基于线性动态跳帧的深度双 Q 网络

陈 松[1] 章晓芳[1),2)] 章宗长[1),2)] 刘 全[1),3)] 吴金金[1)] 闫 岩[1)]

[1)](苏州大学计算机科学与技术学院 江苏 苏州 215006)

[2)](南京大学计算机软件新技术国家重点实验室 南京 210023)

[3)](吉林大学符号计算与知识工程教育部重点实验室 长春 130012)

摘 要 深度 Q 网络模型在处理需要感知高维输入数据的决策控制任务中性能良好.然而,在深度 Q 网络及其改进算法中基本使用静态的跳帧方法,即动作被重复执行固定的次数.另外,优先级经验重放是对均匀采样的一种改进,然而目前各个研究仅将样本的时间差分误差作为评价优先级的标准.针对这两个问题,该文提出一种基于线性动态跳帧和改进的优先级经验重放的深度双 Q 网络.该算法使得跳帧率成为一个可动态学习的参数,跳帧率随网络输出 Q 值的大小线性增长,Agent 将根据当前状态和动作来动态地确定一个动作被重复执行的次数,并利用经验池中样本的每个动作的跳帧率和样本的时间差分误差共同决定样本的优先级.最后在 Atari 2600 游戏中进行实验,结果表明该算法相比于传统动态跳帧和优先级经验重放算法具有更优的效果.

关键词 深度强化学习;深度 Q 网络;动态跳帧;优先级经验重放

中图法分类号 TP18 **DOI 号** 10.11897/SP.J.1016.2019.02561

Deep Double Q-Network Based on Linear Dynamic Frame Skip

CHEN Song[1)] ZHANG Xiao-Fang[1),2)] ZHANG Zong-Zhang[1),2)] LIU Quan[1),3)] WU Jin-Jin[1)] YAN Yan[1)]

[1)](*School of Computer Science and Technology, Soochow University, Suzhou, Jiangsu* 215006)

[2)](*State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing* 210023)

[3)](*Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun* 130012)

Abstract Deep Q-Network is able to perform human-level control for handling problems requiring both rich perception of high-dimensional raw inputs and policy control. However, the current state of the art architectures like Deep Q-Network and it improved algorithms adopt a traditional framework with a static frame skip rate, where the action output from the network is repeated for a fixed number of frames regardless of the current state. Although Dynamic Frame skip Deep Q-Network uses a dynamic frame skip rate, it doubles the number of nodes in the network output layer with a frame skip rate of 4 or 20. Such settings may cause an increase in the amount of computation of the network, and cause bad actions to be performed multiple times, thereby affecting the efficiency of learning. In addition, an important technique in Deep Q-Network is the use of an experience replay mechanism. The traditional method of uniformly sampling samples ignores the importance of samples. In order to increase the sampling rate of important samples, the prioritized experience replay is an improvement on uniform sampling, using only the temporal difference error of a sample as the criterion of evaluation priority. However, such priority evaluation

收稿日期:2018-07-14;在线出版日期:2019-05-31.本课题得到国家自然科学基金项目(61472262,61502329,61772355,61876119)、江苏省自然科学基金面上项目(BK20181432)、吉林大学符号计算与知识工程教育部重点实验室基金项目(93K172014K04,93K172017K18)、苏州市重点产业技术创新-前瞻性应用研究项目(SYG201807)资助.**陈 松**,硕士研究生,主要研究方向为强化学习、深度学习和深度强化学习.E-mail:20174227010@stu.suda.edu.cn.**章晓芳**(通信作者),博士,副教授,中国计算机学会(CCF)专业会员,主要研究方向为强化学习、软件测试、深度强化学习.E-mail:xfzhang@suda.edu.cn.**章宗长**,博士,副教授,主要研究方向为强化学习、智能规划和多智能体系统.**刘 全**,博士,教授,博士生导师,主要研究领域为智能信息处理、自动推理和机器学习.**吴金金**,硕士研究生,主要研究方向为深度强化学习.**闫 岩**,硕士研究生,主要研究方向为深度强化学习.

criterion only considers the temporal difference error of the sample, and there may be other factors that affect the priority of the sample. In this paper, we propose a new algorithm: Deep Double Q-Network based on Linear Dynamic Frame Skip and Improved Prioritized Experiential Replay (LDF-IPER-DDQN in short). The value of the frame-skip rate increases linearly with the magnitude of the network output Q value, which allows Agent to dynamically select the number of times an action is repeated based on the current state and action. For the action with the largest Q value, the maximum frame skip rate is given to this action. In contrast, the action with the smallest Q value is given the minimum frame skip rate. In this way, the frame skip rate becomes a dynamic learnable parameter. Furthermore, the value of the frame skip rate for each action in the experience pool is considered as another factor to evaluate the sampling priority. The priority of the sample is determined by the frame skip rate of each action of the sample in the experience pool and the temporal difference error of the sample. As a sequence, if two transition samples have similar temporal difference errors, the transition samples with larger frame skip rates are replayed more frequently. In the experiment of this paper, we evaluate the performance of the new algorithm through eight challenging strategic games with sparse reward from the set of classic Atari 2600 games. They are Seaquest, Assault, Asterix, Q*bert, SpaceInvaders, Berzerk, BeamRider and Gopher. We evaluate the performance of the new algorithm during the training and testing phases. The experimental results show that LDF-IPER-DDQN performs better than some traditional deep reinforcement learning algorithms on these Atari 2600 games. It achieves better performance in terms of the average reward per episode and has better stability on these games.

Keywords deep reinforcement learning; Deep Q-Network; dynamic frame skip; prioritized experience replay

1 引 言

深度学习(Deep Learning,DL)是机器学习领域的研究热点之一[1].其基本思想是结合多层深度网络,例如神经网络[2]和卷积网络[3],并且使用非线性的激活函数变换来对数据进行表征学习.深度学习已在图像处理、语音识别、自然语言处理等领域取得了很大的突破[4-6].与深度学习不同,强化学习(Reinforcement Learning,RL)注重的是与环境交互进行学习,最终获得最优策略[7].深度学习和强化学习有着各自的特性,结合这两者就产生了深度强化学习(Deep Reinforcement Learning,DRL)[8-9].深度强化学习融合了深度学习的抽象表征能力和强化学习的序贯决策能力,可以帮助 Agent 在一些复杂的环境中更好地进行学习和决策.

Gym 软件包中 Atari 2600 游戏的画面具有高维度的原始特征[10],这对于 Agent 来说是个独特的挑战.Agent 不仅需要理解游戏的当前状态,还需要选择能够获得长期奖励的动作.Mnih 等人[11]结合深度学习中处理图像数据最常用的卷积神经网络和强化学习中求解最优动作值函数的 Q 学习算法,提出了一种深度 Q 网络模型(Deep Q-Network,DQN):首先利用卷积神经网络有效提取出游戏中状态的特征,然后用神经网络逼近状态动作对的值,可以泛化到很多没有经历过的状态.实验结果表明,DQN 在 Atari 2600 大部分游戏中的表现已经赶上甚至超过人类玩家的水平.

深度 Q 网络的成功让众多深度强化学习算法得到了广泛关注和研究.由于 Q 学习算法存在着高估 Q 值的问题,Van Hasselt 等人[12]提出一种深度双 Q 学习网络(Deep Double Q-Network,DDQN).为解决深度 Q 网络训练耗时大的问题,Nair 等人[13]提出一种并发式结构的 DQN.Hausknecht 等人[14]将具有记忆功能的长短期记忆单元 LSTM 加入到深度 Q 网络中,提出一种带有循环网络的深度 Q 网络结构(Deep Recurrent Q-Network,DRQN).在 DRQN 的基础上,Sorokin 等人[15]提出了基于视觉注意力机制[16]的 DRQN,使得 Agent 在每个时刻都能够自适应地将注意力集中在对提升奖赏有促进作

用的图像区域. 翟建伟等人[17]在循环神经网络的基础上使用双层门限循环单元，提出了一种基于视觉注意力机制的深度循环 Q 网络. Junhyuk 等人[18]提出基于内存的深度 Q 网络. Ziyu 等人[19]则根据动作的优势函数[20-21]提出基于竞争的深度 Q 网络. 最近，Hessel 等人结合了众多基于 DQN 的改进算法（包括基于分布的 DQN[22]、基于噪声的 DQN[23]等算法）形成 Rainbow 算法[24].

