深度 Q 网络及其改进算法介绍

合肥工業大學

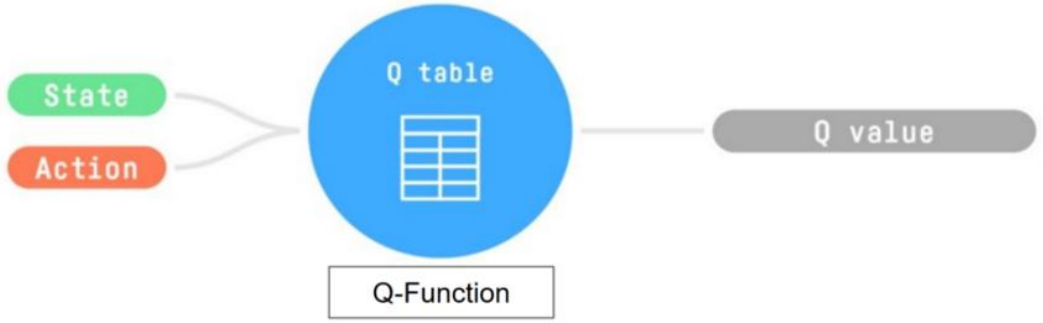
汪悦

2023.6.19

研究进展

■ Q 学习

Q 学习算法是一种离策略时序差分（Off-Policy Temporal Difference）算法，训练状态-动作价值函数（Q 函数），利用经验累积，学习到最优策略。



Q表

	动作1	动作2	动作3	动作4
状态1				
状态2				
状态3				
状态4				
状态5				

算法步骤：

（1）初始化 Q 表。

（2）使用贪心策略选择动作。

（3）执行动作 At，在 Q 表中搜索相应的值，得到奖励 Rt+1 和状态 St+1 。

（4）更新 Q 表，不断训练，不断探索环境，直至得到最优 Q 表和最优 Q 函数。

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \partial[R_{t+1} + \gamma maxQ(S_{t+1}, a) - Q(S_t, A_t)]$$ ∂：学习率

（5）根据最优 Q 函数可以得到最优策略，知道每一个状态采取什么是最好的行动。

$$\pi^*(s) = argmaxQ^*(s, a)$$

研究背景

■ Q 学习

Q 学习使用表格来存储每个状态 s 下采取动作 a 获得的奖励，即状态-动作值函数 Q(s, a)。

但这种方法在状态量巨大甚至是连续的任务中，会遇到维度灾难问题，无法使用表格对 Q 函数进行存储。

解决办法：【使用价值函数的近似表示】

价值函数近似价值函数近似能够解决维度灾难问题，通过函数近似来估计实际的价值函数，把从已知的状态学到的函数通用化推广至那些未碰到的状态中，使用蒙特卡洛学习或时序差分学习来更新函数参数。【深度 Q 网络即采用价值数近似的表示方法】

■ 深度 Q 网络

深度 Q 网络（DQN）是指基于深度学习的 Q 学习算法，将 Q 学习与深度学习结合，建立策略网络和目标网络，采用了经验回放的网络训练方法，从历史数据中随机采样，使用深度神经网络近似动作价值函数。

