文章编号: 1001-0920(2018)04-600-07 DOI: 10.13195/j.kzyjc.2017.0261

基于重抽样优选缓存经验回放机制的深度强化学习方法

陈希亮†, 曹 雷, 李晨溪, 徐志雄, 何 明

(解放军理工大学 指挥信息系统学院，南京 210007)

摘 要: 针对深度强化学习算法中经验缓存机制构建问题,提出一种基于TD误差的重抽样优选缓存机制;针对该机制存在的训练集坍塌现象,提出基于排行的分层抽样算法进行改进,并结合该机制对已有的几种典型基于DQN的深度强化学习算法进行改进. 通过对Open AI Gym平台上Cart Port学习控制问题的仿真实验对比分析表明,优选机制能够提升训练样本的质量,实现对值函数的有效逼近,具有良好的学习效率和泛化性能,收敛速度和训练性能均有明显提升.

关键词: 深度强化学习；缓存回放；重抽样

中图分类号: TP273 **文献标志码:** A

Deep reinforcement learning via good choice resampling experience replay memory

CHEN Xi-liang†, CAO Lei, LI Chen-xi, XU Zhi-xiong, HE Ming

(Institute of Command Information Systems, PLA University of Science and Technology，Nanjing 210007，China)

Abstract: In order to build a good experience memory mechanism for deep reinforcement learning, a kind of resample choosing optimal memory cache construction method based on TD error is proposed. Ranking based algorithms on stratified sampling are also developed to avoid the collapse of training data set. Combined with this mechanism, several typical depth based onreinforcement learning algorithms based on DQN(deep Q-networks) are improved. Through the simulation on the control problem of Cart Port on Open AI Gym, experimental results show that the optimization mechanism improves the quality of training samples, and it can effectively enhance the learning value function, and has good learning efficiency and generalization performance. The convergence speed and training performance are improved significantly.

Keywords: deep reinforcement learning；experience replay；resample

0 引 言

强化学习(RL)是一类求解序贯决策的机器学习方法. 与监督学习中的导师信号不同,RL中的激励信号只是对动作好坏的一种评价,而不是告诉强化学习如何产生正确的动作,因此激励信号比导师信号更易获得. 强化学习利用与环境的交互反馈信号修正动作选择策略,以最大化期望回报为学习目标[1-2]. 对于一些决策控制问题,导师信号无法获取,而回报函数相对易于设计,因此成为求解一些复杂决策问题的有效手段.

然而,在很多实际情况中,状态维度和动作维度过高,使得Agent在巨大的状态空间或动作空间下,很难或无法遍历所有情况,导致算法收敛速度慢或无法学到合理的策略. 解决上述问题的一个有效途径是使用函数近似的方法,将值函数或策略用一个函数显性地进行表示. 常用的近似函数有线性函数、核函数、神经网络等[3-5],而近年来最成功的就是将深度神经网络作为近似函数引入到强化学习中. 深度神经网络不仅具有强大的逼近能力,而且实现了端到端的学习,能够直接从原始数据的输入映射到分类或回归. 在使用深度神经网络进行值函数近似方面,近年来成功的应用开始于DQN(deep Q-networks)算

收稿日期: 2017-03-13；**修回日期:** 2017-06-13.

基金项目: 国家自然科学基金项目(61301159, 61303267)；国家重点研发计划项目(2016YFC0800606)；江苏省自然科学基金项目(BK20150721, BK20161469).

责任编委: 侯忠生.

作者简介: 陈希亮(1985–), 男, 博士生, 从事机器学习、决策支持理论与技术的研究；曹雷(1965–), 男, 教授, 从事指挥信息系统工程、决策理论与方法等研究.

†通讯作者. E-mail: 383618393@qq.com

法[6], DQN之前的RL算法[7-8]主要依赖于人工标记的特征和线性模型、加性模型或神经网络等方法对值函数或策略进行表达,导致了系统性能依赖于提取的特征质量和近似模型的逼近能力.