深度 Q 网络中的一个重要参数是跳帧率，反映了 Agent 重复执行选定动作的次数. 在深度 Q 网络及其改进算法中大多使用了静态的跳帧方法，即动作被重复执行固定的次数. Aravind 等人[25]则提出一种动态跳帧的深度 Q 网络算法（Dynamic Frame skip Deep Q-Network，DF-DQN）. DF-DQN 将网络输出层节点数量扩大一倍，跳帧率为 4 或者 20. 然而这样的设置会导致网络的计算量增长一倍，并且可能会导致不好的动作被执行多次，从而影响学习的效率.

此外，深度 Q 网络中的一项重要技术是采用了经验重放机制. 传统的均匀抽取样本的方式忽略了样本的重要性，为了提高重要样本的抽样率，Schaul 等人[26]提出了基于优先级经验重放机制的深度双 Q 网络算法（Prioritized Experience Replay Deep Double Q-Network，PER-DDQN）. 样本的优先级仅由样本的时间差分（Temporal Difference，TD）误差的大小决定. 但是这样的优先级评价标准只考虑到样本的时间差分误差，可能还存在其它影响样本优先级的因素.

针对上述问题，本文提出了一种线性动态跳帧方法. 跳帧率的值随网络输出 Q 值大小呈线性增长，这允许 Agent 根据当前状态和动作来动态地确定当前的动作被重复的次数，使得跳帧率成为一个可动态学习的参数，从而在不增加网络计算量的前提下，高效准确地确定跳帧率的值，此外，考虑跳帧率对样本优先级的影响，样本优先级由时间差分误差和动作跳帧率共同决定. 这样可以让具有较高时间差分误差且动作需要重复执行更多次的转移样本重放更频繁. 本文提出的基于线性动态跳帧和改进的优先级经验重放的深度双 Q 网络算法（Deep Double Q-Network based on Linear Dynamic Frame Skip and Improved Prioritized Experiential Replay，记为 LDF-IPER-DDQN）将应用于 Atari 2600 的 8 个经典游戏中，并和其它相关算法进行对比实验. 实验结果表明，我们的方法可以有效提升 Agent 在 Atari 2600 游戏上的表现.

2　相关工作

2.1　强化学习

在一个标准的强化学习设置中，Agent 在多个离散的时间步中与环境进行交互[7]. 在每个时间步，Agent 接收一个状态 s_t，并根据策略 π 从可能的动作集合 A 中选取一个动作. 策略 π 是从状态到动作的映射. 作为反馈，Agent 执行这个动作进入到下一个状态，并得到一个标量的奖赏 r_t. 回报 R_t 指的是从当前时间步开始一直到情节结束所得到的累积折扣奖赏，定义为

$$R_t=\sum_{k=0}^{\infty}\gamma^k r_{t+k} \tag{1}$$

其中 γ 为折扣因子，$\gamma\in(0,1]$. Agent 的目标就是最大化每个状态的期望累积奖赏.

动作值函数表示在状态 s 根据策略 π 选择动作 a，并一直遵循策略 π 到情节结束得到的期望回报，记为

$$Q^{\pi}(s,a)=E[R_t\mid s_t=s,a_t=a,\pi] \tag{2}$$

类似的，值函数表示在状态 s 遵循策略 π 到情节结束得到的期望回报，记为

$$V^{\pi}(s)=E[R_t\mid s_t=s] \tag{3}$$

对于所有状态，如果一个策略 π 的期望回报大于或等于其它策略的期望回报，那么这个策略 π 就被称为最优策略 π^*.

$$Q^*(s,a)=\max_{\pi}Q^{\pi}(s,a) \tag{4}$$

最优 Q 值函数也遵循贝尔曼最优方程[7]：如果在下一状态 s_{t+1}，关于所有动作的 Q 值函数都已知，那么最优策略就选择可以最大化期望回报的动作.

$$Q^*(s,a)=E_{s_{t+1}}[r+\max_{a_{t+1}}Q^*(s_{t+1},a_{t+1})\mid s,a] \tag{5}$$

在经典强化学习中，不断地迭代贝尔曼方程，动作值函数最终会收敛并得到最优策略. 然而在解决一些实际问题时，状态空间往往很大，通过迭代的方式求解最优策略的方法通常是不可取的. 因此，在求解最优策略之前，首先需要对状态空间进行泛化，再使用一些函数逼近的方法（如神经网络、决策树等）来评估状态值函数和状态动作值函数.

在强化学习中，研究人员通常构造线性函数逼近器来近似地表示状态动作值函数，即 $Q(s,a\mid\theta)\approx Q^*(s,a)$. 当然也可以使用神经网络等非线性函数逼近器来表示值函数，但是神经网络往往会导致算法的性能很不稳定，Q 值函数经常不收敛，这一直是强化学习中的一个难题.

2.2 深度 Q 网络

DQN[11]在一定程度上解决了使用非线性函数逼近器带来的问题.不同于传统的 Q 学习算法,DQN 使用了 2 种关键技术:评估网络和目标网络分离技术,经验重放技术.

评估网络和目标网络分离是指 DQN 使用 2 个不同的网络:评估网络用来近似表示当前状态动作对的值函数,目标网络用来计算目标值函数.DQN 在每次迭代 i 中优化以下损失函数序列:

$$L_i(\theta_i)=E_{s,a,r,s_{t+1}}[(y_i^{\mathrm{DQN}}-Q(s,a|\theta_i))^2] \quad (6)$$

其中:

$$y_i^{\mathrm{DQN}}=r+\gamma\max_{a_{t+1}}Q(s_{t+1},a_{t+1}|\theta^-) \quad (7)$$

θ 表示评估网络参数,θ^- 表示目标网络参数.$Q(s,a|\theta)$ 表示当前网络的输出,用来评估当前状态动作对的值函数.$Q(s,a|\theta^-)$则表示目标值网络的输出,用来计算目标值.

DQN 使用 Q 学习算法来在线学习网络 $Q(s,a|\theta)$ 的参数.当用梯度下降在线更新评估网络参数 θ 时,将目标网络的参数 θ^- 保持一定的时间步不变.当前评估网络的参数 θ 是不断更新的,每经过 N 次迭代,将当前评估网络的参数 θ 复制给目标网络.通过优化上述目标值函数和当前值函数之间的均方误差 $L_i(\theta_i)$来更新参数.具体地,使用随机梯度下降算法来更新网络的参数 θ,对损失函数求偏导,得到以下梯度:

$$\nabla_{\theta_i}L_i(\theta_i)=E_{s,a,r,s_{t+1}}[(y_i^{\mathrm{DQN}}-Q(s,a|\theta_i))\nabla_{\theta_i}Q(s,a|\theta_i)] \quad (8)$$

经验重放技术指从经验池中等概率地抽取样本.在 DQN 学习训练的过程中,Agent 将很多个情节经历的经验,记为 $e_t=(s_t,a_t,r_t,s_{t+1})$存放在经验池 $D_t=(e_1,e_2,\cdots,e_t)$中.训练 Q 网络时,从经验池中随机选取批量的样本来更新网络的参数.计算损失函数如下:

$$L_i(\theta_i)=E_{(s,a,r,s_{t+1})\sim U(D)}[(y_i^{\mathrm{DQN}}-Q(s,a|\theta_i))^2] \quad (9)$$

经验重放技术通过在多个更新中重复使用经验样本来提高数据效率并减小方差.此外,从经验池中均匀采样的方式降低了更新中所使用的样本之间的相关性.