方法介绍

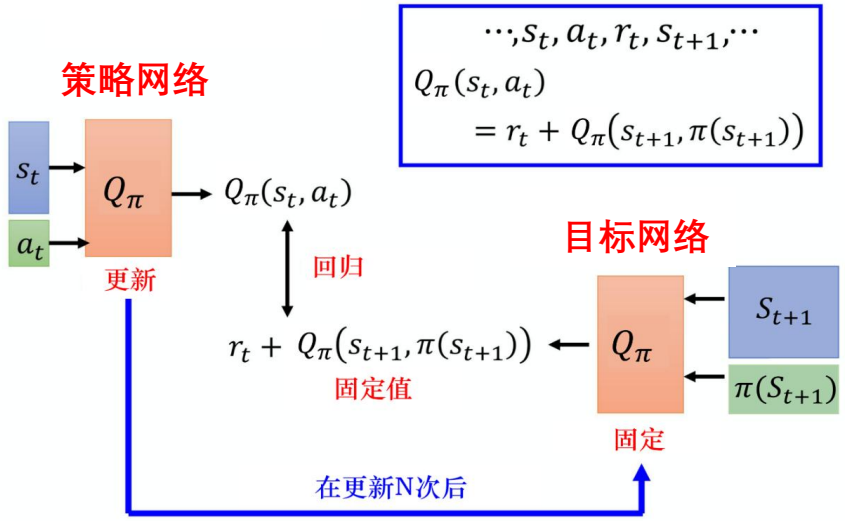
■ 深度 Q 网络

深度 Q 网络（DQN）主要使用了神经网络、目标网络、经验回放等技巧。

➢ 目标网络

【设计两个相同结构的 Q 网络：策略网络、目标网络】

在训练的时候，将右边的目标网络固定。使用策略网络 Q 用来选择动作，更新模型参数。多次更新策略网络，每更新c 次后，将更新后的策略网络参数复制到目标网络（目标网络延时更新），改变目标值 。



优势：

✓ 用两个 Q 网络来减少目标 Q 值计算和更新 Q 网络参数的依赖关系。

✓ 减少目标 Q 值和当前 Q 值相关性，避免目标值一直变动影响性能。

方法介绍

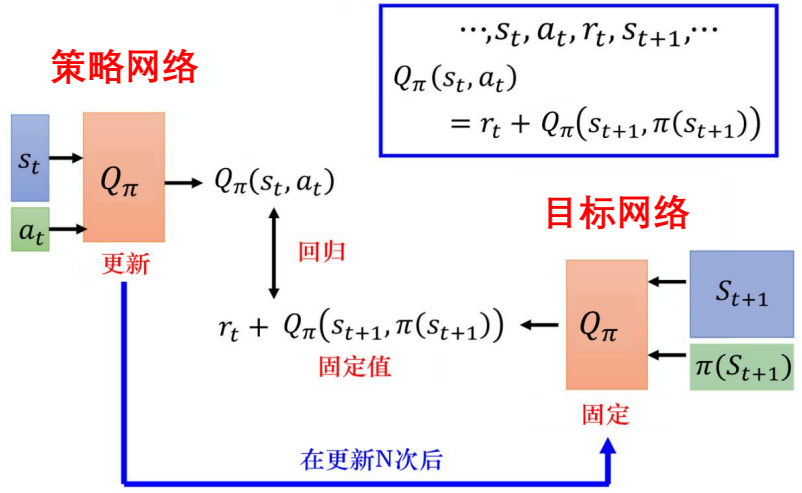
■ 深度 Q 网络

深度 Q 网络（DQN）主要使用了神经网络、目标网络、经验回放等技巧。

➢ 经验回放

【使用经验回放更新动作价值函数】

构建回放缓冲区（replay buffer），又称为回放内存（replay memory）。经验回放将每次和环境交互得到的奖励与状态更新作为经验保存起来。回放缓冲区只有在它装满的时候，才会把旧的数据丢掉。采样时，从缓冲区随机采取批量数据，用于后面的目标 Q 值的更新。



好处：

✓ 放入回放缓冲区可以减少与环境交互的次数（最花时间的步骤）

✓ 一个批量的数据越多样越好（如果批量里面都是一样数据，训练性能会比较差）

方法介绍

■ 深度 Q 网络

算法流程

算法输入：迭代轮数T，状态特征维度n，动作集A，步长α，衰减因子γ，探索率ϵ，当前Q网络Q，目标Q网络Q'，批量梯度下降的样本数m，目标Q网络参数更新频率C。

输出：Q网络参数

1. 随机初始化所有的状态和动作对应的价值Q. 随机初始化当前Q网络的所有参数w,初始化目标Q网络Q'的参数$w' = w$。清空经验回放的集合D。

2. for i from 1 to T，进行迭代。

a) 初始化S为当前状态序列的第一个状态，拿到其特征向量$\phi(S)$

b) 在Q网络中使用$\phi(S)$作为输入，得到Q网络的所有动作对应的Q值输出。用$\epsilon-$贪婪法在当前Q值输出中选择对应的动作A **选择动作**