随着深度学习领域研究的进展,直接从原始感知数据中自适应地提取高层特征变成可能,开始出现了二者相结合的技术,即深度强化学习. 方法包括: 使用深度神经网络拟合强化学习中最优动作估值函数[9]; 使用限制玻尔兹曼机[10](RBM)模型用于估计价值函数或策略函数等;使用自编码机模型用于提取特征[2],然后利用这些特征在NFQ算法中学习非线性策略并在多个基准数据集中获得比传统算法更好的效果. 通过直接输入游戏的原始视频作为状态进行强化学习,使用经验回放和目标分离两个思想解决训练稳定性的问题[6]. 在传统的DQN基础上引入Target network进一步扩展[9],此后DQN方法又有了新的发展,为解决神经网络中的估计偏差问题,提出采用Double DQN方法来解耦选择和评估耦合问题[11]. 但是先前的工作主要在记忆缓存中均匀随机地进行采样,而不考虑其重要性,这种方式导致学习中获取的稀缺的经验数据可能被快速遗忘. 为提高训练效率,Schaul等[12]提出了Prioritized replay的采样策略,对采样的数据单元进行重要性评估,取得了较好的效果. 但Prioritized replay算法在根据重要性选取训练样本时需要对采样的训练集根据重要性进行排序,增加了算法的复杂性.

为进一步提升算法的效率,降低算法复杂性,本文提出一种优选缓存经验回放机制,并利用该机制对Nature DQN、DSarsaN、double DQN等几种深度强化学习算法进行改进,既能对采样数据单元进行重要性评估,增加稀缺经验的回放概率,又能减少算法的复杂性,加快算法的收敛速度. 通过对Open AI Gym平台上Cart port学习控制问题的仿真实验对比分析表明,优选机制能够提升训练样本的质量,实现对值函数的有效逼近,具有良好的学习效率和泛化性能,收敛速度和训练性能均有明显提升.

1　相关工作

1.1　强化学习基础

强化学习是系统从环境中学习以使得奖励最大化的机器学习方法. 强化学习和有监督学习的主要不同之处在于监督信号的不同: 强化学习的监督信号作为执行当前“动作”的奖励,可用感知环境的奖惩函数获得;监督学习的监督信号作为正确的样例标记使用. 标准的强化学习设置为一个Agent在离散时间步长的方式与环境ϵ进行交互,可用马尔科夫决策过程(MDP)描述[2,12], MDP可以定义为一个5元组(S, A, π, R, γ). 在每一个时间步t, Agent接收状态s_t,依据策略π在可能的动作集合A中选择一个动作,π是s_t到a_t的映射. 作为回报, Agent接收到下一个状态的s_{t+1}和一个奖励R_t,该过程持续到Agent达到最终状态并重新启动, $R_t = \sum_{k=0}^{\infty} \gamma^k r^{t+k}$是在时间步长$t$采用折扣因子$\gamma \in (0,1)$计算的累计收益. Agent的目标是在每个状态$s_t$最大限度地提高预期收益.

强化学习的目的是要学习一个策略. 该策略是一个函数,输入为当前的状态s,输出为采取动作a的概率$\pi(s,a)$. 强化学习的策略便是选择相应的行动使未来的奖励最大化[1].

状态-动作值函数$Q^{\pi}(s,a) = E\{R_t|s_t = s, a\}$表示策略$\pi$的情况下,在状态$s$出发,选取动作$a$后使用策略$\pi$的累计期望回报. 最优值函数是采用任意策略在状态为s、动作为a时的最大动作值回报[1],有

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a) = \max_{\pi} E[R_t|s_t = s, a_t = a, \pi]. \tag{1}$$

最优状态-动作值函数服从贝尔曼方程[1],基于以下假设: 如果在下一时间步长的序列s'的最优值$Q^*(s',a')$对于所有可能的动作a'是已知的,则最佳策略是选择动作a',使得期望值$r+\gamma Q^*(s',a')$最大[1],有

$$Q^*(s,a) = E_{s' \sim \varepsilon}[r + \gamma \max_{a'} Q^*(s',a')|s,a]. \tag{2}$$

同样,状态值函数$V^{\pi}(s) = E\{R_t|s_t = s\}$是策略$\pi$时状态$s$的值函数,是策略$\pi$时状态$s$的期望回报. 因此状态值函数的$\gamma$折扣累计回报为

$$V^{\pi}(s) = E\Big\{\sum_{t=0}^{\infty} \gamma^t r^{t+1}|s_0 = s\Big\}. \tag{3}$$

对于状态空间和动作都是离散有限的情况下,可以使用传统的强化学习方法进行解决,如时序差分算法、Sarsa算法、Q-learning等. 它们使用表格记录值函数,为每个状态或状态-动作对分配一个存储空间,记录其对应的函数值. 虽然基于表格的经典算法在小规模离散空间的强化学习任务上表现不错,但更多的实际问题中状态数量很多,甚至是连续状态空间,这时经典强化学习方法难以有效地进行学习. 解决的方法有分层强化学习、迁移强化学习和值函数近似的方法.