DDQN 算法[12]能有效解决 DQN 算法存在高估 Q 值的问题.DDQN 和 DQN 的网络结构是一致的,两者的不同之处在于:DDQN 采用强化学习中 Double Q-learning 方式更新,即使用如下公式计算目标值:

$$y^{\mathrm{DDQN}}=r+\gamma Q(s_{t+1},\arg\max_{a_{t+1}}(s_{t+1},a_{t+1}|\theta)|\theta^-) \quad (10)$$

2.3 动态跳帧

跳帧率是指 Agent 选择的动作被重复执行的次数,跳帧率不宜过高或者过低.如果跳帧率过低,Agent 的决策频率会很高,导致策略变化得很快,且每个决策都涉及运行卷积神经网络的计算,将极大降低游戏的速度.反之,如果跳帧率过高,会导致在一个情节训练的时间过少.

不同于 DQN 中使用固定值为 4 的跳帧方法,在 DF-DQN 模型[25]中,跳帧率根据当前状态和动作变化,Agent 决定是否基于输入图像采取较长的重复动作序列,即较高的跳帧率,取值为 20;或采取较短的重复动作序列,即较低的跳帧率,取值为 4.

具体而言,DF-DQN 将 DQN 网络结构中的最后一层全连接层的神经元数量加倍,即具有 1024 个神经元.给定一个状态 s,DF-DQN 输出的是一组离散的动作 Q 值函数$\{Q(s,a_1),\cdots,Q(s,a_{2|A|})\}$,其中 $|A|$表示动作的个数,其 Q 值函数的数量是 DQN 网络输出的 2 倍.如果 Agent 选择的动作小于$|A|$,则跳帧率为 4;如果选择的动作值在$|A|+1$ 和 $2|A|$之间,则跳帧率为 20.这里,选择$|A|+1$ 到 $2|A|$之间的动作和选择 1 到$|A|$之间的动作的执行效果是一致的.

2.4 优先级经验重放

针对传统的经验重放机制无法区分样本重要性的问题,优先级经验重放机制[26]将每个样本的时间差分误差项作为评价优先级的标准.如果样本的时间差分的误差项绝对值越大,其成为训练样本的概率也越高.

在抽样过程中,优先级经验重放还使用了随机比例化和重要性采样权重这两种技巧.

随机比例化是介于纯贪心优先级和均匀抽样之间的一种抽样方式.这种方式确保了样本被采样的概率在优先级上是单调的,且确保了最低优先级的样本被采样的概率大于零.具体来说,样本 i 被采样的概率定义为

$$P(i)=\frac{p_i^\alpha}{\sum_j p_j^\alpha} \quad (11)$$

其中 p_i是样本的优先级,$p_i=|\delta_i|+\varepsilon_0$,$p_i>0$,$\varepsilon_0$是一个很小的正值,保证样本的优先级不为 0.参数 α 决定优先级有多少被利用,$0\leq\alpha\leq1$.当 $\alpha=0$ 时,这种采样方式就变成了随机均匀采样方式.此外,还有一种间接的优先级方式,即基于排序的优先级,$p_i=1/rank(i)$,其中 $rank(i)$是样本 i 根据时间差分误差项$|\delta|$降序排列后的等级.

优先级经验重放会带来增大偏差的问题，为了解决这个问题，使用重要性采样权重来修正带来的偏差. 重要性采样权重定义为

$$w_i=\left(\frac{1}{N}\cdot\frac{1}{p(i)}\right)^{\beta} \tag{12}$$

其中 N 和 β 是超参数，N 表示经验池的大小，β 是一个常数且 $0\leq\beta\leq1$. 重要性采样权重都被归一化为 $1/\max_i w_i$. 在 Q 学习更新中，使用 $w_i\delta_i$ 替代 δ_i 去更新.

3　基于线性动态跳帧的深度双 Q 网络

本节将具体阐述 LDF-IPER-DDQN 算法，包括线性动态跳帧方法的实现流程，以及如何将线性动态跳帧和优先级经验重放进行有效地结合.

3.1　线性动态跳帧

为了弥补 DF-DQN 算法中网络计算量大和不能有效确定跳帧率的不足，本文提出线性动态跳帧方法. 在 LDF-IPER-DDQN 框架中，根据评估网络中输出的每个动作的 Q 值来动态决定其跳帧率. Q 值大小表明这个动作在当前状态的优劣. 对于具有最大 Q 值的动作，赋予该动作最大的跳帧率；具有最小 Q 值的动作则赋予最小的跳帧率. 其中，根据 DQN 网络中跳帧率参数值，最小跳帧率 $k_{\min}=4$. 其余动作的跳帧率随其 Q 值大小线性增长，Q 值越大，跳帧率越大.

定义集合 $A=\{a_1,a_2,\cdots,a_{|A|}\}$ 为合法动作集合，其中 $|A|$ 为合法的动作数量. 针对多组动作集合，存在 $|A|$ 的上界，记为 $A_{\max}$，则每个动作的跳帧率 k 定义为

$$k=k_{\min}+i\times([A_{\max}/|A|]) \tag{13}$$

其中 $i=\{0,1,2,\cdots,|A|-1\}$，为每个动作按 Q 值升序排列后的索引. 例如，Atari 2600 游戏环境中最多有 18 个合法动作，即 $A_{\max}=18$. 而在 Seaquest 游戏中，$|A|=18$，则该游戏中跳帧率的取值范围为 $[4,21]$.

具体而言：给定一个状态，LDF-IPER-DDQN 算法输出一组离散的 Q 值和跳帧率；然后，行为策略根据当前状态下每个动作的 Q 值确定 Agent 选择的动作和跳帧率，即 Agent 基于行为策略重复执行动作 a 共计 k 次. 在经验重放池中存放的经验信息既带有动作信息也带有跳帧率参数的信息，记为 $e_t=(s_t,(a_t,k_t),r_t,s_{t+1})$.

线性动态跳帧根据网络输出动作 Q 值的大小来线性动态地决定每个动作的跳帧率. 这种方式在不增加网络计算量的前提下，高效准确地计算出了每个动作需要重复执行的次数.

3.2　线性动态跳帧和优先级经验重放的结合

LDF-IPER-DDQN 算法中将样本的 TD 误差和样本动作跳帧率 k 共同作为评价样本优先级的标准. 在两个转移样本具有相近 TD 误差的情况下，研究希望具有较大跳帧率的转移样本更频繁地被重放. 这是因为，在游戏的某些状态下，选择的动作需要被重复执行较多的次数. 例如在 Seaquest 游戏中，潜艇没有氧气但是还在深海作战时，就需要不断地执行向上的动作. 在这样的转移样本中，向上动作的跳帧率较大. 为了让 Agent 快速地学习到在没有氧气的情况下，不断执行向上这一动作，从而学习该游戏的关键性策略，需要在设计样本优先级时考虑跳帧率的取值. 因此，定义转移样本的优先级 p 与转移样本的 TD 误差和跳帧率相关，具体如下：

$$p=|\delta|+\varepsilon_0+\varepsilon^{-}\cdot(k/k_{\max}) \tag{14}$$

其中 δ 是样本的 TD 误差；ε_0 是一个很小的正值以避免优先级为 0；ε^{-} 是一个超参数，用来控制让 $(k/k_{\max})$ 和 TD 误差具有相近的数量级，ε^{-} 随时间步的增长而衰减. 在抽样过程中，与基于优先级的深度双 Q 网络类似，算法仍采用了随机比例化和重要性采样权重这两项技术.

3.3　网络结构和算法描述

LDF-IPER-DDQN 的网络结构示意图如图 1 所示. 该网络由三个卷积层和两个全连接层构成；第

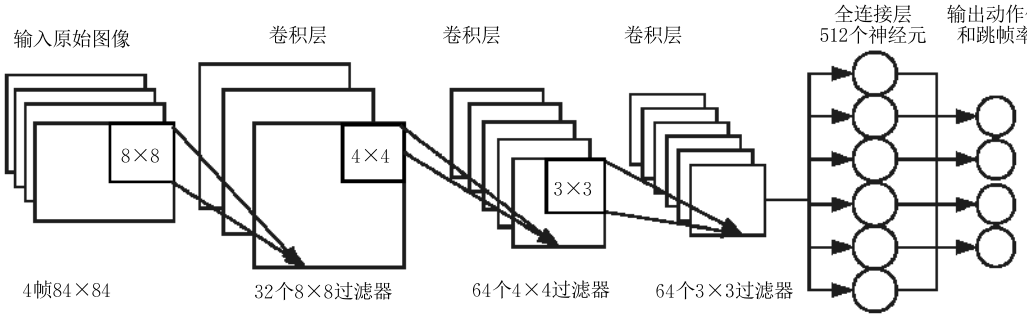


图 1　LDF-IPER-DDQN 网络结构示意图

一个卷积层为 32 个步幅为 4 的 8×8 的过滤器,第二个卷积层为 64 个步幅为 2 的 4×4 的过滤器,第三个卷积层为 64 个步幅为 1 的 3×3 的过滤器,最后隐藏层是由 512 个神经元组成的全连接线性层,输出层神经元数量和游戏合法动作数量相同.网络的输入是游戏的画面,输出是该游戏状态下每个动作的 Q 值和跳帧率.

