王文龙, 张帆, 唐超,等. 基于三值估算法的深度双确定性策略梯度算法[J]. 智能计算机与应用,2024,14(5):75-82. DOI: 10.20169/j.issn.2095-2163.240510

基于三值估算法的深度双确定性策略梯度算法

王文龙, 张 帆, 唐 超, 李 徐, 郝正阳, 张帆扬
(上海工程技术大学 机械与汽车工程学院, 上海 201620)

摘 要: 深度强化学习算法在机器人控制领域应用越来越广泛,但用于连续动作空间的算法,如 DDPG,一直存在估值高估的问题,在机器人控制领域应用尚不成熟。本文为了提高深度强化学习算法中目标值估值的准确性,得到更适用于机器人控制的深度强化学习算法,提出了一种基于三值估算法的深度双确定性策略梯度算法,该算法采用三值估算法来估计目标评论家网络的估值,去计算目标值作为当前网络的评估标准,采用双确定性策略网络在当前时间步数下生成最优策略,采用更适用于机械臂深度强化学习控制的 OU 噪声加到动作策略中。实验证明,该算法在复杂模型和环境中能够表现更好的性能。

关键词: 深度强化学习; 三值估算法; 双确定性策略; 机器人控制
中图分类号: TP18; TP242.6 **文献标志码**: A **文章编号**: 2095-2163(2024)05-0075-08

A gradient algorithm based on three-valued estimation method for depth double-deterministic strategy

WANG Wenlong, ZHANG Fan, TANG Chao, LI Xu, HAO Zhengyang, ZHANG Fanyang
(School of Mechanical and Automotive Engineering, Shanghai University of Engineering Science, Shanghai 201620, China)

Abstract: Deep reinforcement learning algorithms are increasingly widely used in the field of robot control, but algorithms used in continuous action spaces, such as DDPG, have always had the problem of overestimation, and their application in the field of robot control is not yet mature. In order to improve the accuracy of target value estimation in deep reinforcement learning algorithms and obtain a more suitable deep reinforcement learning algorithm for robot control, this paper proposes a deep doubly deterministic strategy gradient algorithm based on the three-value estimation method. The algorithm uses the three-value estimation method to estimate the target critic network's estimation, and calculates the target value as the evaluation standard for the current network, adopting a dual deterministic strategy network to generate the optimal strategy at the current time step, and incorporating OU noise that is more suitable for deep reinforcement learning control of robotic arms into the action strategy. Experimental results have shown that this algorithm can perform better in complex models and environments.

Key words: deep reinforcement learning; three-value estimation method; double-deterministic strategy; robot control

0 引 言

随着深度强化学习的快速发展和广泛应用,一系列的优秀算法也相继出现,均可有针对性地解决一些实际的问题。迄至目前,算法的应用范围已越来越广泛,但对算法的泛化能力和复杂模型的学习能力要求也变得更高。如何提高算法性能以及在机器人领域的应用即已成为热点的研究内容。

在深度强化学习算法中,基本可以分为基于价值和基于策略的算法,这 2 种算法都有各自优缺点。其中,基于价值的算法将神经网络加入到 Q-学习[1] 算法中, 成为深度强化学经典的 DQN(Deep Q-Learning Network)算法[1],解决了 Q-学习算法中 Q 表对大规模数据查找和储存缓慢造成的维度灾难的问题, 也将环境状态转换为高维图像数据,表现出了更优越的性能。针对 DQN 算法的估值高估问题,陆续研发了多种改进算法将基于值算法推向成熟,如双 DQN(Double DQN)、竞争 DQN(Dueling DQN)[3-4] 等。

基金项目: 上海市科委生物医药领域科技支撑计划(17441901200)。
作者简介: 王文龙(1995-),男,硕士研究生,主要研究方向:深度强化学习。
通讯作者: 张 帆(1980-),女,博士,副教授,主要研究方向:并联机器人机构学,医疗人机协作机器人,深度强化学习。Email: pdszhangfan@sues.edu.cn
收稿日期: 2023-03-23

DQN 算法适用于离散动作空间,但不适用于连续动作空间,针对这一问题,学者们提出了一系列的基于策略梯度算法[5],如深度确定性策略梯度算法(Deep Deterministic Policy Gradient, DDPG)。该算法使用演员-评论家(Actor-Critic)框架下结合 DQN 算法的优点的一种异策略(off-policy)算法[6];分布式分布确定性策略梯度(Distributed Distributional Deterministic Policy Gradient, D3PG)使用多个并行演员(Actor)网络收集数据,并分享一个大的经验池数据,与策略的学习分开,是 DDPG 算法的分布式版本[7]。双延迟深度确定性策略梯度(Twin Delayed Deep Deterministic Policy Gradient, TD3)解决了 DDPG 算法的估值高估问题,使用了双评论家(Critic)网络结构,计算目标值时取两者的较小值,抑制网络中的过高估计。计算目标值时,在下一个状体的预测动作上加入正则平滑项,对策略的评价更加准确,又将目标 Actor 网络的更新延迟,Critic 网络更新多次后再进行 Actor 网络的更新,使 Actor 网络的训练更加稳定,但存在估值低估问题[8]。

本文为了将深度强化学习算法更好地应用于机器人控制领域,提出了一种基于三值估算法的深度双确定性策略梯度算法(Three-value Estimation Method Deep Double Deterministic Policy Gradients, TEMD3),来减少 DDPG 算法高估和 TD3 算法的低估问题,并提高在复杂机器人模型和任务上的学习能力,经过实验证明该方法是有效的。