c) 在状态S执行当前动作A,得到新状态S'对应的特征向量$\phi(S')$和奖励R$,是否终止状态is_end

d) 将$\{\phi(S), A, R, \phi(S'), is_end\}$这个五元组存入经验回放集合$D$ **经验回放**

e) $S = S'$

f) 从经验回放集合D中采样m个样本$\{\phi(S_j), A_j, R_j, \phi(S'_j), is_end_j\}, j = 1, 2., , , m$，计算当前目标Q值$y_j$：**采取批量数据**

$$y_j = \begin{cases} R_j & is\_end_j\ is\ true \\ R_j + \gamma \max_{a'} Q'(\phi(S'_j), A'_j, w') & is\_end_j\ is\ false \end{cases}$$

g) 使用均方差损失函数$\frac{1}{m}\sum_{j=1}^{m}(y_j - Q(\phi(S_j), A_j, w))^2$，通过神经网络的梯度反向传播来更新Q网络的所有参数w **更新策略网络**

h) 如果i%C=1,则更新目标Q网络参数$w' = w$ **更新目标网络**

i) 如果S'是终止状态，当前轮迭代完毕，否则转到步骤b)

方法介绍

■ 深度 Q 网络

【 DQN 存在Q 值“高估”问题】

➢ 自举导致偏差传播

用自己的预测的最大结果作为目标，更新自己。智能体总是会选择 Q 值被高估的动作，总是会选奖励被高估动作的 Q 加上即时奖励作为目标。当预测的结果偏大，更新参数后会导致 DQN 输出的预测结果进一步偏大，形成正反馈。

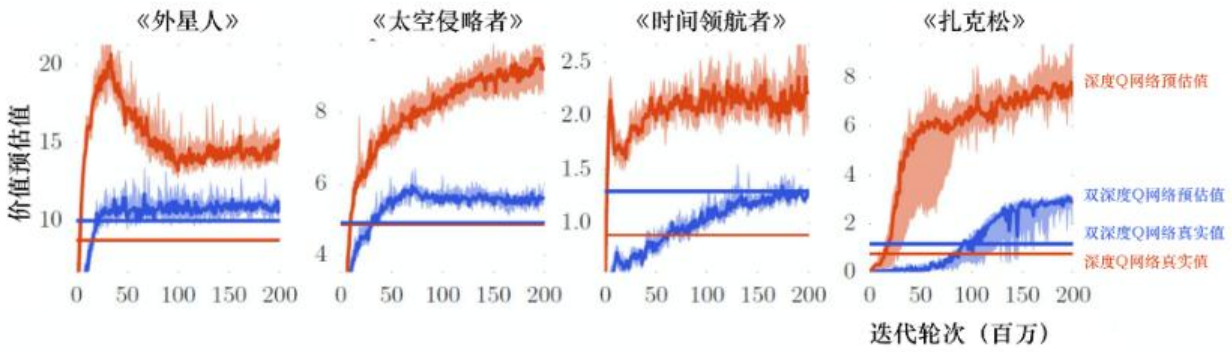


图 7.1 被高估的Q值

- DQN高估自己会得到的奖励，DDQN的预估值与真实值比较接近。
- DDQN得出真实Q值比DQN高，说明学习出来的策略比DQN的强。

方法介绍

■ 深度 Q 网络

【 DQN 存在Q 值“高估”问题】

➢ 非均匀高估

DQN 在训练过程中，从 replay buffer 中随机选取样本用于更新参数，对于不同的 Q(s, a) 所更新的次数与程度是不同的，因此 DQN 是非均匀的高估问题。

➢ 如何解决“高估”问题?