LDF-IPER-DDQN 的实施过程如算法 1 所示.(1)输入各初始化超参数,Agent 在初始状态根据 ε-greedy 策略选择动作(a_0,k_0),并执行 a_0 动作 k_0 次,得到下一状态和奖赏;(2)将转移样本存入经验池中;(3)训练网络时,从经验池中以优先级方式采样,并使用随机比例化和重要性采样权重两种技巧,然后使用样本的时间差分误差和跳帧率来更新样本的优先级;(4)Agent 在每个时间步根据网络输出的 Q 值和跳帧率来确定需要执行的动作和对应执行的次数;(5)经过一定的时间步后,将当前评估网络的权重 θ 复制给目标网络.

算法 1. LDF-IPER-DDQN 算法.

输入:批量大小 n,重放周期 K,经验池内存大小 N,跳帧率 k,$4\leq k\leq 21$,指数参数 α 和 β,最大训练时间 T,学习率 η 等超参数.

输出:动作 Q 值,跳帧率 k.

1. 初始化经验池内存大小 N,增量 $\Delta=0$,初始样本采样概率 $p_1=1$.
2. 观察状态 s_0,根据 ε-greedy 策略选择动作(a_0,k_0).
3. FOR $t=1$ TO T:
4. 执行动作,观察状态 s_t,r_t,
5. 将转移样本$(s_{t-1},(a_{t-1},k_{t-1}),r_t,s_t)$以最大的优先级 $p_t=\max_{i<t}p_i$ 存放到经验池中.
6. IF $t\equiv 0 \bmod K$ THEN:
7. FOR $j=1$ TO n:
8. 从经验池中抽取样本:
$$j\sim P(j)=p_j^{\alpha}\Big/\sum_i p_i^{\alpha}$$
9. 计算重要性采样权重:
$$w_j=(N\cdot P(j))^{-\beta}/\max_i w_i$$
10. 计算样本 TD 误差:
$$\delta_j=r_j+\gamma_j Q_{\text{target}}(s_j,\arg\max_a Q(s_j,a))-Q(s_{j-1},a_{j-1})$$
11. 更新样本优先级:
$$p_j=|\delta_j|+\varepsilon_0+\varepsilon^{-}\cdot(k_j/k_{\max})$$
12. 计算梯度:
$$\Delta=\Delta+w_j\cdot\delta_j\cdot\nabla_\theta Q(s_{j-1},a_{j-1})$$
13. END FOR
14. 更新权重 $\theta=\theta+\eta\cdot\Delta$
15. 经过一定的时间步后,将权重 θ 复制给目标网络的参数 θ_{target}.
16. END IF
17. 根据策略选择动作$(a_t,k_t)\sim\pi_\theta(s_t)$,根据跳帧率 k_t 的值,让 Agent 重复执行 k_t 次.
18. END FOR

4 实验设置和评估

本节主要介绍实验平台、实验对比算法、实验参数设置和实验评估标准.

4.1 实验平台

本文基于 OpenAI 开发的 Gym[10] 实验平台进行实验.Gym 是一款用于研发和比较强化学习算法的工具包,包含了很多游戏平台,例如 Atari 2600,MuJoCo 等.游戏的类型涉及体育竞技类、桌游类和训练思维能力的战略性游戏.Gym 提供了很多 Atari 2600 游戏环境的接口,Gym 还提供了一个游戏处理层,通过识别累积得分和游戏是否结束,将每个 Atari 2600 游戏转换为标准强化学习问题.具体地,动作空间由操纵杆控制器定义的 18 个离散动作组成.每个时间步长的奖赏是通过游戏来定义的,通常通过得分或帧之间的差异来定义奖赏.

DQN 及其改进模型使用了深度 Q 学习算法来训练 Agent,然而在一些战略性游戏上,DQN 的性能并不很令人满意.为了提升 Agent 在这类游戏上的性能,本文提出了 LDF-IPER-DDQN 算法,并选取了 Atari 2600 游戏中单 Agent 的 8 个战略性游戏:Seaquest、Assault、Asterix、SpaceInvader、Berzerk、Q*bert、BeamRider 和 Gopher 来设计实验.这 8 个战略性游戏已在相关工作中得到广泛应用[10-11,24-25].

4.2 对比算法

本文实验分别在训练期间和训练完成后,评估了 PER-DDQN、DF-PER-DDQN、LDF-PER-DDQN、LDF-IPER-DDQN 这四种算法在部分 Atari 2600 游戏上的表现.具体如下:

(1)PER-DDQN 是基于优先级经验重放机制的深度双 Q 网络[26],该算法为基准比较算法;

(2)DF-PER-DDQN 是在优先级经验重放的深度双 Q 网络中加入动态跳帧方法[25];

(3)LDF-PER-DDQN 是将线性动态跳帧加入到深度双 Q 网络中,替代先前的固定跳帧,并且使

用优先级经验重放，但是样本的优先级仅仅考虑样本的 TD 误差，没有考虑其它信息.

(4) LDF-IPER-DDQN 是本文提出的将线性动态跳帧和优先级经验重放结合的深度双 Q 网络，和 LDF-PER-DDQN 不同之处在于将样本的 TD 误差和样本动作跳帧率共同作为衡量优先级的标准.

4.3　参数设置

在本文设计的实验中，四种算法使用的大部分超参数是相同的，超参数的设置与 DQN[11]、PER-DDQN 算法[26]中超参数设置保持一致，一方面保证了对比实验的统一性和有效性，另一方面该参数设置可以获得较优的实验效果.

上述四种算法的网络结构与 DDQN 很类似，网络模型中包含的卷积网络结构有很强的特征表达能力，能够自动学习到游戏画面的良好特征表达，因此实验过程中不需要针对特定任务去人工设计一些特征作为输入数据. 值得注意的是，在 DF-PER-DDQN 中，最后全连接层的隐藏层有 1024 个节点，且输出层节点的数量是游戏合法动作数量的两倍；而在 LDF-PER-DDQN 和 LDF-IPER-DDQN 中，最后输出动作的 Q 值和跳帧率.

在 PER-DDQN 中，使用了固定的跳帧方式，跳帧率 $k=4$. 在 DF-PER-DDQN 中，$k=4$ 或 20. 而在 LDF-PER-DDQN 和 LDF-IPER-DDQN 中，采用线性动态跳帧，$k_{\min}=4$，$k_{\max}=21$.

样本池的最大容量为 100 万个转移样本. 在训练的开始阶段，为了使 Agent 在训练初期有足够的样本，在 50000 更新时间步之前，Agent 采取随机的策略存储转移样本到样本池中. 这样的方式使得 Agent 的训练没有偏向性. 在优先级采样的方式中，分别设置 α 为 0.7，β 为 0.5，ε_0 为 0.0001，ε^- 随着时间步的增长从 1 衰减到 0.0001. β 随着时间步的增长从 0.5 增长到最大值 1. 超参数 α 和 β 的设置对本文算法具有一定的影响，因此本文依据 Schaul 等人[26]的工作，选取了较优值设置超参数 α 和 β，并在实验中探究不同参数值对本文算法的影响. 在本文实验中，优先级经验重放技术均采取比例化的优先级方式.