1 相关知识

深度强化学习是利用深度学习的感知功能以及强化学习的决策功能的算法,强化学习问题可以用一个定义为五元组 (S,A,r,p,γ) 的马尔可夫决策过程表示[9]。其中,S 为状态的集合,A 为动作的集合,r 为奖励函数,p 为状态转移概率,即在状态 s 执行动作 a 后转移到状态 s' 的概率值,γ 为折扣系数。DDPG 算法将 Q-learning 拓展到基于深度确定性策略算法连续运动空间,本文也是在这个算法的基础上进行研究和拓展。该算法将确定性策略参数化,以 $\pi(s,\varphi)$ 拟合动作策略,以最大化 Q 值逼近最优策略,目标是寻找最优策略神经网络参数 φ 使累计期望趋于最大化,即得到 φ^*,公式为:

$$\varphi^* = \arg\max_{\varphi} J(\varphi) = \arg\max_{\varphi} E_{\pi_\varphi}(R_t) \tag{1}$$

其中,R_t 为 t 时刻的奖励值,$J(\varphi)$ 为策略的目标函数。对 $J(\varphi)$ 进行梯度计算:

$$\tilde{\nabla}_\varphi J(\varphi) = \mathrm{E}_{\pi_\varphi}[\tilde{\nabla}_\varphi \log \pi_\varphi(a \mid s) Q_{\pi_\varphi}(s,a)] \tag{2}$$

其中,a 为动作;s 为状态,Q_{π_φ} 为动作策略的 Q 值,策略神经网络参数 φ 的更新公式为:

$$\varphi \leftarrow \varphi + \alpha \tilde{\nabla}_\varphi L(\varphi) \tag{3}$$

其中,α 为学习率,$L(\varphi)$ 为损失函数。

在训练中使用随机梯度下降(Stochastic Gradient Descent, SGD)方法来使损失函数最小化,得到策略梯度的最优参数 φ^*,因此具有更好的稳定性和更快的收敛速度。

在 DDPG 算法中的本地 Actor 网络采用批量采样的策略梯度方法进行优化,由 Silver 等学者[5]证明:

$$\tilde{\nabla}_\varphi J = \frac{1}{N}\sum_i \tilde{\nabla}_a Q(s,a \mid \theta)\mid_{s=s_i,a=\mu(s_i)} \tilde{\nabla}_\varphi \mu(s \mid \varphi)\mid_{s_i} \tag{4}$$

其中,s_i 表示采样样本中第 i 个智能体观测到的状态;θ 表示 Critic 网络的参数;$\mu(s \mid \varphi)\mid_{s_i}$ 表示 s_i 的状态下输入为状态 s,神经网络参数为 φ 的动作;$Q(s,a \mid \theta)$ 表示输入为状态 s 和动作 a,神经网络参数为 θ 的 Q 值。本地 Critic 网络使用均方差损失函数来进行梯度求解并优化参数,在这个过程中使用的目标网络对下一个策略动作进行估计作为目标值,估值为:

$$y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1} \mid \varphi) \mid \theta') \tag{5}$$

其中,r_i 表示第 i 个策略所得到的奖励值;Q' 表示目标 Critic 网络对下一个策略的估值;μ' 表示目标 Actor 网络对 s_{i+1} 的状态生成的下一个动作。网络损失函数 L 表示为:

$$L = \frac{1}{N}\sum_i (y_i - Q(s_i,a_i \mid \theta))^2 \tag{6}$$

其中,N 表示一个批次的数量。

2 基于三值估算法的深度双确定性策略梯度算法

2.1 基于三值估算法的深度双确定性策略梯度算法框架

基于三值估算法的深度双确定性策略梯度(Three-value Estimation Method Deep Double Deterministic Policy Gradients, TEMD3),在 DDPG 算法和 TD3 算法的基础上,进行了一系列的优化改进,采用了 2 套神经网络、即当前神经网络和目标神经网络,各使用了 5 个神经网络块、2 个 Actor 神经网络和 3 个 Critic 网络。这里,Actor 网络的输入为状态,输出为智能体需要执行的动作;Critic 网络的输入为执行的动作和状态,输出为对该动作的评价,即获得的 Q 值,当前神经网络和目标神经网络的区

别是当前神经网络处理当前的状态和动作，目标神经网络处理下一个状态和动作，每执行一次当前神经网络，将获得的当前状态 s，执行动作 a，获得的奖励值 r，以及智能体执行动作后下一个状态 s' 存储到经验回放池中。当满足一定经验后进行离线学习，通过目标网络获得目标的 Q 值，与当前的神经网络获得的 Q 值做均方差计算，获得损失函数，通过梯度求解实现 Critic 网络参数的更新。

在该算法中采用了双确定性策略，通过比较 2 个 Actor 网络获得的动作的 Q 值，将较大的 Q 值对应的动作作为当前神经网络的输出动作，并加入 Ornstein-Uhlenbeck(OU)噪声[10]来探索空间，作为智能体需要执行的动作策略。使用了 3 个 Critic 网络对获得的动作策略进行评估，当前神经网络得到 Q_1，Q_2，Q_3，目标神经网络利用这 3 个 Q 值使用三值估算法得到下一个状态和动作的估计 Q 值，通过贝尔曼方程计算得到目标 Q 值，TEMD3 算法框架图如图 1 所示。

