计算机科学
COMPUTER SCIENCE

基于softmax的加权Double Q-Learning算法

钟雨昂, 袁伟伟, 关东海

引用本文

钟雨昂, 袁伟伟, 关东海. 基于softmax的加权Double Q-Learning算法[J]. 计算机科学, 2024, 51(6A): 230600235-5.

ZHONG Yuang, YUAN Weiwei, GUAN Donghai. Weighted Double Q-Learning Algorithm Based on Softmax [J]. Computer Science, 2024, 51(6A): 230600235-5.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.230600235

基于 softmax 的加权 Double Q-Learning 算法

钟雨昂　袁伟伟　关东海
南京航空航天大学计算机科学与技术学院　南京 211106
(zhongyuang666@163.com)

摘　要　强化学习作为机器学习的一个分支,用于描述和解决智能体在与环境的交互过程中,通过学习策略以达成回报最大化的问题。Q-Learning 作为无模型强化学习的经典方法,存在过估计引起的最大化偏差问题,并且在环境中奖励存在噪声时表现不佳。Double Q-Learning(DQL)的出现解决了过估计问题,但同时造成了低估问题。为解决以上算法的高低估问题,提出了基于 softmax 的加权 Q-Learning 算法,并将其与 DQL 相结合,提出了一种新的基于 softmax 的加权 Double Q-Learning 算法(WDQL-Softmax)。该算法基于加权双估计器的构造,对样本期望值进行 softmax 操作得到权重,使用权重估计动作价值,有效平衡对动作价值的高估和低估问题,使估计值更加接近理论值。实验结果表明,在离散动作空间中,相比于 Q-Learning 算法、DQL 算法和 WDQL 算法,WDQL-Softmax 算法的收敛速度更快且估计值与理论值的误差更小。

关键词:强化学习;Q-Learning;Double Q-Learning;Softmax
中图分类号　TP181

Weighted Double Q-Learning Algorithm Based on Softmax

ZHONG Yuang, YUAN Weiwei and GUAN Donghai
College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China

Abstract　As a branch of machine learning, einforcement learning is used to describe and solve the problem that agents maximize returns through learning strategies in the process of interaction with the environment. Q-Learning, as a classical model free reinforcement learning method, has the problem of maximizing the bias caused by overestimation, and performs poorly when there is noise in the environment. The emergence of double Q-Learning(DQL) solves the problem of overestimation, but at the same time causes the problem of underestimation. To solve the problem of high and low estimation in the above algorithms, weighted Q-Learning algorithm based on softmax is proposed. And combined with DQL, a new weighted double Q-Learning algorithm based on softmax(WDQL-Softmax) is proposed. This algorithm is based on the construction of weighted dual estimators, which perform softmax operations on the expected values of the samples to obtain weights. The weights are used to estimate the action value, effectively balancing the problem of overestimation and underestimation of the action value, making the estimated value closer to the theoretical value. Experimental results show that in the discrete action space, compared with Q-Learning algorithm, double Q-Learning algorithm and weighted double Q-learning algorithm, weighted double q-learning algorithm based on softmax has faster convergence rate and smaller error between the estimated value and the theoretical value.

Keywords　Reinforcement learning, Q-Learning, Double Q-Learning, Softmax

1　引言

强化学习用于描述和解决智能体在与环境的交互过程中,通过学习策略以达成回报最大化问题[1]。强化学习涉及到智能体在环境中进行决策,根据行动的结果获得奖励或惩罚,从而逐步学习如何在不同情境下采取最优的行动,以获得最大的累积奖励。

强化学习中的环境通常被建模为一个马尔可夫决策过程(MDP)[2]。MDP 是一种数学框架,用于描述序列决策问题中的状态转移和奖励信号。在 MDP 中,智能体在环境中的状态转移和获得奖励的过程是基于当前状态和执行的行动,且与之前的状态和行动无关。强化学习在游戏[3]、机器人控制[4]、自动驾驶[5]、金融交易[6]等领域都有广泛的应用。