在实验中，还运用了一些常用方法[11]. (1) 由于每个游戏的环境不同，不同游戏的得分之间有着很大的差异，为了缩小 Q 值的范围，实验过程中，将正的奖赏设置为 1，负的奖赏设置为 −1，奖赏为零则保持不变；(2) 使用基于均方根的随机梯度下降来更新网络的参数，其中动量系数设置为 0.95. 在训练过程中，将梯度裁剪到[−5，5]区间中，将 TD 误差项裁剪到[−1，1]区间中. 裁剪 Q 值、梯度和 TD 误差项到特定的区间将有利于不同游戏之间使用相同量级的学习率，从而提高了学习的稳定性，避免陷入局部最优；(3) 每次更新网络参数时，都是从样本池中选取批量的转移样本，实验中，设置批量 n 为 32. 同时，折扣因子设置为 0.99. 学习率 η 和行为策略的参数 ε 都设置为从游戏开始到 100 万幅视频帧区间内递减的方式，即学习率从 0.005 递减到 0.00025，探索因子 ε 从 1.0 递减到 0.1.

4.4　实验评估

深度网络模型的训练往往需要较长的时间周期，也需要大量的训练数据. 因此，大多通过分阶段(epoch)来评估网络模型的训练过程. 而在 RL 方法中，一般采用从一个情节(episode)开始到结束获得的累积奖赏作为评价标准. Atari 2600 游戏中，一个情节指游戏的开始到结束，即到达游戏的终止状态；获得的累积奖赏就是一局游戏所获得的得分. 结合这两类学习的度量标准，本文采用训练过程中各阶段的平均每情节奖赏数来评估四个算法的性能. 在评估不同训练阶段的过程中，四种算法采取的都是 ε-greedy 策略.

为了保证不同算法之间的参数一致性，本文实验对每个游戏开展 200 个独立的训练阶段(epoch). 在每个训练阶段中，Agent 会经历不同数量的训练情节(episode). 因此，在训练初期情节数较多，随着训练的不断进行，Agent 逐步学习到更高效的策略，情节数将逐步降低. 在每个训练情节中包含了多次参数更新和评估的操作，称为一个训练步(step). 由于每个训练阶段包含的训练情节数量不尽相同，本文实验借鉴文献[11]中训练 DQN 的做法，设置每个训练阶段包含 50000 步的参数更新和 25000 步的评估. 通过累加每个训练阶段过程中所有游戏情节得到的奖赏获得每个情节的平均奖赏.

5　实验结果和分析

本节主要进行了如下两组实验并对实验结果进行具体的分析.

(1) 比较了四种算法在训练 Seaquest、Assault、Asterix、SpaceInvader、Berzerk、Q*bert、BeamRider 和 Gopher 这 8 个游戏过程中的表现，并结合 Seaquest

游戏具体分析说明 LDF- IPER-DDQN 算法能够有效提升 Agent 性能的原因.

(2)将训练完成后的四个模型用来测试这 8 个游戏,比较它们在测试过程中的性能.

5.1 训练时的表现

图 2 展示了四种算法在训练 8 种游戏时的各阶段平均每情节奖赏,其中 x 轴表示训练阶段,y 轴表示平均每情节奖赏.从图 2 中可以获得以下信息:

(1)本文提出的 LDF-IPER-DDQN 算法的训练效果好于其余三种算法,平均每情节得到的奖赏随着训练阶段的增加越来越大,明显超过了其余三种算法,且差距随着训练时间的推移越发显著.这是因为将线性动态跳帧方法和优先级采样结合,可以在两个转移样本具有相近 TD 误差时,让具有较大跳帧率的转移样本更频繁地被重放,从而更加快速地学习到玩游戏的一些关键性策略.同时,LDF-IPER-DDQN 的性能好于 LDF-PER-DDQN 证实了将跳帧率作为评价样本优先级标准的有效性.