值函数逼近法使用函数逼近器替代传统强化学习中的查表法实现泛化,以克服维数灾难. 该方法逼近器的结构和参数选取直接决定了算法的效率,对于

连续状态空间或动作空间不大的MDP问题,一般可以直接采用离散化方法泛化,对于大规模MDP或连续空间MDP问题,则需要采用更复杂的函数逼近器,如神经网络、核机器等.早期的研究中,值函数逼近器的结构和控制参数都是不变的,近年来进化值函数逼近成为研究热点[3-4].

强化学习算法的基本思想是,通过使用Bellman方程作为迭代更新估计动作值函数$Q^*(s,a) = E_{s'\sim\varepsilon}[r+\gamma\max_{a'}Q^*(s',a')|s,a]$. 这样的值迭代算法在$i\to\infty$时$Q_i\to Q^*$,收敛到最优的动作值函数[1]. 在实践中,由于动作值函数是为每个序列单独估计的,难以获取,通常使用函数近似估计动作值函数$Q(s,a;\theta)\approx Q^*(s,a)$. 一般采用线性函数近似,有时也使用神经网络这样的非线性函数近似. 在采用神经网络时,θ表示具有权重θ的神经网络函数近似器[6].

动作值函数$Q(s,a;\theta)$的参数θ通过最小化损失函数的方式进行计算,损失函数定义为

$$L_i(\theta_i)=E_{s'\sim\varepsilon}(y_i-Q(s,a;\theta_i))^2. \tag{4}$$

其中:s'为状态s后状态,$y_i=E_{s'\sim\varepsilon}(r+\gamma\max_{a'}Q(s',a';\theta_{i-1}))$. 求取最优动作是在前代网络参数$\theta_{i-1}$保持固定的情况下,优化损失函数$L_i(\theta_i)$. 对损失函数的参数进行微分得到梯度公式[6]

$$\nabla_{\theta_i}L_i(\theta_i)=E[(r+\gamma\max_{a'}Q(s',a';\theta_{i-1})- Q(s,a;\theta_i))\nabla_{\theta_i}Q(s,a;\theta_i)]. \tag{5}$$

强化学习通过求解Bellman最优化方程

$$Q^*(s,a)=E_{s'\sim\varepsilon}[r+\gamma\max\nolimits'_aQ^*(s',a')|s,a] \tag{6}$$

得出最优解策略

$$\pi^*=\mathrm{argmax}_\pi Q(s,\pi(s)),\ s\in S. \tag{7}$$

1.2 DQN中经验回放机制

Mnih等[6]在NIPS基础上提出DQN算法,一方面利用深度神经网络进行动作值函数的近似,另一方面采用Experience replay机制(如图1所示),将探索环境得到的数据以记忆单元$(s_t,a_t,r_{t+1},s_{t+1})$的形式储存起来,然后采取从Experience replay memory中随机选取样本的方式更新(训练)神经网络的参数,称为Nature DQN. DQN采用参数为θ的深层神经网络值函数进行近似,$Q(s,a;\theta)\approx Q^*(s,a;\theta)$,这种无模型的强化学习算法可以解决“模型灾难问题”,采用值函数的泛化逼近方法可以解决强化学习的“维数灾难问题”.

然而,研究者在对Nature DQN的实验中发现[9,13-15],采用Nature DQN训练的强化学习在对Q函数进行逼近时存在不稳定现象,主要原因有:1)观察序列的数据具有较大的相关性,导致基于梯度下降的优化算法失效;2)训练的Q函数的微小变化会导致策略的巨大改变,使算法不易收敛. Nature DQN的Experience replay机制能够解决观察序列的相关性问题. Experience replay机制首先将探索环境中的数据存储起来,然后从存储的数据中随机选择样本以更新深度神经网络的参数. Target DQN迭代式更新的方式[9]进一步减小了数据的关联性,有

$$\min(r_j+\gamma\max_{a'}\tilde{Q}(s_{j+1},a';\theta^-)-Q(s_j,a_j;\theta))^2.$$

该方法采用深度Q网络的参数延迟更新的方式降低Q网络抖动对训练的影响,同时减小了Q网络和Target Q网络的相关性.