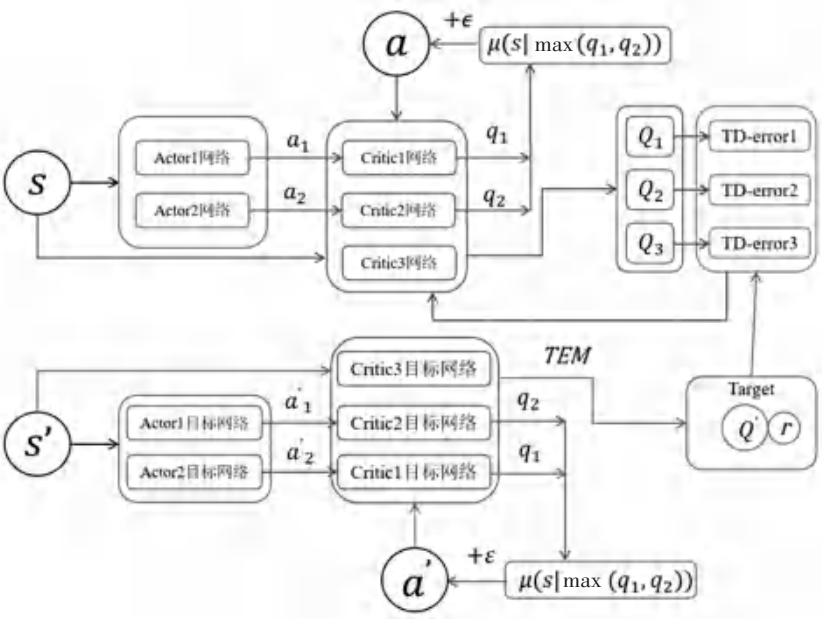

图 1　TEMD3 算法框架图

Fig. 1　Framework of TEMD3 algorithm

2.2　双确定性策略

双确定性策略使用 2 个 Actor 网络实现，Actor 作为策略生成的神经网络，通过对输入状态信息的映射，得到机器人的动作策略，用 π_φ 表示，参数为 φ，输入为状态 s，策略网络的输出动作为 $\pi_\varphi(s \mid \varphi)$，策略网络的参数见表 1。研究中，一层神经网络的输出变量和输入变量的函数关系式为：

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b_i\right) \tag{7}$$

表 1　策略网络参数信息表

Table 1　Strategic network parameters information

参数名称	描述
输入层	输入机器人的状态信息
隐藏层 1	400 个神经元
隐藏层 2	300 个神经元
激活函数	输出层为双曲正切函数(Tanh)，隐藏层为线性整流函数(*ReLU*)
归一化	隐藏层后都对输入的最后一维进行归一化，输出层除外

其中，$f(\cdot)$ 表示在该层神经网络中采用非线性激活函数，即 *ReLU* 和 Tanh 函数。

通过对强化学习算法的研究，在解决连续动作空间的算法中，DDPG 算法为确定性策略算法，仅有一个确定的动作策略，仅在每次动作策略后增加了一次噪音，进行一次空间探索，而在这次探索中对 Actor 输出的动作策略的好坏具有不确定性。SAC 算法[11]是面向最大熵强化学习开发的一种 off-policy 算法，使用的是随机策略，在最优的概率相同的多个动作策略中，随机选择其中一个动作，相比确定性策略具有较强的环境探索能力，遇到干扰时能更好地调整策略，训练速度较快。

为了增加 DDPG 等确定性策略算法的性能，减少 Actor 网络陷入局部最优策略的概率，本文引入了 2 个确定性策略网络，2 个网络会在一次运行中产生 2 个动作策略，增加对环境的探索能力，并且 2 个 Actor 同时局部最优的可能性呈指数级下降，1 个 Actor 网络出现局部最优的概率为 p，则 2 个 Actor

网络同时出现局部最优的概率为 p^2。但是在训练 Actor 网络时采用了同一批从经验池中采样的数据，所以数据分布相同。为了减少同分布数据带来的影响，采用 2 个不同的 Critic 网络分别对 2 个 Actor 网络进行评价和策略梯度优化，保证双 Actor 网络架构所具备的优秀性能。

初始化 2 个 Actor 网络，每个网络的输出用公式 $a_1 = \pi_{\varphi_1}(s \mid \varphi_1)$，$a_2 = \pi_{\varphi_2}(s \mid \varphi_2)$ 表示，则策略网络的动作策略输出为：

$$a = \underset{a}{\operatorname{argmax}}\{\max[Q(s,a_1 \mid \theta_1), Q(s,a_2 \mid \theta_2)]\} \quad (8)$$

其中，$Q(s,a_1 \mid \theta_1)$ 为 Critic 网络对状态 s 和第 1 个动作策略生成的动作 a_1 的评价值，网络参数为 θ_1，2 个 Actor 网络参数通过确定性策略梯度更新，公式为：

$$\tilde{\nabla}_{\varphi_{i=1,2}}(J(\varphi_i)) = N^{-1} \sum \tilde{\nabla}_a Q_{\theta_i}(s,a) \mid_{a=\pi_{\varphi_i}(s)} \tilde{\nabla}_{\varphi_i} \pi_{\varphi_i}(s) \quad (9)$$

其中，$\varphi_{i=1,2}$ 表示第 1 个或者第 2 个 Actor 网络的动作策略，$J(\varphi_i)$ 表示第 i 个 Actor 网络的损失函数。