改进：双深度 Q 网络（DDQN） 在 DQN 基础上，解耦目标 Q 值动作选择和目标 Q 值计算这两步。

DQN 使用目标网络 Q’ 来找使 Q 值最大的动作。

DDQN 使用策略网络 Q 来找使 Q 值最大的动作。

方法介绍

■ 双深度 Q 网络

【DDQN 与 DQN 区别在于目标 Q 值计算方式不同】

➢ DQN 目标 Q 网络的计算：

$$y_j = R_j + \gamma \max_{a'} Q'(\phi(S'_j), A'_j, w')$$

➢ DDQN 不再直接从目标 Q 网络找各动作的最大 Q 值，而是先在当前 Q 网络找到最大 Q 值对应的动作：

$$a^{max}(S'_j, w) = arg\max_{a'} Q\left(\phi(S'_j), a, w\right)$$

再利用这个动作$a^{max}(S'_j, w)$在目标网络中计算目标Q值：

$$y_j = R_j + \gamma Q'\left(\phi(S'_j), a^{max}(S'_j, w), w'\right)$$

综合即得：

$$y_j = R_j + \gamma Q'\left(\phi(S'_j), arg\max_{a'} Q\left(\phi(S'_j), a, w\right), w'\right)$$

方法介绍

■ 双深度 Q 网络

算法流程

算法输入：迭代轮数T，状态特征维度n，动作集A，步长α，衰减因子γ，探索率ϵ，当前Q网络Q，目标Q网络Q'，批量梯度下降的样本数m,目标Q网络参数更新频率C。

输出：Q网络参数

1. 随机初始化所有的状态和动作对应的价值Q. 随机初始化当前Q网络的所有参数w,初始化目标Q网络Q'的参数$w' = w$。清空经验回放的集合D。

2. for i from 1 to T，进行迭代。

a) 初始化S为当前状态序列的第一个状态，拿到其特征向量$\phi(S)$

b) 在Q网络中使用$\phi(S)$作为输入，得到Q网络的所有动作对应的Q值输出。用$\epsilon-$贪婪法在当前Q值输出中选择对应的动作A **选择动作**

c) 在状态S执行当前动作A,得到新状态S'对应的特征向量$\phi(S')$和奖励R$,是否终止状态is_end

d) 将$\{\phi(S), A, R, \phi(S'), is_end\}$这个五元组存入经验回放集合$D$ **经验回放**

e) $S = S'$

f) 从经验回放集合D中采样m个样本$\{\phi(S_j), A_j, R_j, \phi(S'_j), is_end_j\}, j = 1, 2., , , m$，计算当前目标Q值$y_j$： **采取批量数据**

$$y_j = \begin{cases} R_j & is\_end_j\ is\ true \\ R_j + \gamma Q'(\phi(S'_j), \arg\max_{a'} Q(\phi(S'_j), a, w), w') & is\_end_j\ is\ false \end{cases}$$

g) 使用均方差损失函数$\frac{1}{m}\sum_{j=1}^{m}(y_j - Q(\phi(S_j), A_j, w))^2$，通过神经网络的梯度反向传播来更新Q网络的所有参数w **更新策略网络**

h) 如果i%C=1,则更新目标Q网络参数$w' = w$ **更新目标网络**

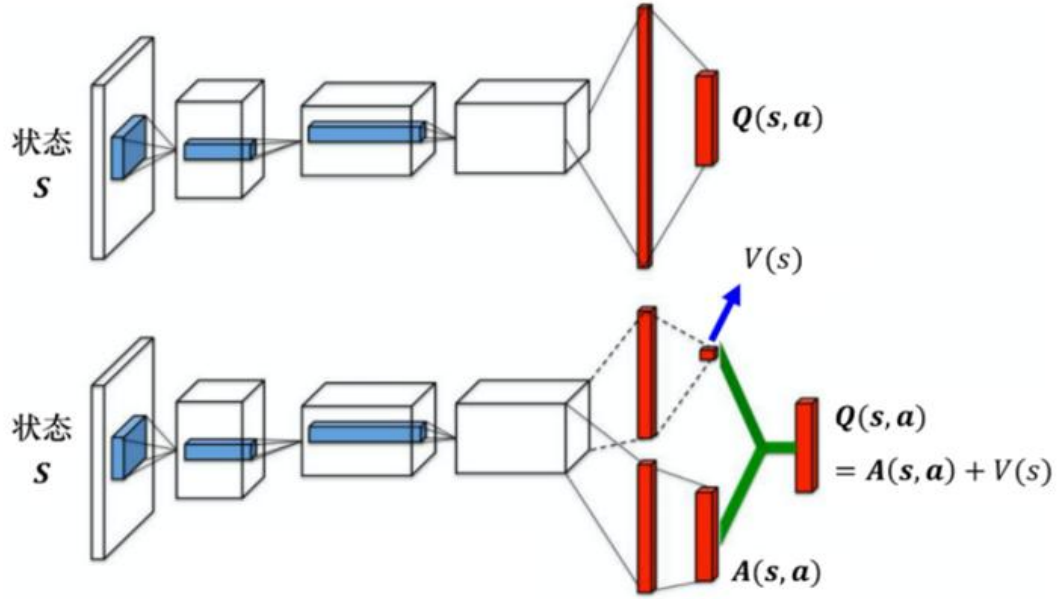
i) 如果S'是终止状态，当前轮迭代完毕，否则转到步骤b)