(2)LDF-PER-DDQN 平均每情节获得的奖赏数大于 DF-PER-DDQN 算法,说明了线性动态跳帧

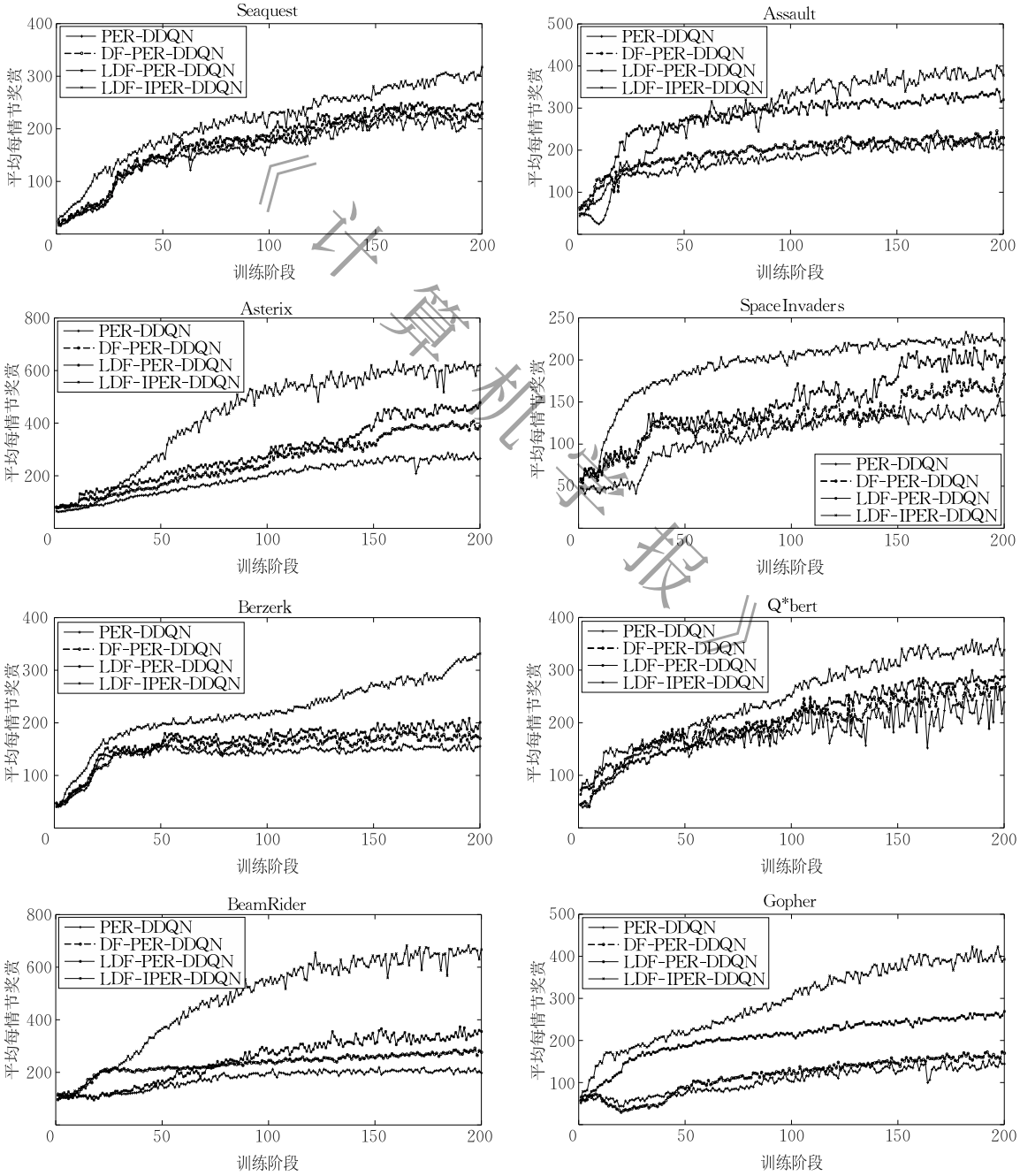


图 2 采用不同算法训练 8 种游戏时各阶段平均每情节奖赏对比

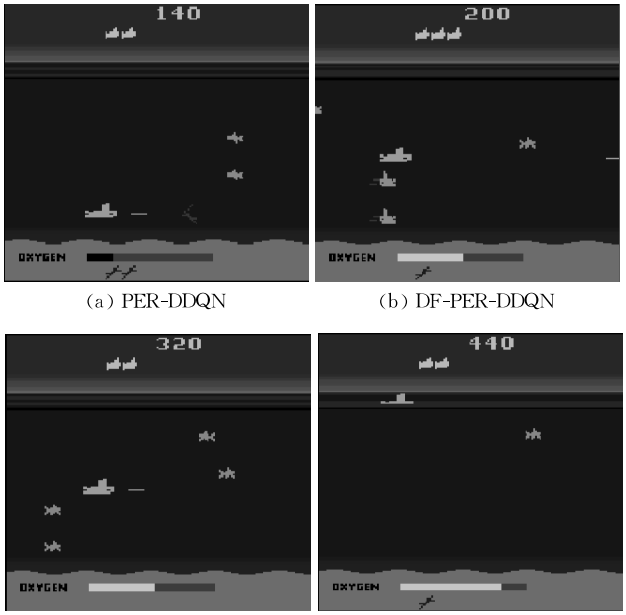
方法可以有效改进算法的性能. 这是因为线性动态跳帧方法可以更好地确定每个状态下每个动作需要重复执行的次数，让 Agent 在大部分时间里利用那些需要重复执行多次的动作，同时也不会完全忽略不需要重复执行多次的动作. 通过这样的方法，Agent 能够学习到更喜欢扩展的动作，从而提升自身的决策能力. 而 DF-PER-DDQN 不管当前状态下选择动作的好坏，都将重复执行选择的动作 4 次或者 20 次，这样的做法很可能使得不好的动作也被重复执行很多次，从而影响了学习的效率. 此外，DF-PER-DDQN 和 LDF-PER-DDQN 算法平均每情节获得的奖赏都高于 PER-DDQN，这说明将动态跳帧和优先级经验重放结合，能够提升 Agent 的性能.

(3) 四种算法在训练过程中，各阶段平均每情节奖赏的值都存在一定的波动，但是本文的 LDF-IPER-DDQN 算法波动相对来说较小，且平均每情节奖赏一直处于上升的趋势中. 奖赏值产生波动的原因主要是在训练过程中，网络的参数不断地在更新，即使网络参数的变动很小，也会导致输出的每个动作 Q 值发生很大的变化，进而导致 Agent 下一阶段学习到的策略也发生较大的变化. 观察图 2，我们还能发现在有些游戏(例如 Asterix 游戏)的训练初期，LDF-IPER-DDQN 算法的性能并没有表现得很好，这可能是由于训练初期样本数量不足以及训练过程具有随机性和不稳定性导致的. 但是经过一定时间步的不断训练，LDF-IPER-DDQN 算法的性能变得越来越好，并且超过其余三种算法.

下面结合 Seaquest 游戏，对不同算法在训练时产生较大性能差异的原因进行具体分析. Seaquest 游戏是一种战略性的游戏，Agent 在训练这个游戏时，需要学习到一些关键的策略，才能提升自身的性能. 在 Seaquest 的游戏过程中，有些动作需要在一个状态下重复执行多次，且该动作带来的效益需要很多时间步后才会体现在游戏画面中并被 Agent 感知. 比如，潜水艇在氧气不足时应多次执行上浮到水面储备氧气的动作. 如果 Agent 学习不到这个动作，那么得分会很低.

首先，为了直观地展示这四种算法对 Seaquest 游戏的影响，图 3 给出了这四种算法在训练中期阶段(500 万步)时游戏的屏幕截图.

观察图 3 可知：图 3(a)中此时游戏得分仅为 140，并且氧气条快要耗尽，潜艇也没有执行向上的



(a) PER-DDQN　(b) DF-PER-DDQN

(c) LDF-PER-DDQN　(d) LDF-IPER-DDQN

图 3　四种算法训练 Seaquest 时中期阶段游戏截屏

动作. 由于 PER-DDQN 采用固定的跳帧方式，不管当前选择的动作如何，该动作都将被重复执行 4 次，难以学习到 Seaquest 游戏中氧气不足时不断向上的动作，因此其性能不是很理想. 图 3(b)中此时游戏得分为 200，潜艇的氧气条已消耗一半，此时潜艇没有执行向上的动作. 在 DF-PER-DDQN 中，当前的输入状态仅仅由离当前时刻最近的 4 幅连续游戏画面构成，并且不管当前选择的动作好坏，Agent 将重复执行 4 次或者 20 次. 这很可能导致不好的动作也被执行 20 次，Agent 将无法及时地得到延迟的奖赏，将一直学习不到关键动作和关键策略. 因此 DF-PER-DDQN 性能提升得较慢但其总体性能仍优于 PER-DDQN，这也验证了动态跳帧的有效性. 图 3(c)中此时游戏得分为 300，得分有所提高，但是当潜艇的氧气条也已消耗一半时，潜艇仍然没有学习到上浮到水面储备氧气的动作. 这是因为 LDF-PER-DDQN 所使用的线性动态跳帧可以更好地根据当前状态和动作选择跳帧率，Agent 可以学习到当前时刻状态下需要重复执行的动作的对应次数，所以可以更快地提升 Agent 的性能. 进一步地，图 3(d)中此时游戏得分为 440，在前期潜艇氧气不足时，潜艇上浮到水面补充氧气，该动作有利于游戏的继续进行，从而获得了更高的游戏得分. 由于 LDF-IPER-DDQN 算法将线性动态跳帧方法和改进的优先级经验重放机制结合，更频繁地将这些重要的样本重放，从而更加快速地学习到游戏中潜水艇氧气不足且还在海底深处时，重复多次执行向上动作直到储

备了足够的氧气的关键策略，提升在游戏上的表现.

其次，为了进一步说明这四种算法在训练时的稳定性，以 Seaquest 游戏为例，图 4 对比了四种算法在训练 Seaquest 游戏中各阶段的平均每情节的最大动作 Q 值. 动作 Q 值函数表示在当前任意给定的状态下，Agent 根据当前策略选择动作到情节结束获得的累积折扣奖赏. 动作 Q 值函数的波动越小，说明该算法的性能越稳定. 为了在不同游戏中比较学习过程的稳定性，本实验中已将 Q 值裁剪到一定的区间中.

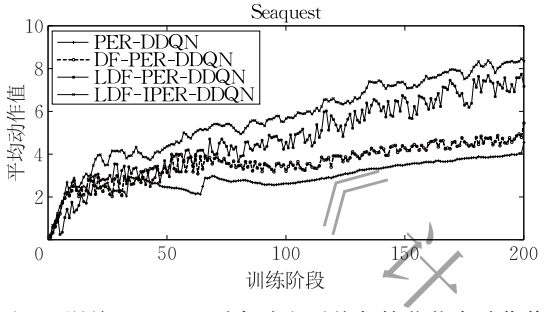


图 4 训练 Seaquest 时各阶段平均每情节状态动作值

从图 4 中可以得出以下观察：

(1) 四种算法均使用了优先级采样的方式，因此在训练过程中网络评估的 Q 值都呈现了平缓上升并趋于收敛的趋势，充分说明了这四种算法在训练时的有效性和稳定性.

(2) 在训练过程中平均每情节最大动作值的表现方面，LDF-IPER-DDQN 最优，LDF-PER-DDQN 与 DF-PER-DDQN 次之，PER-DDQN 最差. 这是因为 DF-PER-DDQN 使用了动态跳帧，进一步地，LDF-PER-DDQN 和 LDF-IPER-DDQN 则使用了线性动态跳帧方法. 动态跳帧方法可以将延迟奖赏及时反馈给 Agent，从而促使 Q 值函数增长.

(3) LDF-PER-DDQN 和 LDF-IPER-DDQN 的 Q 值曲线一直处于较快的上升趋势中，可以预见：随着训练阶段的延长，这两种算法的性能会越来越好. 而对于 DF-PER-DDQN，因为一直学习不到游戏中提高游戏得分的关键性策略，Q 值曲线的上升较为缓慢并趋于平缓.