在该算法中，采用了延迟更新的方法更新 Actor 网络的参数，即当 Critic 网络更新 2 次后再对 Actor 网络的参数更新，更新方式为式(3)给出的计算方法。

2.3 基于三值估算法的 Q 值平滑方法

DDPG 的一个主要问题是，DDPG 与基于价值的 Q 学习[12]方法一样存在高估问题，这可能会对函数近似的性能产生负面影响[13]。因此，有良好的价值估计是至关重要的，因为对评论家的更好的价值函数的估计可以促使行动者学习更好的策略。在双 DQN 算法中使用了双 Q - learning 来缓解基于价值的 Q - learning 方法的高估，在 TD3 算法中保留了这种思路，采用了双 Critic 网络对行为进行估值，具体公式为：

$$y = r + \gamma \min_{i=1,2} Q_i(s', \pi(s' \mid \varphi'); \theta_i') \quad (10)$$

其中，r 为当前动作的奖励值；γ 为折扣系数；$\min_{i=1,2} Q_i(s', \pi(s' \mid \varphi'); \theta'_i)$ 是求 Q_1，Q_2 的最小值；s' 为下一个状态；φ' 为目标 Actor 网络的动作策略；θ'_i 为第 i 个 Critic 网络的参数。

TD3 算法主要取 2 个 Critic 网络估值的最小值来估计值函数，但是可能会导致很大的低估偏差，影响模型的性能[14]。

为了得到更准确的估值函数，解决上述问题，TEMD3 算法采用了三值估算法对目标 Q 值进行估算，三值估算法是一种用于估算概率分布的方法，主要是利用最优可能值、最乐观值以及最悲观值进行期望值的计算。这种方法在于有限的数据和不确定的情况下估算概率分布，该方法能够更好地反映数据的不确定性，并能够提供更准确的估算结果，应用较为广泛。

在计算目标值的过程中，仅使用一个 Critic 网络的估值决定后续的累计奖励值，可能由于策略更新和当前状态网络训练的不充分导致估值的偏差过大，在反向传播中引起误差传播，因此在 TEMD3 算法中使用 3 个 Critic 网络对动作策略进行评估，得到 3 个 Q 值：$Q_1(s',a';\theta_1')$，$Q_2(s',a';\theta_2')$，$Q_3(s',a';\theta_3')$，使用三值估算法对 3 个 Q 值进行更精确的估算来代替对目标 Critic 网络对下一个状态的估值 Q，不妨假设：

$$Q_1(s',a';\theta'_1) < Q_2(s',a';\theta'_2) < Q_3(s',a';\theta'_3),$$

则目标 Critic 网络对下一个状态的预测动作的估值计算公式为：

$$Q(s',a') = (Q_1(s',a';\theta'_1) + 4 \times Q_2(s',a';\theta'_2) + Q_3(s',a';\theta'_3))/6 \quad (11)$$

则该网络的目标值为：

$$y = r + \gamma Q(s',a') \quad (12)$$

该目标值与当地网络的 $Q_1(s,\pi(s \mid \varphi);\theta_1)$，$Q_2(s,\pi(s \mid \varphi);\theta_1)$，$Q_3(s,\pi(s \mid \varphi);\theta_1)$ 进行均方误差的计算，作为损失函数对本地网络进行梯度更新，计算方法如下：

$$L_i = \frac{1}{N} \sum_{j}^{N} (y_j - Q_{i=1,2,3}(s_j, \pi(s_j \mid \varphi); \theta_i))^2 \quad (13)$$

其中，N 为样本数；y_j 为第 j 样本的目标值；$Q_{i=1,2,3}(s_j,\pi(s_j \mid \varphi);\theta_i)$ 为第 j 样本的 3 个本地策略网络的估值。

在本地神经网络进行学习和参数更新后，需要对算法的目标网络进行更新，除了上节提到的延迟更新方法外，5 个神经网络的参数采用软更新的方式[2]，具体公式为：

$$\theta'_{i=1,2,3} = \tau\theta_i + (1-\tau)\theta'_i \quad (14)$$

$$\varphi'_{j=1,2} = \tau\varphi_j + (1-\tau)\varphi'_j \quad (15)$$

其中，θ_i 表示当前第 i 个 Critic 网络的参数；i 等于 1、2、3；θ'_i 表示第 i 个目标 Critic 网络的参数；φ_j 表示第 j 个 Actor 网络的参数，j 等于 1、2；φ'_j 表示第 j 个目标 Actor 网络的参数；τ 为软更新时对当前网络参数的贪婪程度。

2.4 噪音的选择

由于 TEMD3 算法中策略选择为确定性的，所以在该算法中当前 Actor 网络获得的动作增加了 OU 噪声，强化对环境的探索能力，弥补策略网络在这方面的不足。在目标 Actor 网络中增加了高斯噪音，作为目标网络的平滑正则项。

高斯噪音为策略网络直接的输出作为均值，再叠加上高斯分布 $\varepsilon \sim N(0,\sigma^2)$，则执行策略满足 $\pi_\varphi(s)+\varepsilon \sim N(\pi_\varphi(s),\sigma^2)$，高斯噪音明显在时序上是不相关的，前一步和后一步选取动作的时候噪声是独立的，高斯噪音时刻如图 2 所示。