方法介绍

- **竞争深度 Q 网络**

  【改变 Q 网络架构】

- **DQN 直接输出 Q 值。**
- **竞争深度 Q 网络（Dueling DQN）不直接输出 Q 值，分为两个输出路径，分别输出标量 V(s) 和向量 A(s, a)，将其加起来即为 Q(s, a)。**



$$Q(s,a)=V(s)+A(s,a)$$

图 7.3 竞争深度Q网络的网络结构

方法介绍

■ 竞争深度 Q 网络

【竞争深度 Q 网络能够有效利用数据】

例如，假设只有 4 个不同的状态、 3 个不同的动作，Q(s,a) 可以看成一个表格（右图）。

当目标是将 Q 表格中 3 变成 4 、-1 变成 0 ，可以不用修改 A(s, a) 的值，只改变 V(s) 的值，同步将 -2 变成 -1。

当在某一状态下，只采样到两个动作，没采样到第三个动作，但也可以更改第三个动作的 Q 值。

竞争深度 Q 网络不需要把所有的状态-动作对都采样，使用比较高效的方式估计 Q 值。

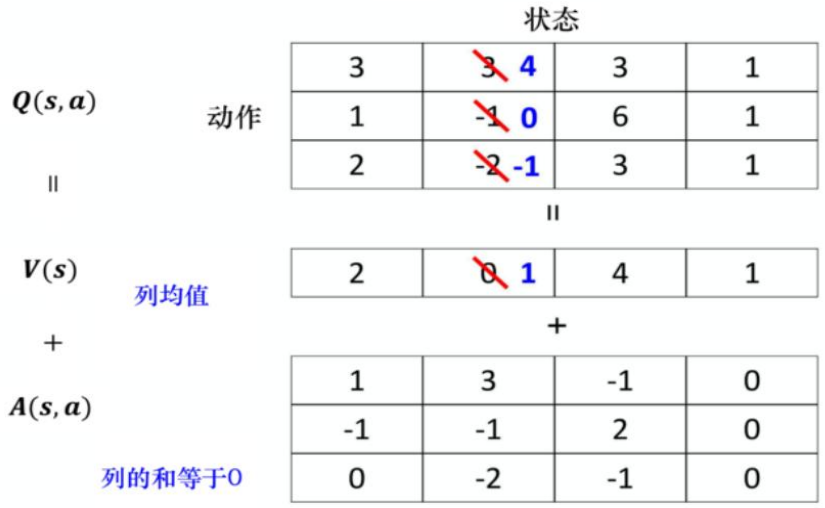
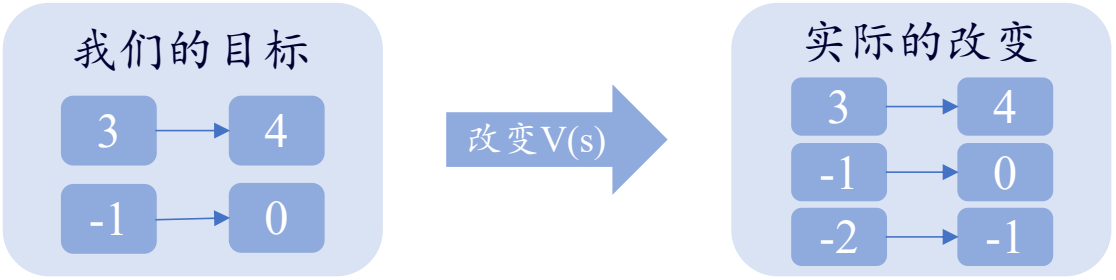