最后，为了进一步说明超参数 α 和 β 的取值对本文算法性能的影响，以 Seaquest 游戏为例，图 5 对比了 LDF-IPER-DDQN 算法在游戏中每阶段平均每情节奖赏. 从图 5 中可以得出结论，超参数 α 和 β 不同的值会对本文算法产生一定的影响，但是差距不是很大. 我们发现当 $\alpha=0.7$，$\beta=0.5$ 时，LDF-IPER-DDQN 算法性能略优于其余两组设置，因此在本文实验中，超参数 α 和 β 分别设置为 0.7 和 0.5.

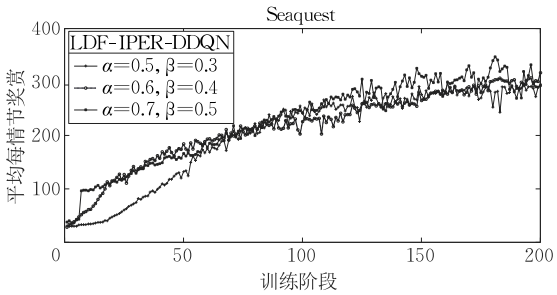


图 5 不同 α 和 β 值对 LDF-IPER-DDQN 算法影响

5.2 训练后的表现

一个深度网络算法模型的优劣不仅要在训练过程中评估，还需要在模型训练完成后进行评估. 我们的目标是通过训练得到一个可以反复测试的网络模型，使得 Agent 可以根据模型训练得到的策略进行游戏，获得更高的得分. 根据 PER-DDQN、DF-PER-DDQN、LDF-PER-DDQN、LDF-IPER-DDQN 这四种算法在训练时的表现，可以猜想 LDF-IPER-DDQN 在训练完成后，指导 Agent 进行游戏时的性能是优于其它模型的.

为了验证上述猜想，接下来将对比不同算法在进行 8 种游戏时的性能差异. 实验中，通过一个步长为 25 000 的游戏测试过程来评估训练完成后不同算法模型的性能好坏. 此时，Agent 仍执行 ε-greedy 策略，其中，$\varepsilon=0.05$. 采取这种策略是为了最小化过拟合的可能性. 针对不同游戏，每个训练完成的模型均被测试 200 次，且每次游戏的初始状态均设置为不同状态以确保测试结果的多样性. 每次测试都将获得一个得分，代表此次游戏的平均每情节所得奖赏. 实验比较了四种算法模型经过 200 次测试后的平均得分和最高得分.

测试得分结果如表 1 所示，根据平均得分和最高得分的数据，均可以看出：与其余三种算法相比，训练完成后的 LDF-IPER-DDQN 在指导 Agent 进行这 8 种游戏时的得分都高于其它三种算法. 特别地，在 SpaceInvaders 和 Berzerk 这两个游戏中性能提升地非常明显. 这充分证实了我们的猜想，LDF-IPER-DDQN 不仅仅能够在训练过程中表现得很好，在训练完成完成后的测试阶段也优于其余算法模型. 总而言之，与训练过程类似，针对这 8 种游戏，四种算法的性能基本保持了 LDF-IPER-DDQN＞LDF-PER-DDQN＞DF-PER-DDQN＞PER-DDQN 的态势.

表 1　训练完成后的不同模型在战略性游戏上的测试得分

游戏	Agent	平均得分	最高得分
Seaquest	PER-DDQN	1326.5	1840
	DF-PER-DDQN	1454.4	2010
	LDF-PER-DDQN	1678.5	2250
	LDF-IPER-DDQN	**1864.4**	**2430**
Assault	PER-DDQN	1245.4	1560
	DF-PER-DDQN	1456.8	1730
	LDF-PER-DDQN	1765.2	1980
	LDF-IPER-DDQN	**2056.5**	**2500**
Asterix	PER-DDQN	1731.8	2500
	DF-PER-DDQN	1750.5	2800
	LDF-PER-DDQN	2431.8	3000
	LDF-IPER-DDQN	**2804.7**	**3300**
Q*bert	PER-DDQN	4211.3	4575
	DF-PER-DDQN	4324.6	4590
	LDF-PER-DDQN	4581.4	4600
	LDF-IPER-DDQN	**4805.2**	**4625**
SpaceInvaders	PER-DDQN	1025.3	1320
	DF-PER-DDQN	1154.7	1410
	LDF-PER-DDQN	1325.3	1620
	LDF-IPER-DDQN	**1563.5**	**1940**
Berzerk	PER-DDQN	578.3	820
	DF-PER-DDQN	705.7	900
	LDF-PER-DDQN	878.3	1020
	LDF-IPER-DDQN	**1075.4**	**1500**
BeamRider	PER-DDQN	1784.3	2400
	DF-PER-DDQN	1967.6	2650
	LDF-PER-DDQN	2480.5	2870
	LDF-IPER-DDQN	**2990.3**	**3400**
Gopher	PER-DDQN	1245.8	1640
	DF-PER-DDQN	1435.7	1850
	LDF-PER-DDQN	1743.2	2230
	LDF-IPER-DDQN	**2130.5**	**2560**

5.3　实验总结

在本文实验中，分别使用 PER-DDQN、DF-PER-DDQN、LDF-PER-DDQN、LDF-IPER-DDQN 这四种算法模型训练 Agent 进行 Seaquest、Assault、Asterix、Q*bert、SpaceInvaders、Berzerk、BeamRider 和 Gopher 这 8 种游戏，分析比较它们在训练阶段的表现. 随后，将这四种训练完成后的模型用来测试这 8 种游戏，并再次分析比较它们的性能.

通过分别比较训练过程和测试过程的结果，可以得出以下结论：

(1) 线性动态跳帧可以根据当前网络输出的 Q 值函数来动态决定动作的跳帧率，即决定这个动作要被重复执行的次数. 它根据当前动作的重要程度来决定跳帧率，较之 DF-PER-DDQN 中仅仅用了固定为 4 或 20 的跳帧率，线性动态跳帧方法提高了 Agent 的性能，充分弥补动态跳帧的不足.

(2) 不同于 PER-DDQN 中的样本优先级，LDF-IPER-DDQN 中样本的优先级考虑了两个要素：样本的 TD 误差以及样本中动作的跳帧率，这样的设置使样本池中 TD 误差较高且需多次执行的动作对应的样本重放得更频繁，从而使得 Agent 更加快速地学习到提高游戏得分的关键策略.

(3) 在 8 种游戏中，我们验证了 LDF-IPER-DDQN 的性能，LDF-IPER-DDQN 在训练过程和训练完成后测试过程中，表现均优于 PER-DDQN，DF-PER-DDQN 和 LDF-PER-DDQN，充分说明 LDF-IPER-DDQN 算法的有效性，它能够进一步有效提升 Agent 的学习能力.

6　结　论

本文创新性地提出了一种线性动态跳帧方法，并将它和优先级经验重放技术结合起来，给出了 LDF-IPER-DDQN 算法. 该算法融合了线性动态跳帧方法和优先级重放的优势：线性动态跳帧方法可以使得 Agent 学习到每个状态下执行选择动作的次数；优先级经验重放则可以加速这一过程. 在样本池中，有些样本的动作在该状态下可能需要重复执行很多次，LDF-IPER-DDQN 能够将这些重要的样本重放得更频繁，从而学习到对决策更有帮助的策略.

本文设计了多组实验，从多个角度验证了 LDF-IPER-DDQN 算法的有效性. LDF-IPER-DDQN 在训练过程中的平均每情节的奖赏值，训练速度和训练稳定性总体上是优于 PER-DDQN，DF-PER-DDQN 和 LDF-PER-DDQN 的. 进一步地，本文还测试了训练好的每个算法模型在不同游戏上的表现. 结果表明，LDF-IPER-DDQN 的表现也是最优的. 因此，线性动态跳帧方法和基于跳帧的优先级重放机制相比传统的动态跳帧和优先级重放机制具有更优的性能.

本文提出的算法还具有很好的适用性，可以和很多网络模型相结合，比如基于竞争的 Q 网络、深度循环网络模型等，这样的结合可以有利于提高 Agent 在 Atari 2600 游戏上的表现.