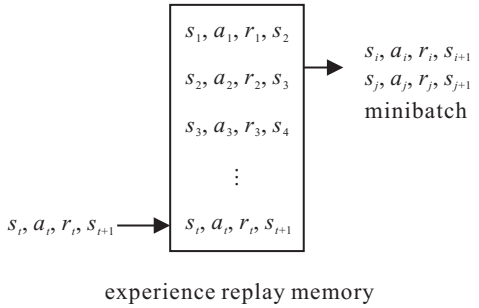

图1 经验回放机制

Experience replay机制使强化学习Agent能够记住并回放过去的经验,通过与最近的序列混合能够破坏观察序列的相关性. 然而,随机化策略简单地在记忆回放缓存中采样,而不管其意义,导致训练的效率不高. Schaul等[12]提出了Prioritized replay的采样策略来改进DQN,实现了对重要记忆单元的优先回放,并取得了更好的效果. 然而,Prioritized replay采用二叉堆优先级队列存储带有优先级的记忆单元,在加入优先级的同时引入时间复杂度,二叉堆结构查找最大优先级的记忆单元时间复杂度为$O(1)$,更新优先级队列的时间复杂度是$O(\log N)$[12]. 本文提出了一种基于优选缓存的Experience replay memory构建方法,在记忆单元进入队列时可根据评估的重要性指标计算采样概率,因此不再需要更新优先级,查找的时间复杂度为$O(1)$,解决了在回放中涉及的选择哪些记忆单元进行回放的问题. 在样本选择时间复杂度的比较上,与Prioritized replay方法相比由$O(\log N)$减小为$O(1)$.

2 基于优选缓存方法的DQN学习算法

为解决观察序列相关性和算法抖动不易收敛问题而采用的经验回放机制在运行过程中回放涉及两

个问题:一是选择哪些观察序列进行存储;二是选择哪些观察序列进行回放. 优选缓存机制主要针对第1个问题进行研究.

2.1 基于TD-error的重抽样优选机制

神经科学的研究表明,海马体对于动物的某些记忆和学习功能具有核心作用,先前的经验会在需要时被回放. 海马体通过对未探索空间构建奖赏序列构建记忆回放缓存,与奖赏有关的序列重播频率更高[16-17]. 与此类似的是值迭代这样的规划算法可以通过恰当的排序,以优先更新的方式获取更高的效率,TD-error提供了衡量这些优先级的一种方式[18-19].

优先缓存的一个核心组成部分是评判每个记忆单元优先的重要性. 理想的重要性度量标准是,RL Agent能够从当前状态下的记忆单元中学习到优先级的重要性. 然而,这个度量值并不能直接访问到,比较合理的替代量是采用记忆单元的TD error δ来表示该重要性. 这种度量值非常适合增量的RL算法、SARSA或Q-learning,因为这些算法已经计算了TD-error. 在算法更新的过程中,具备最大$|\delta|$的记忆单元将以更大的概率从缓存中进行回放,然后对这些记忆单元依据$|\delta|$值按权重进入记忆缓存,进行Q网络的更新,记忆单元进入记忆缓存的概率为

$$P(i)=\frac{p_i}{\sum\limits_k p_k}, \tag{8}$$

其中$p_i=|\delta_i|$. 从重抽样的结果中可以看到,每一个样本的消除、存储操作都是由基于TD误差的正则化权重决定的,引入重抽样后,权重低的记忆单元以一定概率被淘汰,从而限制了训练模型的退化现象. 重抽样后,记忆缓存中的记忆单元不再独立,经过若干次迭代后,具有高权重的记忆单元将被多次复制,而具有较低权重的记忆单元将逐渐消失,使记忆缓存中的训练集太小或不充分.

2.2 优选机制的改进

如第2.1节所述,算法导致的训练集太小或不充分的主要原因是基于TD误差的贪婪优选缓存机制有几个问题: 1) 为避免在整个缓存单元中扫描更新带来的计算代价,TD误差仅仅更新被回放的记忆单元,这样带来的一个后果是,带有低TD误差的记忆单元很有可能长时间因为无法加入记忆缓存而不能被回放; 2) 在初始深度Q网络没有训练好的情况下,估计误差较大; 3) 贪婪优选可能导致训练集更新较慢,误差收缩慢,初始的具有高TD误差记忆单元回放概率偏高,缺乏多样性使得该训练容易过估计.