强化学习根据是否需要模型来学习,可以分为有模型强化学习和无模型强化学习,无模型强化学习不需要对环境进行假设和建模,而是直接学习从状态到动作的映射关系,从而指导决策,因此,灵活性高且适应性强的无模型强化学习受到更多的关注。Q-Learning 是一种非常经典的无模型强化学习算法,它由 Watkins 等[7]于 1989 年提出,通过估计最优状态动作价值函数来实现。直到现在 Q-Learning 仍然是无模型强化学习的基础,很多算法都是基于它的改进和衍生。

但是 Q-Learning 存在多个缺陷。第一,它在估计最优状态动作价值函数时,会由于过估计产生最大化偏差[8]。过估计指的是,对一系列数先求最大值再求平均,所得值通常比先求平均再求最大值要大。Q-Learning 算法在估计样本期望值的最大值时,使用样本最大值的期望作为估计值,导致最终结果有很严重的高估,产生最大化偏差,算法无法快速收敛。第二,过估计产生的最大化偏差会形成误差累积,如果过度估计了某个动作的价值,会使得算法在接下来的许多步中选择该动作,进而导致整个学习过程中的行动偏差,可能会错过最优解或者陷入次优解。第三,当奖励是噪声或是满足某个分布的随机值时,Q-Learning 表现很差,奖励的大范围浮动导致

通信作者:袁伟伟(yuanweiwei@nuaa.edu.cn)

Q-Learning 算法几乎无法收敛。

本文的主要贡献在于提出了基于 softmax 的加权 Q-Learning 算法，该算法对每个动作的样本期望值进行 softmax 操作，估计每个动作具有最大期望值的概率来获得权重，以此估计动作的最大期望值。并将其与 Double Q-Learning 相结合，提出了一种基于 softmax 的加权 Double Q-Learning 算法(Weighted Double Q-Learning Based on Softmax，WDQL-Softmax)。该算法基于加权双估计器的构造，可以有效平衡 Q 值的高估和低估问题，在一些实验环境下取得了良好的效果。

2 相关工作

2.1 马尔可夫决策过程

马尔可夫决策过程提供了一种形式化的框架，用于描述强化学习问题中的环境。强化学习算法通过将问题建模为 MDP 来理解和解决决策问题。马尔可夫决策过程由五元组 (S,A,P,R,γ) 组成，五元组中 S 表示状态空间；A 表示动作空间；P 表示在某一状态 S_i 下，采取动作 A_i 转移到 S_{i+1} 转态的概率；R 表示在当前状态 S 下，采取动作 A 获得的奖励；γ 表示折扣因子，对未来的奖励打折扣，代表了未来的奖励在当下的价值。整个决策过程的累积回报 G_t 可以定义为式(1)：

$$G_t=R_0+\gamma R_1+\cdots=\sum_{i=0}^{t}\gamma^i R_i \tag{1}$$

强化学习的目标是让智能体与环境交互，通过价值学习或策略学习，使得累计回报的期望最大化。

2.2 相关算法

在 Q-Learning 被提出之后，陆续有很多对其的改进算法被提出。针对收敛速度问题，Azar 等提出了 Speedy Q-Learning(SQL)算法[9]，该算法通过增大学习率的方式加快收敛速度；Lee 等在保证渐进收敛性的基础上，校正 Q-Learning 中的最大算子偏差，提出偏差校正 Q-Learning 算法[8]，该算法能够缓解过估计问题；Hasselt 使用双估计器，构造两个 Q 表 Q^A 和 Q^B，使用不同的 Q 表分别处理最优动作的选择和取值，该方法被称为 Double Q-learning(DQL)[10]。D'Eramo 等认为在样本数较大的情况下，可以使用正态分布近似样本平均值的分布，通过高斯近似估计最大期望值，降低最大化偏差[11]；Zhang 等将 Double-Q-Learning 和 Q-Learning 以加权的方式结合，提出了 Weighted Double Q-Learning(WDQL)算法[12]，其在各种 MDP 问题上表现优秀；Ren 等发现 DQL 中存在多个非最优不动点，它们会导致低估偏差，因而提出了 Doubly Bound Q-Learning(DBQL)算法[13]，利用动态规划排除这些非最优不动点，避免陷入次优解。Wang 等将迁移学习与强化学习结合，将旧任务中已经学习好的 Q 函数迁移到新任务中，提出目标迁移 Q-Learning(TTQL)[14]，TTQL 具有更快的收敛速度。

