Q\_{\*}(s, a)=V\_{\*}(s)$，设Qt为任意值函数估计，其为真实值函数整体上的无偏估计，即 $\sum\_{a}\left(Q\_{t}(s, a)-V\_{\*}(s)\right)=0$，但由于估计误差的存在，设 $\frac{1}{m} \sum\_{a}\left(Q\_{t}(s, a)-V\_{\*}(s)\right)^{2}=C$，C>0且m≥2，m为状态s下的动作数。在这种条件下满足 $\max \_{a} Q\_{t}(s, a) \geq V\_{\*}(s)+\sqrt{\frac{C}{m-1}}$，但是采用Double Q-learning，则绝对误差为0。

更典型的是，过高估计会随着动作数量的增加而增加，如图1所示。Q-learning的过估计确实会随着动作数量而增加，而Double Q-Learning是无偏的。

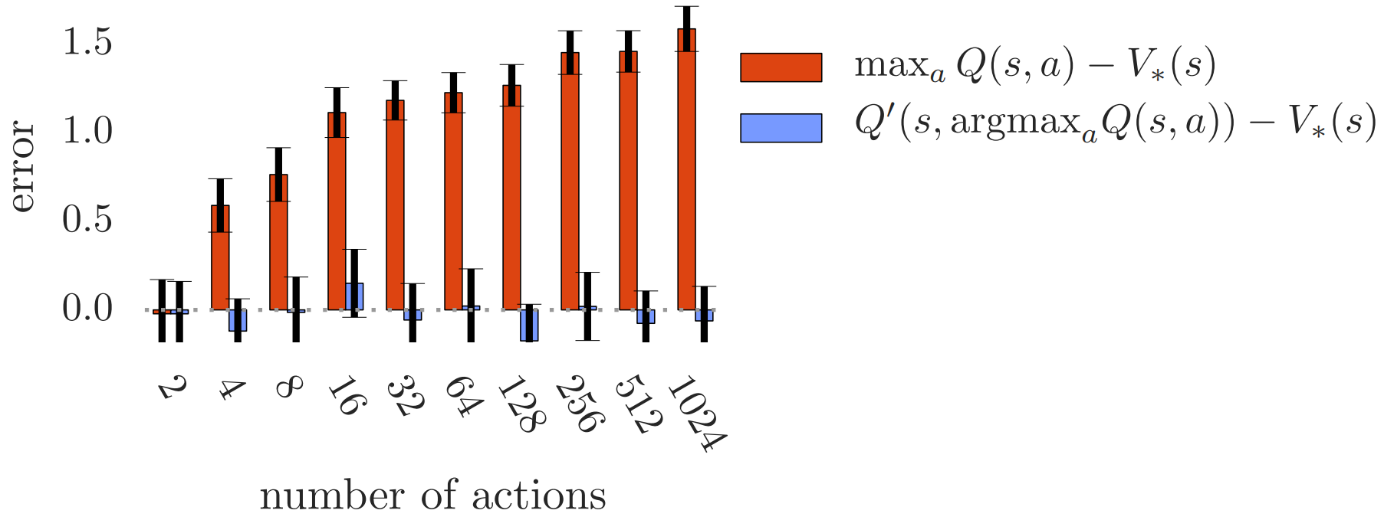


图1 橙色条当动作值函数Q(s,a)=V\*(s)+εa，且{εa}a=1m为独立标准正态随机变量时，单次Q学习更新的偏差。蓝色表示的第二组动作值Q’，是独立同分布的。所有的条都是100次重复的平均值。

## Double DQN

Double Q-learning的思想是通过将目标函数的最大值操作分解为动作选择和动作评估。尽管不是完全的解耦合，DQN框架中的目标网络提供了一个天然的对于第二个价值函数的候选，这就不需要我们再引入一个额外的网络了。因此，建议使用在线网络来评估贪心策略，然后使用目标网络来估计值函数。同时考虑Double Q-learning和DQN，可以得到算法Double DQN。它的更新和DQN一样，只不过使用下式代替YtDQN：



$$

Y\_{t}^{\text {DoubleDQN }} \equiv R\_{t+1}+\gamma Q\left(S\_{t+1}, \underset{a}{\operatorname{argmax}} Q\left(S\_{t+1}, a ; \boldsymbol{\theta}\_{t}\right), \boldsymbol{\theta}\_{t}^{-}\right)

$$

和Double Q学习相比，用目标网络θt-代替第二个网络θt’的权重，用来评估当前贪婪策略。目标网络的更新跟DQN里一样，都是周期性地复制在线网络的参数。

**References:**

[1] H. van HasseltA. Guez and D. Silver, "Deep Reinforcement Learning with Double Q-learning," *Computer Science*, 2015.