为了解决上述问题,通过一种自适应采样方法,引入系数μ和偏移β,结合贪婪优选和随机选择的优点,确保记忆缓存的更新概率在优先级上是单调的. 与此同时,在Q网络没有训练好的情况下,一方面保证所有的记忆单元能够以极高的概率更新记忆缓存;另一方面,保证低优先级的记忆单元以一定概率更新记忆缓存. 随着Q网络逼近值函数,优先级在采样中的权重系数增大. 定义回放缓存采样记忆单元的概率为

$$P(i)=\mu\times\frac{p_i^{\alpha}}{\sum\limits_k p_k^{\alpha}}+\beta, \tag{9}$$

其中$P(i)$为转移i的优先级. 指数α决定使用多少优先级,当α等于0时是均匀的情况. 第1种方法是直接采用$|\delta_i|$成比例更新记忆单元的优先级,$p_i=|\delta_i|+\varepsilon$,其中$\varepsilon$为一个正的常数,防止在TD误差逼近0的情况下,记忆缓存采样该记忆单元的概率逼近0. 第2种方法为了减小$|\delta_i|$差异导致的不均衡采样现象,采用基于排行的优先级计算方法$p_i=1/\mathrm{rank}(i)$,其中$\mathrm{rank}(i)$为回放单元根据$|\delta_i|$误差排行的转移i的排行. 在这种情况下,P是参数为α的指数分布. 这两种分布在$|\delta_i|$上都是单调的,但是后者对离群记忆单元更鲁棒,因为其对TD-error不敏感. 两种计算优先级的方式与均匀随机采样相比都是有较大优势的.

另外,对于rank-base的优先级算法改进而言,假设minibatch的容量为$N_{\mathrm{minibatch}}$,令$k=N_{\mathrm{minibatch}}$,采用k段的等概率模型代替概率密度函数,分段边界仅当N或者α改变时才被重新计算. 在算法运行时,首先从记忆缓存中采样一个片段,然后从片段中均匀采样记忆单元. 这种方法能够较好地与基于minibatch的算法结合. 选取minibatch的大小为k,从每个片段中选取一个记忆单元,这是一种分层抽样方法,这种方法能够获取额外的优势(平衡了minibatch的优势,确定会有一个高的$|\delta_i|$和一个中等大小的$|\delta_i|$),比例的改进也是不同的. 下面给出基于重抽样优选缓存机制的Target DQN算法改进伪代码.

1) 初始化回放记忆缓存D容量为N,minibatch容量为M

2) 初始化Q网络,使用随机化参数θ

3) 初始化目标网络$\tilde{Q}$,权重$\theta^{-}=\theta$

4) 初始化参数μ,α,β

5) 重复(对每个回合episode)

6)　初始化状态s_t

7)　重复(对回合中的每一步)

8)　　以概率ε随机选择一个动作a_i

9) 否则选择$a_i = \mathrm{argmax}_a Q(s_i, a; \theta)$

10) 执行动作a_i后,观察环境得到新状态s_{i+1}和奖赏$\boldsymbol{r_i}$

11) 计算TD误差

$$\delta_i = R_i + \gamma Q_{\mathrm{target}}(S_i, \mathrm{argmax}_a Q(S_i, a)) - Q(S_{i-1}, A_{i-1})$$

12) 计算重要性采样权重$w_i = |\delta_i| + \epsilon$

13) 计算记忆单元i的采样概率

$$P(i) = \mu \times \frac{w_i^\alpha}{\sum\limits_k w_k^\alpha} + \beta$$

14) 设置$s_{i+1} = s_i, a_i, s_i$,以概率$P(i)$将(s_i, a_i, r_i, s_{i+1})数据存储于记忆单元D

15) 从D中根据采样概率采样M个记忆单元minibatch

16) 设置i目标标签

$$y_i = \begin{cases} r_i, \text{ 下一状态结束;} \\ r_i + \gamma \mathrm{max}_{a'} \tilde{Q}(s_{i+1}, a'; \theta^-), \text{ 其他} \end{cases}$$

17) 使用损失函数$(y_j - Q(s_j, a_j; \theta))^2$随机梯度下降更新网络参数$\theta$

18) 每C步将$\tilde{Q}$与Q同步$\tilde{Q} = Q$

19) 结束循环

20) 结束循环

21) 输出策略π

除此之外,对于其他集中算法的改进与Target DQN的改进方法类似,均利用基于优选缓存的方法对记忆缓存的形成进行改进.