3 最大期望值的估计

本章主要讨论随机变量的最大期望值估计问题，有一个包含 M 个随机变量的随机变量集合 $X=\{X_1,X_2,X_3,\cdots,X_M\}$，定义 $f_i(x)$ 为随机变量 X_i 的概率密度函数，$\max_i E\{X_i\}$ 表示 X_i 的最大期望值，可以表示为：

$$\max_i E\{X_i\}=\max_i\int_{-\infty}^{+\infty}x f_i(x)\mathrm{d}x \tag{2}$$

但由于 X_i 的分布是未知的，无法确定 $f_i(x)$，因此无法通过上述公式求得，只能对最大期望值进行估计。接下来首先描述两种现有的方法，即单估计器和双估计器，然后介绍新的 softmax 加权双估计器方法。

3.1 单估计器

对随机变量集合 X 进行采样，定义 $S=\bigcup_{i=1}^{M}s_i$，s_i 表示随机变量 X_i 对应的采样样本集合，定义估计器 $\mu=\{\mu_1,\mu_2,\cdots,\mu_M\}$，$\mu_i$ 表示对随机变量 X_i 的估计值，此时有 $\max_i E\{X_i\}=\max_i E\{\mu_i\}\approx\max_i\mu_i(S)$。假设 s_i 中的样本是独立同分布的，此时期望值的无偏估计可以通过计算每个变量的样本平均值来获得，即 $\max_i\mu_i(S)=\frac{1}{|S_i|}\sum_{s\in S_i}s$。

$$E[\max(\mu_i)]\geqslant\max E[\mu_i] \tag{3}$$

Q-Learning 使用这种方法通过最大化下一个状态的估计动作值来近似下一个状态的值，但先计算动作的最大值再求期望，根据式(3)可知，这种方式得到的估计值比实际值大，也就是过估计，会导致最大化偏差。并且过估计导致的误差会累积，导致整个学习过程中的行动偏差，进而可能会错过最优解或者陷入次优解。

3.2 双估计器

为了避免单估计器产生的最大化偏差，Hasselt 提出了双估计器方法[11]。双估计器的思想是对每个变量使用两个估计器，并将估计器的选择与其值解耦。定义两个估计器 μ^U 和 μ^V，$\mu^U=\{\mu_1^U,\mu_2^U,\cdots,\mu_N^U\}$，$\mu^V=\{\mu_1^V,\mu_2^V,\cdots,\mu_N^V\}$。将采样集合 S 划分为互不相交的两个子集 S^U 和 S^V，满足以下等式，如式(4)所示：

$$\begin{cases}S_i=S_i^U\cup S_i^V\\ \varnothing=S_i^U\cap S_i^V\\ \mu_i^U(S)=\dfrac{1}{|S_i^U|}\displaystyle\sum_{s\in S_i^U}s\\ \mu_i^V(S)=\dfrac{1}{|S_i^V|}\displaystyle\sum_{s\in S_i^V}s\end{cases} \tag{4}$$

其中，$\mu_i^U(S)$ 表示随机变量 X_i 在估计器 μ^U 中采样的样本集合 S_i^U 的期望结果，$\mu_i^V(S)$ 同理。两个估计器中，一个用于寻找动作最大值 a^*，另一个基于 a^* 估计动作价值。$E\{\mu_i^U(S)\}$ 和 $E\{\mu_i^V(S)\}$ 是 $E\{X_{a^*}\}$ 的无偏估计，但因为 $E\{X_{a^*}\}\leqslant\max_i E\{X_i\}$，所以双估计器估计存在负偏差。