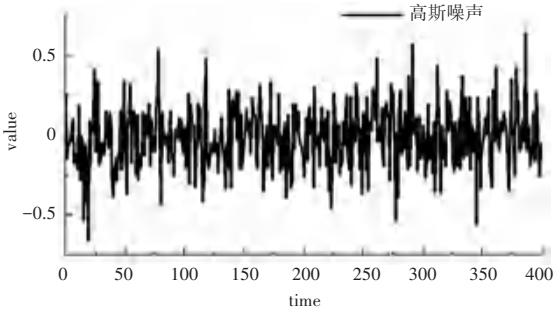

图 2　高斯噪声时刻图

Fig. 2　Gaussian noise time chart

而 OU 噪声适用于惯性系统，尤其是时间离散化度较小的情况，此外也可以保护实际系统[15]。OU 噪声往往不会如高斯噪声一样相邻的 2 步的值差别那么大，而是会绕着均值附近正向或负向探索一段距离，有利于在一个方向上探索，波段形式如 OU 噪声如图 3 所示。

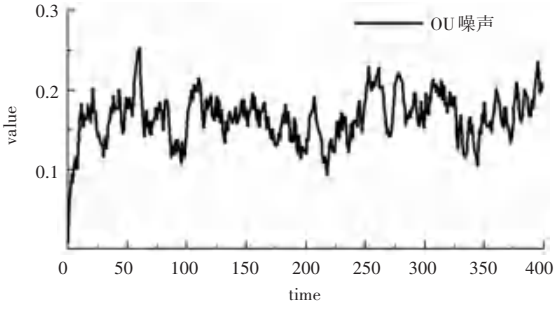

图 3　OU 噪声时刻图

Fig. 3　OU noises time chart

当前 Actor 网络使用 OU 噪声作为环境探索的噪声，原因是该算法的使用对象为机械臂，该对象的环境是带有惯性系统的，高斯噪声在这种独立的噪声前后 2 步相差较大，会被过滤掉，而且高斯噪声使得速度和位移的探索极为有限，OU 噪声可以探索得更远，如果使用独立噪声，当时间离散的粒度越小，要维持同样的随机程度，则需要每一步噪声方差就要越大，这将导致前后 2 步相差较远，并不符合真实的机械臂。

在“FetchReach-v1”的机械臂到达目标位置环境中进行 OU 噪声和高斯噪声的对环境探索的对比试验，在其他超参数和使用技巧相同的条件下，进行 100 k 帧的训练，完成平均奖励值的收敛，不同噪音对训练的影响如图 4 所示。

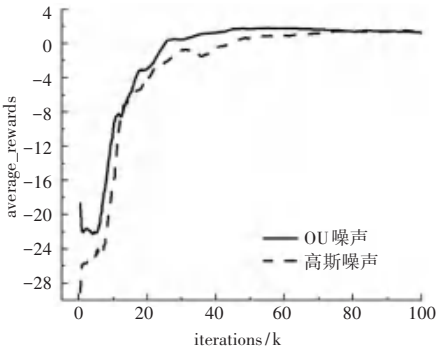

图 4　不同噪音对训练的影响

Fig. 4　Effect of different noises on training

由图 4 可知，在机械臂的环境中，OU 噪声能够更早地收敛到环境的最大平均奖励值，环境探索时间较短，训练较为稳定。

综上所述，在该算法中使用 OU 噪音进行环境的探索，叠加在本地 Actor 网络策略输出后，使用高斯噪音作为目标 Actor 网络输出后策略的平滑正则项。为了使模型训练后期模型较稳定，就会随着训练回合的增多，缓慢减少噪声的大小，增大策略的准确性，从而减少噪声多最优策略的影响。每次叠加噪声的值的具体计算为：

$$\zeta_i = \frac{\zeta}{\sqrt{i}} \tag{16}$$

其中，i 表示训练的回合数。随着回合数的增长，叠加的噪声随之减少。通过对 TEMD3 算法各个模块的设计，算法的伪代码描述如下。

算法 1　TEMD3 算法

1.初始化 Actor 网络 $\pi_{\varphi_1},\pi_{\varphi_2}$，和 Critic 网络 $Q_{\theta_1},Q_{\theta_2},Q_{\theta_3}$

2.随机初始化 5 个网络的参数 $\varphi_1,\varphi_2,\theta_1,\theta_2,\theta_3$

3.初始化目标网络参数

$\theta_1' \leftarrow \theta_1,\theta_2' \leftarrow \theta_2,\theta_3' \leftarrow \theta_3,\ \varphi_1' \leftarrow \varphi_1,\ \varphi_2' \leftarrow \varphi_2$

4.初始化重放缓冲区 B

5.for $ep = 1$ to M do：

6.　for $t = 1$ to T do：

7.　根据当前策略网络选择 2 个动作策略

$a_1 = \pi(s \mid \phi_1), a_2 = \pi(s \mid \phi_2)$

8.　分别用不同的 2 个 Critic 网络计算 a_1,a_2 的估值 q_1,q_2

9.　根据 2 个估值选择动作：$a_m = a_1$ if $q_1 \geqslant q_2$ else a_2

10.　叠加探索噪声完成最终策略输出：

$a = a_m + \zeta, a \in (-1,1)$

11.　与环境交互得到当前动作的奖励值 r 和

下一个状态 s'

12. 将数据 (s,a,r,s') 存入全保留经验池 B 中

13. 从经验池 B 中随机采样一个包含 N 个样本 (s,a,r,s')