3 实验分析

3.1 实验环境选择及参数设置

为验证算法及其改进的有效性,实验环境为OpenAI Gym[20], TensorFlow 0.8, Python 2.7,以OpenAI Gym经典控制问题中的CartPole-v0实验环境为验证对象. 使用3种算法作为baseline,分别为Nature DQN算法[6]、Sarsa算法[21]、double DQN算法[11]. 这些对比算法都具备记忆缓存回放机制,所有的经验记忆单元被存储在一个滑动记忆回放缓存中,缓存的大小为5 000,算法从记忆缓存中采样32个记忆单元形成minibatch,当4个新的记忆单元进入记忆缓存之后,将会更新minibatch.

深度神经网络的输入包括状态值s、动作值a. 最直接的模型构建方式是输入s、a,输出Q值,但这样对于每个a都要遍历一遍网络. 在本实验深度神经网络的构建中,以状态s作为输入,输出层则是每个a对应的Q值. 这种做法的优点是只要输入s,前向传播一遍即可获取所有a的Q值,毕竟a的数量有限. 梯度下降优化选择Adam算法,minibatch容量为32,经验回放缓存容量为5 000,训练时采用ε贪心算法选择动作,ε随着训练的进行而不断减小,ε的初始值为0.5,最小值为0.01. $\varepsilon = \varepsilon -$(初始值 − 最小值)/10 000,折扣因子$\gamma = 0.9$.

深度神经网络的输入为系统状态. 当构建第1个深度神经网络时,首先构建第1个卷积层,该卷基层有20个滤波器,此时必须使用状态维数(state_dim)作为通道数量,因为state_dim是前1层的输出size;然后构建第2层卷积层,第2层接收第1层的输出作为输入,因为前1层的输出size是32,所以通道数量为32,因为输出的是每一个动作的Q值,滤波器数量为动作的维数(action_dim),所以输出的size是动作的维数,在构建的网络中,每个隐藏层的神经元与输入层state_dim(状态的维数)相连,使用ReLU激活函数;之后,构建一个size为action_dim的全连接层,输出每个动作的Q值.

3.2 实验结果及分析

与监督学习不同,深度强化学习没有训练数据集和验证数据集,很难在线评估算法的训练情况. 因此,训练效果的评估主要有两种方式,一是使用奖赏值,网络训练一定周期后,平均奖赏值越大,表明效果越好; 二是训练的Q网络越快稳定,表明算法收敛性越好. 由于在训练期间,网络参数的微小变化可能会引起策略选择的巨大变化,平均奖赏值的噪声较大,导致训练效果有较大震荡. 图2显示了Nature DQN、DSarsaN、double DQN算法、基于优先缓存机制的改进(PER)和基于重采样的优选缓存经验回放机制

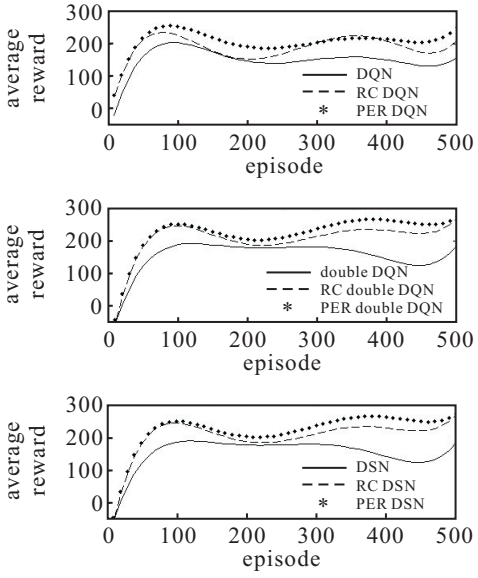

图2 几种算法的平均奖赏值对比

(RC)在训练期间的平均奖赏值.

表1为按照PER方法和RC方法对DQN、DDQN、DSN几种算法改进后,在测试环境中运行10次的平均奖赏值的均值和方差. 可以看出,在性能上,基于RC算法的改进在DQN和DDQN算法上分别比基准算法提升了35.7%和42.2%;在对DSN的改进上,基于PER的改进取得了最优的效果,但基于RC的改进与其并没有明显劣势,分别比基准算法提升了24.8%和22.9%.

表1　实验效果对比

均值/均方差	R	RPER	RRC
DQN	(154.39, 59.57)	(189.59, 72.48)	(**209.46**, 54.97)
DDQN	(168.71, 71.81)	(220.62, 59.05)	(**239.17**, 51.84)
DSN	(201.96, 60.66)	(**251.74**, 52.18)	(247.46, 46.05)

对几种算法的实验效果在显著性水平Alpha = 0.05的情况下进行显著性检验,R为基准算法的平均奖赏值,RRC为基于RC改进的算法的平均奖赏值,RPER为基于PER改进的算法的平均奖赏值,原假设均为相等. 如表2所示,对基准算法改进的检验均拒绝原假设,接受备择假设. 而对于基于RC算法改进和基于PER算法改进效果检验的结果均接受原假设. 因此可以认为,在以episode为计量的实验效果上两者没有显著性区别,基于RC的改进取得了不弱于基于PER改进的性能.