3.3 softmax 加权双估计器

定义两个估计器 $\mu^U=\{\mu_1^U,\mu_2^U,\cdots,\mu_N^U\}$ 和 $\mu^V=\{\mu_1^V,\mu_2^V,\cdots,\mu_N^V\}$，为平衡两者的权重，根据专家意见加权评估法确定两个估计器的权重，如式(5)所示：

$$\mu^{\mathrm{SWDE}}(S)=\delta\mu^U(S)+(1-\delta)\mu^V(S) \tag{5}$$

其中，参数 δ 是一个权重因子，$\delta\in[0,1]$。在极端情况下，当 $\delta=0$ 时，该估计器相当于单估计器；当 $\delta=1$ 时，该估计器相当于双估计器；当 $\delta\in(0,1)$ 时，估计器会融合单估计器和双估计器的特点，理想情况下可以平衡单估计器的高估和双估计器的低估。

现在需要确定 δ 的取值，为了体现 Q-Learning 在随机奖励环境中的缺陷，实验设置的奖励往往都是随机的，因此我们将 δ 的值与随机奖励的随机范围挂钩，如式(6)与式(7)所示：

$$\delta=\frac{c}{b+c} \tag{6}$$

$$b=|R_{\max}-R_{\min}| \tag{7}$$

其中，b 等于随机奖励最大值与最小值的差值的绝对值，c 是一个常数，$c\in[0,+\infty)$。因为 b 恒大于 0，所以 $\delta\in(0,1)$。

4 基于 softmax 的加权 DQL 算法

WDQL-Softmax 算法的流程如算法 1 所示。考虑到

Q-Learning 中的最大值偏差主要来源于 $Q(s',a^*)$，a^* 仅由 Q 表中的最大动作值决定，没有考虑次大动作值等其他动作值的影响。基于此，WDQL-Softmax 算法使用 softmax 函数对 Q 表中状态 s 下的所有动作值进行映射，并将映射后的值 w_i 作为动作 a_i 的概率，加权求和得到值 M。在算法 1 的第 12 行使用值 M 替代 $Q(s',a^*)$，一定程度上减小了高估问题，M 一定小于等于 $Q(s',a^*)$，即满足式(8)：

$$M=\sum w_i Q(s',a_i)\leqslant Q(s',a^*) \tag{8}$$

在计算 δ 时，不同于 DQL 算法，WDQL-Softmax 算法引入参数 β，如算法 1 第 6 行所示。c 为常数，根据实验测试，c 一般取 10 或 15 实验效果最好，b 为衡量随机奖励分布的参数，取值为随机奖励差值的最大值。WDQL-Softmax 算法在使用 softmax 处理动作值的同时融合了 DQL 中的双估计器方法，在减小最大值偏差的同时加快了收敛速度。

算法 1 WDQL-Softmax 算法

1. 初始化 Q^A 表、Q^B 表、状态 s
2. while(s′≠Terminal)：
3. 基于某种策略(如：ε-greedy 算法)从 Q^A、Q^B 表中状态 s 下选择动作 a
4. 执行动作 a，观察奖励 r、状态 s′
5. 随机更新 Q^A 表或 Q^B 表
6. β=c/(b+c)
7. If 更新 Q^A：
8. for $a_i\in a$：
9. $w_i=\mathrm{softmax}(Q^A(s',a_i))$
10. $M=\sum w_i * Q^A(s',a_i)$
11. $a^*=\mathrm{argmax}\, Q^A(s',a)$
12. $\delta=r+\gamma(\beta M+(1-\beta)Q^B(s',a^*))$
13. $Q^A(s,a)=Q^A(s,a)+\alpha(\delta-Q^A(s,a))$
14. If 更新 Q^B：
15. for $a_i\in a$：
16. $w_i=\mathrm{softmax}(Q^B(s',a_i))$
17. $M=\sum w_i * Q^B(s',a_i)$
18. $a^*=\mathrm{argmax}\, Q^B(s',a)$
19. $\delta=r+\gamma(\beta M+(1-\beta)Q^A(s',a^*))$
20. $Q^B(s,a)=Q^B(s,a)+\alpha(\delta-Q^B(s,a))$
21. s=s′