图 7.4 竞争深度Q网络训练



方法介绍

■ 优先级经验回放

【经验回放优先级】

- **原来：在采样数据训练 Q 网络的时候，均匀地从回放缓冲区里面采样数据。**
- **现在：优先级经验回放（PER）赋予优先权，在训练的时候多考虑那些不好训练的数据（时序差分误差特别大，即网络的输出与目标之间差距特别大的数据），使它们有比较大的概率被采到。**

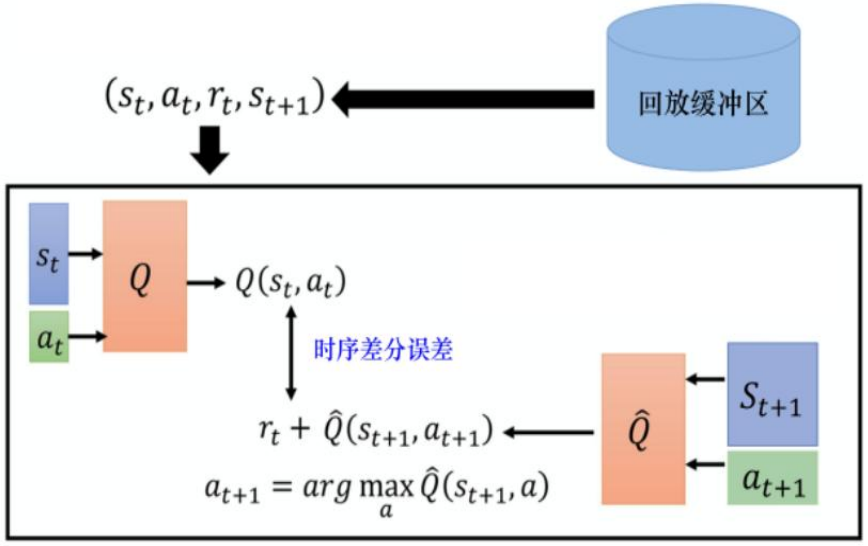


图 7.6 优先级经验回放

实际上在做 PER 的时候，我们不仅更改采样的过程，还会因为更改了采样的过程，而更改更新参数的方法。所以 PER 不仅改变了采样数据的分布，还改变了训练过程。

方法介绍

■ 噪声网络

【参数的空间加噪声】

在 Q 函数网络的每一个参数加上一个高斯噪声，将原来的【 Q 函数】Q 变为【噪声Q函数】 $\tilde{Q}$ 。

使用噪声网络执行的动作：

$$a = \underset{a}{argmax}\, \tilde{Q}(s, a)$$

噪声网络方法

- OpenAI：每一个参数、每一个权重都加一个高斯噪声。
- DeepMind：噪声由一组参数控制，网络自己决定噪声的大小。

➢ **要点：**

- 每一回合的参数噪声固定。
- 在新的一回合，环境交互前重新采样噪声。

➢ **改进之处：**

- -探索等方法遇到相同或类似的状态，有时执行Q函数，有时随机执行动作。
- 添加噪声网络，遇到相同或类似的状态，将采取相同的动作。【依赖状态的探索】

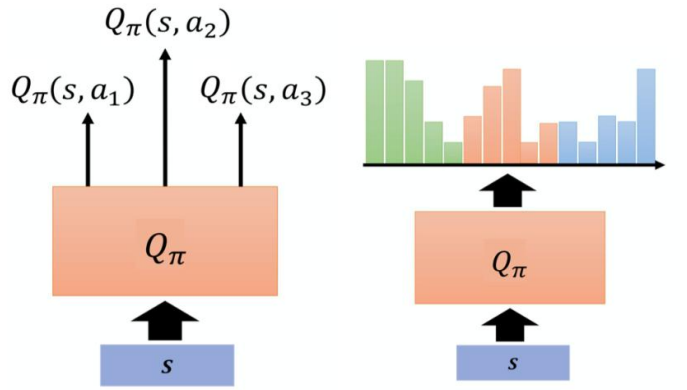
方法介绍

■ 分布式 Q 网络

【对奖励的分布建模】

➢ 原先：Q 函数是累积奖励的期望值。环境具有随机性，系统在某一个状态采取某一个动作，在回合结束的时候统计所有的奖励，实际会得到一个分布，计算该分布的平均值即得累积奖励的期望。