表2　对实验效果的假设检验

(H, SIGNIFICANCE)		备择假设		
		$R < R_{RC}$	$R < R_{PER}$	$R_{RC} \neq R_{PER}$
基准算法	DQN	(1, 0.006)	(1, 0.002 54)	(0, 0.206 0)
	DDQN	(1, 0.000 8)	(1, 0.000 8)	(0, 0.233 8)
	DSN	(1, 0.003 3)	(1, 0.000 44)	(0, 0.835 1)

与平均奖赏值相比,动作值函数Q提供了衡量来自任何给定状态的策略而获得的折扣报酬估计. 实验通过在训练开始之前运行随机策略来收集固定的一组状态,并且跟踪这些状态最大Q的平均值. 如图3所示,动作值函数Q的变化与平均奖赏值相比呈现更加平稳的收敛趋势(不同算法动作值函数的大小并不能表明算法优劣,但其趋势能表明算法的收敛速度). 这表明,实验所采用的方法改进以及基础对比算法都能通过强化学习的训练,使采用随机梯度下降的深度神经网络获取稳定的收敛.

如文献[12]所述,PER算法在提高学习效果的同时引入了时间复杂度,因此以时间为横轴对比PER算法和RC算法的改进,如图4所示. 由图4可见,RC对Nature DQN、DSarsaN、double DQN几种算法的改进能够更快地收敛,与基准算法相比,分别提升了72.6%、67.2%、57.1%,与PER算法比,分别提升了61.3%、45.9%、30%.

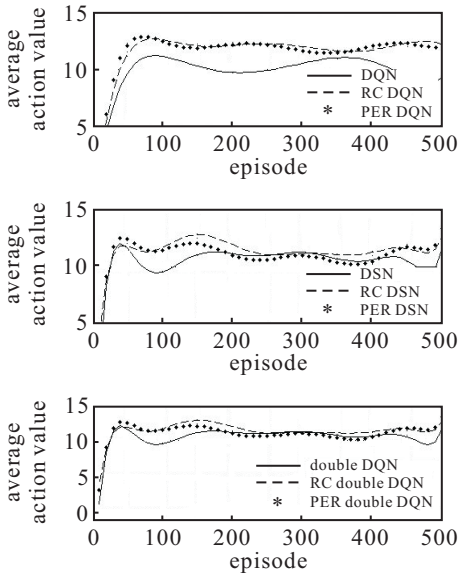

图3　算法的动作值函数变化趋势对比

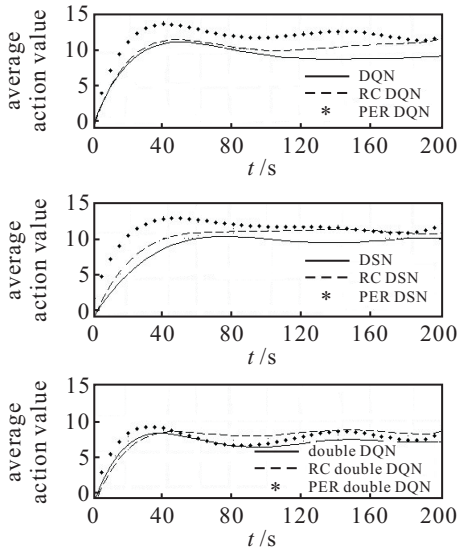

图4　几种算法收敛时间比较

实验表明: 在episode的性能比较上,对DQN、DDQN、DSN的改进与基准算法相比,性能分别提升了35.7%、42.2%、22.9%;在相同时间的对比上,基于RC的改进优于基于PER的改进,收敛速度得到提升. 以收敛前TD-error均值作为衡量样本的质量标准,如表3所示,采用优选回放机制后,TD-error均值明显变大,样本对于提升训练速度效果明显.