其中，α 代表学习率，决定更新的步长，取值范围为(0，1]；γ 代表折现因子，表示对未来奖励的重视程度，取值范围为(0，1)；a^* 表示状态 s' 下 Q 值最大的动作。

5 实验与结果

本章将本文提出的 WDQL-Softmax 算法与 Q-Learning 及其变体作比较，主要关注算法的收敛速度和最大值偏差。实验环境选择 3 个具有代表性的任务。首先，最大化偏差的例子表明，WDQL-Softmax 算法比其他算法可以消除更多的偏差。其次，部分随机奖励和完全随机奖励的网格世界实验展现了 WDQL-Softmax 算法优秀的收敛速度和收敛效果，能有效平衡高估与低估的偏差，并显示其对高奖励可变性的鲁棒性。

5.1 最大化偏差示例

这个简单的 MDP 环境最早由 Sutton 和 Barto 提出[15]，如图 1 所示。A 和 B 分别表示两种状态，Terminal 表示终止状态，a_i 表示动作，R 表示执行相应动作获得的奖励。智能体的起始状态总为 A，选择 a_{left} 会进入状态 B，得到奖励 0，选择 a_{right} 会进入终止状态，得到奖励 0。在状态 B 中，有 10 个可选的动作，每个动作执行完之后都会进入终止状态，得到的奖励服从正态分布 $N(-0.1,1)$。理想情况下，从状态 A 向左移动最终到达终止状态的期望奖励是 −0.1，向右移动到达终止状态的期望奖励是 0，智能体的最优选择是从状态 A 向右移动到达中止状态。

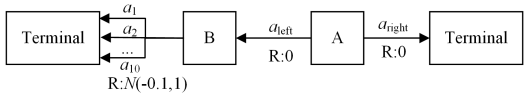


图 1 最大化偏差示例 MDP 过程

Fig. 1 Maximization bias example MDP

图 2 和图 3 展示了进行 5 000 次重复实验的结果。Q-Learning 在大约前 30 次迭代中选择向左动作的频率在不断增大，在 100 次迭代左右才降到 50%以下，到 500 次迭代后，向左的频率仍然比最优理想值高出 5%。其他算法并没有出现选择向左的频率大于向右频率的情况，而是从第 1 次迭代开始就不断减小向左的频率。WDQL 算法的结果与 DQL 近似，直到 500 次迭代后，DQL 向左的频率比最优理想值高出 1%左右，WDQL 高出 1.6%。WDQL-Softmax 表现最好，收敛速度最快，与最优理想值的偏差仅为 0.6%。

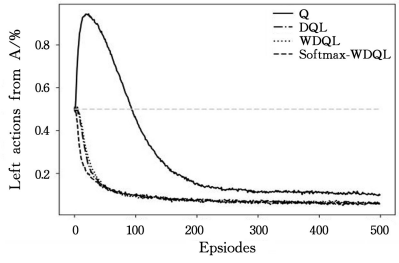


图 2 智能体向左移动百分比

Fig. 2 Percentage of agent moving to the left

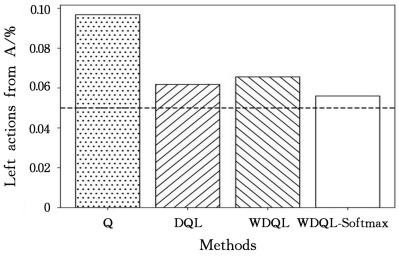


图 3 500 次迭代后，各算法中智能体向左移动百分比

Fig. 3 Percentage of agent moving to the left in each algorithm after 500 episodes