但是不同的分布可以有同样的平均值，如图7.8。假设我们只用 Q 值的期望来代表整个奖励，可能会丢失一些信息，无法对奖励的分布进行建模。



(a) 一个拥有 3 个输出的网络　(b) 一个拥有 15 个输出的网络（每一个动作对应 5 个长条）

图 7.12 分布式Q函数

➢ 在状态 s 下，我们采取a_1、a_2、a_3这3 个动作，输出 3 个值，分别代表 3 个动作的 Q 值。【分布的期望值】

➢ 分布式Q函数（Distributional DQN）就是【直接输出分布】。

➢ 假设奖励空间可以拆成 n 个长条，Q 函数的输出就是要预测在某一个状态采取某一个动作得到的奖励，落在某一个长条里面的概率。

选平均值最大的动作执行

分布方差过大——规避风险

方法介绍

■ 彩虹

【方法整合，能力互补】

➢ 假设每个方法有一种自己的颜色，将所有的颜色组合，就变成了“彩虹”。

➢ 这些方法之间是没有冲突的，把全部方法都用上就变成了七彩的方法，即彩虹方法，性能优于单一算法。

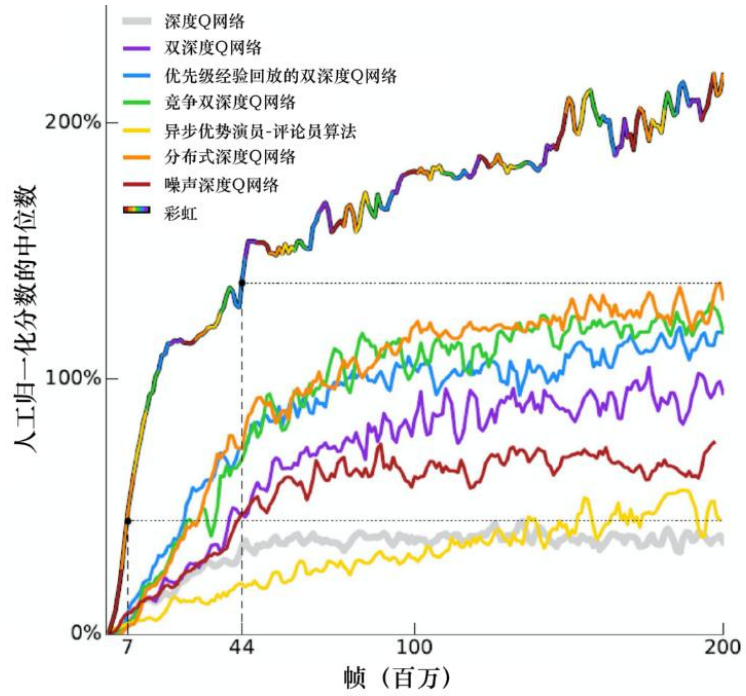


图 7.10 彩虹方法

横轴：训练过程的帧数。

纵轴：玩十几个雅达利小游戏的平均分数和。

性能对比：

彩虹 > 分布式深度 Q 网络 > 竞争双深度 Q 网络 > 优先级经验回放的双深度 Q 网络 > 双深度 Q 网络 > 噪声深度 Q 网络 > 异步优势演员-评论员算法 > 深度 Q 网络

方法介绍

- **研究进展**
  - **找到合适的彩虹算法，并跑通算法代码。**
  - **参考经典Gym 强化学习环境代码，构建所研究的 MDP 模型的强化学习环境。**
  - **查找相关论文，调整文章针对的研究场景、研究主体以及研究立意。**

Q&A

合肥工業大學