表3　实验效果对比

	R	RPER	RRC
DQN	(62, 89, 0.041)	(44, 45, 0.102)	(17, 44, 0.104)
DDQN	(61, 61, 0.032)	(37, 80, 0.092)	(20, 79, 0.099)
DSN	(49, 50, 0.052)	(30, 25, 0.112)	(21, 30, 0.132)

4 结 论

本文提出了一种基于重采样的优选缓存经验回放机制,可以使得深度强化学习从样本中以更大的概率选取更好的样本用于训练深度Q网络,同时提出了避免训练集坍塌的改进算法. 提出了几种典型深度强化学习算法基于优选缓存回放机制的改进方法,并且在经典控制问题上进行了实验. 研究结果表明,在优选缓存回放机制下,算法都能稳定地训练并收敛,与几种典型的深度强化学习算法相比,训练效率均有较大的提升.

参考文献(References)

[1] Mozer S, M C, Hasselmo M. Reinforcement learning: An introduction[J]. IEEE Trans on Neural Networks, 2005, 16(1): 285-286.

[2] Riedmiller M. Neural fitted q iteration – first experiences with a data efficient neural reinforcement learning method[C]. Proc of the 16th European Conf on Machine Learning. Portugal: ECML, 2005: 317-328.

[3] Whiteson S, Stone P. Evolutionary function approximation for reinforcement learning[J]. J of Machine Learning Research, 2006, 7(5): 877-917.

[4] Preux P, Girgin S, Loth M. Feature discovery in approximate dynamic programming[C]. Adaptive Dynamic Programming & Reinforcement Learning. Nashville: IEEE, 2009: 109-116.

[5] Degris T, Pilarski P M, Sutton R S. Model-free reinforcement learning with continuous action in practice[C]. American Control Conf. New York: IEEE, 2012: 2177-2182.

[6] Mnih V, Kavukcuoglu K, Silver D, et al. Playing atari with deep reinforcement learning[EB/OL]. [2017-3-16], https://arxiv.org/abs/1312.5602.

[7] Bianchi R A C, Celiberto L A, Santos P E, et al. Transferring knowledge as heuristics in reinforcement learning: A case-based approach[J]. Artificial Intelligence, 2015, 226(13): 102-121.

[8] Peng J, Williams R J. Incremental multi-step q-learning[J]. Machine Learning, 1996, 22(1): 283-290.

[9] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning[J]. Nature, 2015, 518(7540): 29-33.

[10] Fischer A, Igel C. An introduction to restricted boltzmann machines[C]. Iberoamerican Congress on Pattern Recognition. Heidelberg: Springer, 2012: 14-36.

[11] Van Hasselt H, Guez A, Silver D. Deep reinforcement learning with double Q-learning[EB/OL]. [2017-3-16], https://arxiv.org/abs/1509.06461.

[12] Schaul T, Quan J, Antonoglou I, et al. Prioritized experience replay[EB/OL]. [2017-3-16], https://arxiv.org/abs/1511.05952.

[13] Bellemare M G, Ostrovski G, Guez A, et al. Increasing the action gap: New operators for reinforcement learning[EB/OL]. [2017-3-16]. https://arxiv.org/abs/1512.04860.

[14] Lillicrap T P, Hunt J J, Pritzel A, et al. Continuous control with deep reinforcement learning[EB/OL]. [2017-3-16]. https://arxiv.org/abs/1509.02971.

[15] Mnih V, Badia A P, Mirza M, et al. Asynchronous methods for deep reinforcement learning[EB/OL]. [2017-3-16]. https://arxiv.org/abs/1602.01783.

[16] Atherton L A, Dupret D, Mellor J R. Memory trace replay: The shaping of memory consolidation by neuromodulation[J]. Trends in Neurosciences, 2015, 38(9): 560-570.

[17] Mcnamara C G, Tejerocantero Á, Trouche S, et al. Dopaminergic neurons promote hippocampal reactivation and spatial memory persistence[J]. Nature Neuroscience, 2014, 17(12): 1658-1673.

[18] Andrew W Moore, Christopher G Atkeson. Prioritized sweeping: Reinforcement learning with less data and less time[J]. Machine Learning, 1993, 13(1): 103-130.

[19] Andre D, Friedman N, Parr R. Generalized prioritized sweeping[C]. Conf on Advances in Neural Information Processing Systems. Boston: MIT Press, 1997: 1001-1007.

[20] Greg B, Vicki C, Ludwig P, et al. OpenAI Gym[EB/OL]. [2016-12-16]. https://arxiv.org/abs/1606.01540.

[21] Hu W. Double sarsa and double expected sarsa with shallow and deep learning[J]. J of Data Analysis and Information Processing, 2016, 4(4): 159-176.

（责任编辑：郑晓蕾）