5.2 格子世界

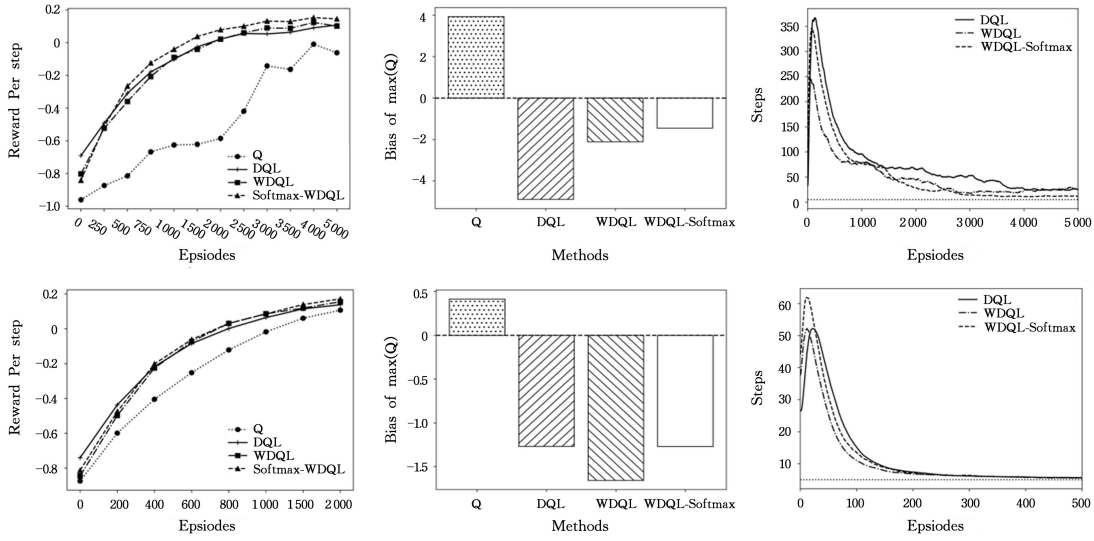
5.2.1 部分奖励随机

一个 $n\times n$ 的格子世界有 n^2 个状态和 4 个动作，即每个格子代表一个状态，每个格子(状态)下可以进行上下左右 4 个动作，选择动作后进入对应的相邻格子(状态)，当执行动作后要突破世界边缘时，将保持原状态，不允许离开格子世界。初始状态下智能体位于格子世界的左下角，终止状态位于右上角。为了保证奖励的随机性，规定当智能体在非终止状态下离开当前状态时，随机获得奖励值 −12 或 10，两者的概率相同；当智能体离开终止状态时，固定获得奖励值 5。最优策略为智能体花费 $2(n-1)$ 步从初始状态到达终止状态，用 1 步离开终止状态，此时平均每一步所得到的奖励为 $\dfrac{5+(-1)\times 2(n-1)}{2(n-1)+1}=\dfrac{7-2n}{2n-1}$。

实验中分别设置线性学习率 $\alpha=1/n(s,a)$ 和多项式学习

率 $\alpha=1/n(s,a)^{0.8}$，$n(s,a)$为智能体在状态 s 下执行动作 a 的次数，比较 Q-Learning 和 DQL 以及 WDQL-Softmax 在不同学习率和不同环境下的收敛速度和最大值偏差。其中，DQL 在更新Q_A表和Q_B表时，采用不同的学习率，更新Q_A表使用 $\alpha_A=1/n_A(s,a)$或$\alpha_A=1/n_A(s,a)^{0.8}$，$n_A(s,a)$表示智能体在$Q_A$表中状态 s 下动作 a 的更新次数，更新Q_B表同理。衰减因子 γ 设置为 0.95，从状态中选择动作时采用ϵ-greedy 策略，根据 ϵ的值平衡动作的探索与利用，$\epsilon=1/\sqrt{n(s)}$，$n(s)$代表状态 s 的访问次数。

图 4 展示了在 3×3 格子世界中迭代 5000 步时，4 种算法的各项指标值变化。图 4(a)显示了迭代步数与平均奖励的关系；图 4(b)显示了在进行 5000 次迭代后，在初始状态下，最大动作值与理论最优值的偏差；图 4(c)显示了迭代步数与步数(智能体从初始状态到终止状态所用步数)的关系。从图 4(a)中可以看出，因为环境中奖励值存在随机性，Q-Learning 算法与其他算法相比，收敛速度均很慢。在线性学习率下，Q-Learning 在 5000 步迭代后，平均奖励值与理论值仍存在不小的差距。其他 3 种算法均使用双估计器，从而保证收敛，其中，WDQL-Softmax 有效平衡了 WQ 的高估和 DQL 的低估，故收敛速度最快。