14. 对下一个状态进行动作预测：

如果公式 $Q_{\theta_1'}(s',\pi_{\varphi_1'}(s');\theta_1') \geqslant Q_{\theta_2'}(s',\pi_{\varphi_2'}(s');\theta_2')$ 成立

则 $a_m' = \pi_{\varphi_1'}(s')$

否则 $a_m' = \pi_{\varphi_2'}(s')$

15. 对目标 Actor 网络输出增加平滑项作为下一个动作的预测值：$a' \leftarrow a_m' + \varepsilon, \varepsilon \sim clip(N(0,\sigma^2), -c, c)$

16. 计算下一个预测动作的估值：

$Q(s',a') = (\min_{i=1,2,3} Q_{\theta_i}(s',a') + \max_{i=1,2,3} Q_{\theta_i}(s',a') + 4 \times median_{i=1,2,3} Q_{\theta_i}(s',a'))/6$

17. 计算目标动作值：$y = r + \gamma Q(s',a')$

18. 通过最小化损失更新 Critic 网络参数 $\theta_{i=1,2,3}$：

$$L_{i=1,2,3} = \frac{1}{N}\sum_j (y - Q_{\theta_i}(s,a))^2$$

19. If t mod d then：

20. 通过确定性策略梯度更新 Actor 网络参数

$\varphi_{i=1,2}: \tilde{\nabla}_{\varphi_{i=1,2}}(J(\varphi_i)) =$

$N^{-1} \sum \tilde{\nabla}_a Q_{\theta_i}(s,a) \mid_{a=\pi_{\varphi_i}(s)} \tilde{\nabla}_{\varphi_i} \pi_{\varphi_i}(s)$

21. 更新目标网络参数：

$\theta_{i=1,2,3}' = \tau\theta_i + (1-\tau)\theta_i'$, $\varphi_{i=1,2}' = \tau\varphi_i + (1-\tau)\varphi_i'$

22. End if

23. End for

24. End for

3 TEMD3 算法创新验证

为了使设计的算法更适合应用于机器人运动控制，本文中采用 OPenAI Gym 环境[16] 中 MuJoCo 物理引擎[17] 中的“FetchReach-v1”机械臂作为验证环境验证提出的算法的模块的性能。

3.1 基于三值估算法的 Q 值估算的验证

简单起见，为了证明三值估算法对动作策略的评估的准确性和有效性，将 TEMD3 算法去掉双 Actor 网络模块，设计成基于三值估算法的深度确定性策略梯度算法（TEMD2）。与 DDPG 算法进行对比，TEMD2 算法为 DDPG 算法一个变体，用三值估算法对目标函数估算是 2 种算法唯一区别。在 TD3 算法中为了减少估值的高估，采用了 2 个 Critic 对行为策略进行估值并取最小值的方法，将这种方法抽离出来，设计新算法，记为 MD2，比较这 3 种估值函数方法对整个训练的作用，证明基于三值估算法的 Q 值估算的方法有效性，以及优于目前算法。

通过将近 300 k 帧的训练，每 100 帧的平均奖励如图 5 所示。训练中，Actor 网络的损失曲线如图 6 所示。从图 6 中看出 TEMD2 算法在同参数条件下表现出更好的收敛性和稳定性，在 3 个算法中收敛速度也是最高的，Actor 网络生成的动作策略的估值较高。

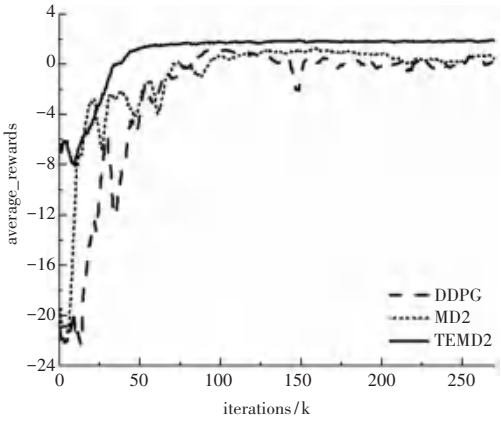

图 5 平均奖励曲线

Fig. 5 Average reward per 100 frames

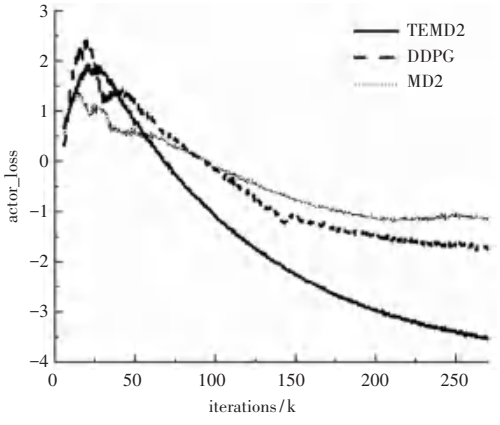

图 6 Actor 网络的损失曲线

Fig. 6 Loss curve of Actor network

3.2 双 Actor 网络效果验证

本节对比 TEMD3 算法、去掉一个 Actor 网络的 TEMD3 算法，验证其有效性。在该实验中，除了 Actor 网络的数量有差异，保持算法的其他部分不变，并且共有参数保持一致，保证实验的可靠性和必要性，每 100 帧的平均奖励如图 7 所示。Actor 网络的损失函数如图 8 所示。

由图 7、图 8 可知，对于 TEMD3 算法，Actor 网络相比于单 Actor 网络能更快速地达到环境所包含

的最大奖励值，在同一帧所采用的行为策略的估值更高，Actor 网络的损失函数更加平滑，收敛性能更好。在训练过程中，稳定性更好。综合来看，双 Actor 性能优于单 Actor。

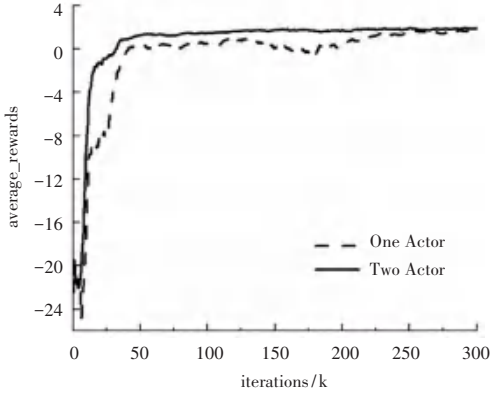

图 7　平均奖励

Fig. 7　Average rewards per 100 frames

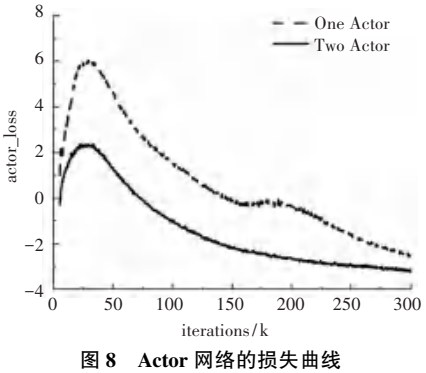

图 8　Actor 网络的损失曲线

Fig. 8　Loss curve of Actor network

3.3　Actor 网络延迟更新验证

在 TEMD3 算法中 2 个 Actor 目标网络的更新策略采用了延迟更新，为了证明该方法在 TEMD3 算法中的有效性，进行了对比试验，采用没有延迟更新。即 $delay = 1$，和带有延迟的 Actor 网络参数的更新，其延迟步数为 2，即 $delay = 2$。训练的 Actor 网络的损失曲线如图 9、图 10 所示。

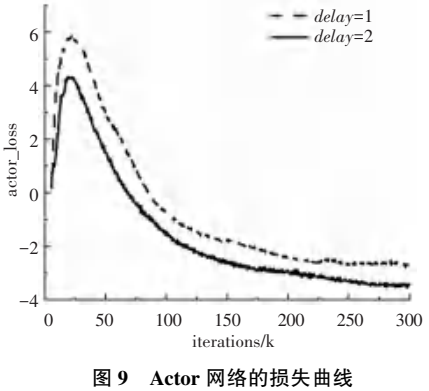

图 9　Actor 网络的损失曲线

Fig. 9　Loss curve of Actor network

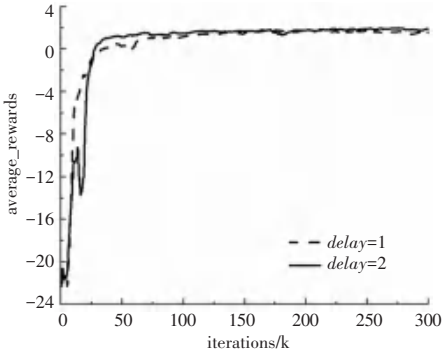

图 10　每 100 帧的平均奖励

Fig. 10　Average reward per 100 frames

从图 9 和图 10 中可以看出延迟更新对 Actor 网络的收敛情况影响较大，其值为 Critic 网络获得的评价结果的负值，故而能够收敛到更高的 Critic 值，从平均奖励图中能得出，有延迟更新的模型能够达到更优的奖励值，并且能够提前达到环境的可收敛的奖励值。

3.4　TEMD3 算法整体验证

使用完整的 TEMD3 算法在机械臂的环境中进行训练验证，采用 Adam 优化器学习神经网络的参数[18]，Actor 网络和 Critic 网络学习率分别设置为10^{-4}和10^{-3}。与目前确定性策略梯度算法中在机械臂的运动控制中表现较好的算法（TD3 算法）做性能对比。

在 300 k 帧的训练中，每 100 帧的平均奖励值如图 11 所示。在相同的条件下，就平均奖励值的收敛速度和收敛程度而言、TEMD3 均高于 TD3 算法，表现出良好的性能。Actor 网络的损失曲线如图 12 所示。Actor 损失函数主要作用为更新策略网络参数，找到最优策略，相对应的目标就是得到最大化奖励值，奖励值最大化将会带来 Q 值的最大化，因此策略网络的训练目标就是最大化 Q 值，在本文的采用 $L_A = -Q(s,a)$ 为损失函数的基础下，Actor 网络的损失函数为最小化 L_A。从图中可以看出，TEMD3 算法在训练中每一帧达到的更大的 Q 值。