(a)平均每步奖励　(b)最大动作值偏差　(c)到达终止状态所用步数

图 4　线性学习率(上)和多项式学习率(下)下，3×3 格子世界中不同算法各项指标值

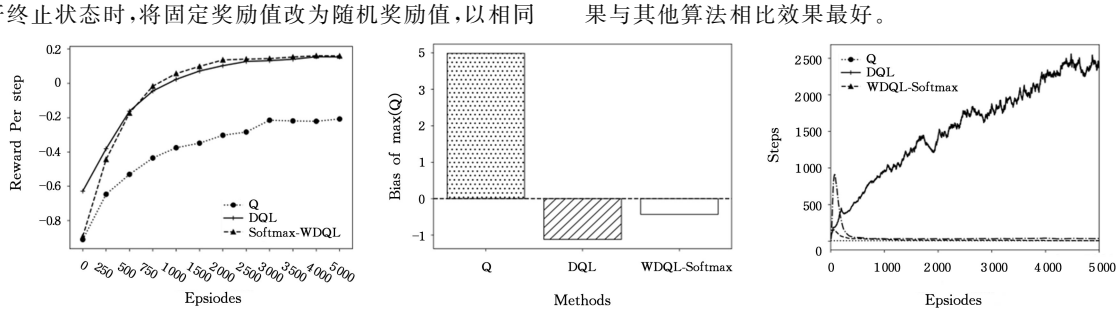
Fig. 4　Various indicator values of different algorithms in 3×3 grid world at different linear(up) and polynomial learning rate(down)

图 4(b)显示使用 Q-Learning 得到的估计值偏高，在线性学习率下高估尤为明显，使用 DQL 和 WDQL 得到的估计值偏低，使用 WDQL-Softmax 得到的估计值与理论值的偏差最小。因为收敛太慢，Q-Learning 的步数远超其他算法，故没有在图 4(c)中展示。在线性学习率中，WDQL 虽然前期收敛速度快，但是在 5000 次迭代后，步数在 20 左右提前收敛，只有 WDQL-Softmax 最终收敛在 5 步左右，且收敛速度比 DQL 快。在多项式学习率中，几种算法的收敛速度和收敛效果相当，在 500 次迭代后均收敛。

5.2.2　全部奖励随机

与 5.2.1 小节中的部分奖励随机实验不同，本实验改变了非终止状态下的每一步随机奖励的差值，从 −12 或 10 缩小为 −6 或 4，每一步奖励的期望值保持不变，仍为 −1。当智能体离开终止状态时，将固定奖励值改为随机奖励值，以相同概率随机获得奖励值 −30 或 40。实验参数与部分随机实验相同，$\gamma=0.95$，采用ϵ-greedy 策略，$\epsilon=1/\sqrt{n(s)}$。

图 5 展示了在 3×3 格子世界中迭代 5000 步时，3 种算法的平均每步奖励、最大动作值的偏差和到达终止状态所用步数变化情况。从图 5(a)中可以看出 Q-Learning 算法与其他算法相比，收敛速度很慢，并且有提前收敛的趋势。其他算法均使用双估计器，能保证收敛，其中，WDQL-Softmax 有效平衡了 WQ 的高估和 DQL 的低估，故收敛速度最快。图 5(b)中使用 Q-Learning 和 DQL 得到的估计值与理论值的偏差也印证了高估和低估问题，WDQL-Softmax 与理论值的偏差最小。图 5(c)展示了不同算法的步数变化情况，Q-Learning的步数远超其他算法。WDQL-Softmax 的步数峰值为 200 步，远远低于 DQL 的步数峰值，并且收敛速度和收敛结果与其他算法相比效果最好。



(a)平均每步奖励　(b)最大动作值偏差　(c)到达终止状态所用步数

图 5　线性学习率下，不同算法在 3×3 格子世界中的各项指标值

Fig. 5　Various indicator values of different algorithms in 3×3 grid world at different linear learning rate