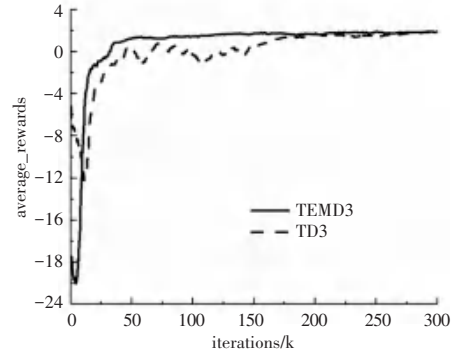

图 11　每 100 帧的平均奖励

Fig. 11　Average reward per 100 frames

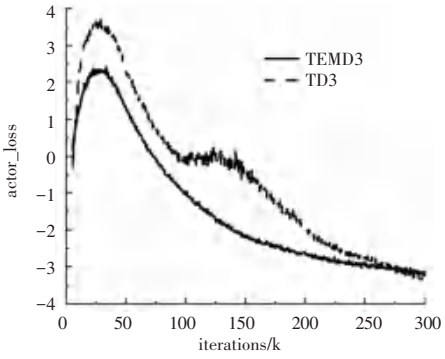

图 12 Actor 网络的损失曲线

Fig. 12 Loss curve of Actor network

图 13 为每个回合对训练的策略网络进行验证所得到的奖励值的曲线图,从图 13 中可以看到 TEMD3 算法比最先进的 TD3 算法产生更好的值估计,可以提高低估偏差,并有更好的性能和更高的样本效率,能够提升将近 50%的收敛速度。实验结果表明,TEMD3 优于最先进的算法—TD3 算法。

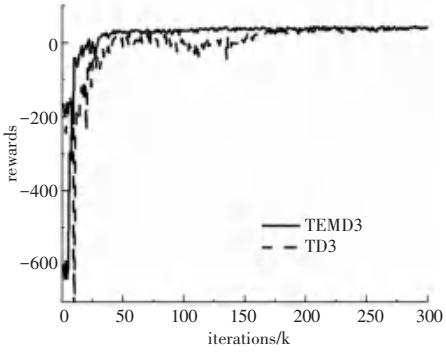

图 13 每回合奖励值

Fig. 13 Reward values for each iteration

4 结束语

总结目前在深度强化学习算法领域的优秀算法,为了解决 DDPG 算法和 TD3 算法中的目标值高估低估问题,同时提升在复杂机器人模型中的学习能力,提出了一种基于三值估算法的深度双确定性策略梯度算法。该算法使用三值估算法确定目标值的准确估计,并采用双确定性策略,以及延迟更新等方法,让该算法性能有明显的提升,更适应于机械臂等复杂环境中的训练和学习,提高了智能体对环境的探索能力,加快训练速度,降低了学习成本。但是 TEMD3 算法仅使用了随机采样和全保留的经验回放的方法,还有更加有效的提高训练速度的方法可以使用,比如采用双经验池或优先重放方法对数据进行采样。

参考文献

[1] WATKINS C J C H, DAYAN P. Q-learning[J]. Machine Learning, 1992,8(3-4):279-292.

[2] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Playing atari with deep reinforcement learning[J]. arXiv preprint arXiv:1312.5602,2013.

[3] HASSELT V H, GUEZ A, SILVER D. Deep reinforcement learning with double q-learning[C]//Thirtieth AAAI Conference on Artificial Intelligence. Arizona, USA:AAAI, 2016:1-12.

[4] WANG Ziyu, SCHAUL T, HESSEL M, et al. Dueling network architectures for deep reinforcement learning[C]//International Conference on Machine Learning. New York,USA:ACM , 2016:1995-2003.

[5] SILVER D, LEVER G, HEESS N, et al. Deterministic policy gradient algorithms[C]// Proceedings of the 31st International Conference on International Conference on Machine Learning (ICML). Beijing,China:ACM, 2014:387-395.

[6] LILLICRAP T P, HUNT J J, PRITZEL A, et al. Continuous control with deep reinforcement learning[J]. arXiv preprint arXiv:1509.02971, 2015.

[7] BARTH-MARON G, HOFFMAN M W, BUDDEN D, et al. Distributed distributional deterministic policy gradients[J]. arXiv preprint arXiv:1804.08617, 2018.

[8] FUJIMOTO S, HOOF H V, MEGER D. Addressing function approximation error in Actor-Critic methods[J]. arXiv preprint arXiv:1802.09477, 2018.

[9] BELLMAN R A. Markovian Decision process[J]. Journal of Mathematics and Mechanics, 1957, 6(5): 679-684.

[10] UHLENBECK G E, ORNSTEIN L S. On the theory of the brownian motion[J]. Physical Review, 1930,36(5):823.

[11] HAARNOJA T, ZHOU A, ABBEEL P, et al. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor [C]//International Conference on Machine Learning. Stockholm:PMLR, 2018: 1861-1870.

[12] WATKINS C J C H. Learning from Delayed Rewards[D]. Cambridge, UK:University of Cambridge, 1989.

[13] THRUNS, SCHWARTZ A. Issues in using function approximation for reinforcement learning [C]// Proceedings of the 1993 Connectionist Models Summer School. Hillsdale, USA: Lawrence Erlbaum Publisher, 1993:1-10.

[14] LAN Qingfeng, PAN Yangchen, FYSHE A, et al. Maxmin q-learning: Controlling the estimation bias of q-learning[J]. arXiv preprint arXiv:2002.06487, 2021.

[15] WAWRZY′NSKI P. Control policy with autocorrelated noise in reinforcement learning for robotics[J]. International Journal of Machine Learning and Computing, 2015,5:91-95.

[16] BROCKMANG, CHEUNG V, PETTERSSON L, et al. Openai gym[J]. arXiv preprint arXiv:1606.01540, 2016.

[17] TODOROV E, EREZ T, TASSA Y. Mujoco: A physics engine for model-based control[C]// 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Portugal: IEEE, 2012:5026-5033.

[18] DIEDERIK K, JIMMY B. Adam: A method for stochastic optimization[J]. arXiv preprint arXiv:1412.6980, 2014.