表 1 列出了在不同实验下的 5000 次迭代后，各种算法的平均每步奖励的实际值与理论值的差值以及到达终止状态所用步数实际值与理论值的差值。可以看出，WDQL-Softmax 算法得到的平均每步奖励差值和步数差值均最小，说明该算法能有效提升收敛速度和收敛效果，且能够平衡高估和低估问题。

表 1　平均每步奖励偏差与总步数偏差

Table 1　Bias of average reward per step and total steps

Task	Algorithm	Bias of Average Reward Per Action	Bias of Steps
Part Reward Random	Q-Learning	0.4621	11790.0373
	DQL	0.0702	17.0966
	WDQL	0.0995	20.3681
	WDQL-Softmax	**0.0379**	**3.5770**
Total Reward Random	Q-Learning	0.4001	2341.1706
	DQL	0.0483	30.6654
	WDQL-Softmax	**0.0419**	**0.5772**

结束语　本文提出了一种新的无模型强化学习算法——基于 softmax 的加权 Double Q-Learning 算法，该算法将加权 Q-Learning 与 DQL 相结合，平衡高估与低估问题。在离散空间中进行了一系列实验，实验结果表明，基于 softmax 的加权 Double Q-Learning 算法能够有效提升收敛速度、降低估计误差，收敛效果与其他算法相比也有明显提升。未来将考虑研究该算法在连续空间中的表现和优化。

参 考 文 献

[1] WIERING M, VAN OTTERLO M. Reinforcement Learning: State of the Art[M]. New York: Springer, 2012.

[2] LI Y. Deep reinforcement learning: An overview [J]. arXiv: 1701.07274, 2017.

[3] KAISER L, BABAEIZADEH M, MILOS P, et al. Model Based Reinforcement Learning for Atari[C]// International Conference on Learning Representations. 2019.

[4] JOHANNINK T, BAHL S, NAIR A, et al. Residual reinforcement learning for robot control[C]// 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019: 6023-6029.

[5] KIRAN B R, SOBH I, TALPAERTV, et al. Deep reinforcement learning for autonomous driving: A survey[J]. IEEE Transactions on Intelligent Transportation Systems, 2021, 23(6): 4909-4926.

[6] WU X, CHEN H, WANG J, et al. Adaptive stock trading strategies with deep reinforcement learning methods[J]. Information Sciences, 2020, 538: 142-158.

[7] WATKINS C J C H, DAYAN P. Q-learning [J]. Machine learning, 1992, 8: 279-292.

[8] LEE D, DEFOURNY B, POWELLW B. Bias-corrected q-learning to control max-operator bias in q-learning[C]// 2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL). IEEE, 2013: 93-99.

[9] AZAR M G, MUNOS R, GHAVAMZADEH M, et al. Speedy Q-learning[C]// Advances in neural information processing systems. 2011: 2411-2419.

[10] HASSELT H. Double Q-learning[C]// Proceedings of the 23rd International Conference on Neural Information Processing Systems. 2010: 2613-2621.

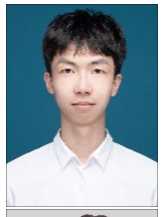
[11] D'ERAMO C, RESTELLI M, NUARA A. Estimating maximum expected value through gaussian approximation[C]// International Conference on Machine Learning. PMLR, 2016: 1032-1040.

[12] ZHANG Z, PAN Z, KOCHENDERFERM J. Weighted double Q-learning[C]// IJCAI. 2017: 3455-3461.

[13] REN Z, ZHU G, HU H, et al. On theEstimation Bias in Double Q-Learning[J]. Advances in Neural Information Processing Systems, 2021, 34: 10246-10259.

[14] WANG Y, LIU Y, CHENW, et al. Target transfer Q-learning and its convergence analysis[J]. Neurocomputing, 2020, 392: 11-22.

[15] SUTTON R S, BARTOA G. Reinforcement learning: An introduction[M]. MIT press, 2018.



ZHONG Yuang, born in 2000, postgraduate. His main research interests include reinforcement learning and so on.

YUAN Weiwei, born in 1981, Ph.D, Professor, Ph.D supervisor. Her main research interests include data mining and intelligence computing.