未来的研究工作包括如何将优先级经验重放机制和行动者-评论家算法[27-29]的通用动态跳帧方法结合，将跳帧率参数变成一个可通过网络学习的参数. 此外，我们尝试使用 AlphaGo Zero[30] 中将值网络和策略网络结合的单一网络来训练 Agent，网络输出每个状态的值和该状态选择的动作和跳帧率.

参 考 文 献

[1] Yu Kai, Jia Lei, Chen Yu-Qiang, Xu Wei. Deep learning: Yesterday, today, and tomorrow. Journal of Computer Research and Development, 2013, 50(9): 1799-1804(in Chinese)
(余凯，贾磊，陈雨强，徐伟. 深度学习的昨天、今天和明天. 计算机研究与发展，2013，50(9)：1799-1804)

[2] Jiao Li-Cheng, Yang Shu-Yuan, Liu Fang, et al. Seventy years of neural network: Review and prospect. Chinese Journal of Computers, 2016, 39(8): 1697-1716(in Chinese)
(焦李成，杨淑媛，刘芳等. 神经网络七十年：回顾与展望. 计算机学报，2016，39(8)：1697-1716)

[3] Li Yan-Dong, Hao Zong-Bo, Lei Hang. Convolution neural network research review. Journal of Computer Applications, 2016, 36(9): 2508-2515(in Chinese)
(李彦冬，郝宗波，雷航. 卷积神经网络研究综述. 计算机应用，2016，36(9)：2508-2515)

[4] Graves A, Mohamed A, Hinton G. Speech recognition with deep recurrent neural networks//Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing. Vancouver, Canada, 2013: 6645-6649

[5] Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks//Proceedings of the 26th Annual Conference on Neural Information Processing Systems. Nevada, USA, 2012: 1097-1105

[6] Liang Bin, Liu Quan, Xu Jin, et al. Specific target sentiment analysis based on multi-attention convolutional neural network. Journal of Computer Research and Development, 2017, 54(8): 1724-1735(in Chinese)
(梁斌，刘全，徐进等. 基于多注意力卷积神经网络的特定目标情感分析. 计算机研究与发展，2017，54(8)：1724-1735)

[7] Sutton R S, Barto A G. Reinforcement Learning: An Introduction. Cambridge: MIT Press, 1998

[8] Lange S, Riedmiller M. Deep auto-encoder neural networks in reinforcement learning//Proceedings of the 7th International Joint Conference on Neural Networks. Barcelona, Spain, 2010: 1-8

[9] Liu Quan, Zhai Jian-Wei, Zhang Zong-Zhang, et al. Review of deep reinforcement learning. Chinese Journal of Computers, 2018, 41(1): 662-728(in Chinese)
(刘全，翟建伟，章宗长等. 深度强化学习综述. 计算机学报，2018，41(1)：662-728)

[10] Brockman G, Cheung V, Pettersson L, et al. OpenAI Gym. arXiv preprint arXiv: 1606.01540, 2016

[11] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning. Nature, 2015, 518(7540): 529-533

[12] Van Hasselt H, Guez A, Silver D. Deep reinforcement learning with double Q-learning//Proceedings of the Association for the Advance of Artificial Intelligence. Phoenix, USA, 2016: 2094-2100

[13] Nair A, Srinivasan P, Blackwell S, et al. Massively parallel methods for deep reinforcement learning. arXiv preprint arXiv: 1507.04296, 2015

[14] Hausknecht M, Stone P. Deep recurrent Q-learning for partially observable MDPs//Proceedings of the Association for the Advance of Artificial Intelligence Fall Symposium Series. Arlington, USA, 2015: 29-37

[15] Sorokin I, Seleznev A, Pavlov M, et al. Deep attention recurrent Q-network. arXiv preprint arXiv: 1512.01693, 2015

[16] Xu K, Ba J, Kiros R, et al. Show, attend and tell: Neural image caption generation with visual attention//Proceedings of the 32nd International Conference on Machine Learning. Lille, France, 2015: 2048-2057

[17] Liu Quan, Zhai Jian-Wei, Zhong Shan, et al. A deep recurrent Q-network based on visual attention mechanism. Chinese Journal of Computers, 2017, 40(6): 1353-1366(in Chinese)
(刘全，翟建伟，钟珊等. 一种基于视觉注意力机制的深度循环Q网络模型. 计算机学报，2017，40(6)：1353-1366)

[18] Oh J, Chockalingam V, Singh S, et al. Control of memory, active perception, and action in minecraft. arXiv preprint arXiv: 1605.09128, 2016

[19] Wang Z, Freitas N D, Lanctot M. Dueling network architectures for deep reinforcement learning//Proceedings of the International Conference on Machine Learning. New York, USA, 2016: 1995-2003

[20] Baird III L C. Reinforcement Learning Through Gradient Descent. Carnegie Mellon University, USA, 1999

[21] Bellemare M G, Ostrovski G, Guez A, et al. Increasing the action gap: New operators for reinforcement learning//Proceedings of the Association for the Advance of Artificial Intelligence. Phoenix, USA, 2016: 1476-1483

[22] Bellemare M G, Dabney W, Munos R. A distributional perspective on reinforcement learning//Proceedings of the 34th International Conference on Machine Learning. Sydney, Australia, 2017: 449-458

[23] Fortunato M, Azar M G, Piot B, et al. Noisy networks for exploration. arXiv preprint arXiv: 1706.10295, 2017

[24] Hessel M, Modayil J, Van Hasselt H, et al. Rainbow: Combining improvements in deep reinforcement learning//Proceedings of the 32nd Association for the Advance of Artificial Intelligence. New Orleans, Louisiana, USA, 2018: 3215-3222

[25] Lakshminarayanan A S, Sharma S, Ravindran B. Dynamic frame skip deep Q network. arXiv preprint arXiv: 1605.05365, 2016

[26] Schaul T, Quan J, Antonoglou I, Silver D. Prioritized experience replay. arXiv preprint arXiv: 1511.05952, 2016

[27] Mnih V, Badia A P, Mirza M, et al. Asynchronous methods for deep reinforcement learning//Proceedings of the International Conference on Machine Learning. New York, USA, 2016: 1928-1937

[28] Hausknecht M, Stone P. Deep reinforcement learning in parameterized action space. arXiv preprint arXiv: 1511.04143, 2015

[29] Foerster J N, Assael Y M, De Freitas N, et al. Learning to communicate to solve riddles with deep distributed recurrent Q-networks. arXiv preprint arXiv: 1602.02672, 2016

[30] Sliver D, Schrittwieser J, Simonyan K, et al. Mastering the game of Go without human knowledge. Nature, 2017, 550 (7676): 354

CHEN Song, M. S. candidate. His main research interests include reinforcement learning, deep learning and deep reinforcement learning.

ZHANG Xiao-Fang, Ph. D., associate professor. Her research interests include reinforcement learning, software testing, and deep reinforcement learning.

ZHANG Zong-Zhang, Ph. D., associate professor. His research interests include reinforcement learning, intelligent planning and multi-agent systems.

LIU Quan, Ph. D., professor, Ph. D. supervisor. His main research interests include intelligence information processing, automated reasoning and machine learning.

WU Jin-Jin, M. S. candidate. Her main research interest is deep reinforcement learning.

YAN Yan, M. S. candidate. His main research interest is deep reinforcement learning.

Background

Deep reinforcement learning, as a novel combination of reinforcement learning and deep learning in the artificial intelligence community, has achieved unprecedented progress in a variety of domains — involving both rich perception of high-dimensional raw inputs and action selection — such as robot control, text recognition and games. The Deep Q-Network (DQN) is a state-of-the-art deep reinforcement learning method, and gains wide attention due to its excellent human-level performance over several Atari 2600 games. However, DQN's performance falls far below human level in some strategic games, especially those require long sequential decision making to find a near optimal solution. To be efficient in handling these kinds of strategic games, our paper proposes a new algorithm: Deep Double Q-network based on Linear Dynamic Frame Skip and Improved Prioritized Experiential Replay. Experimentally, our preliminary results demonstrate that training agents generated through our new model surpass DQN's performance on eight strategic Atari 2600 games.

This paper is partially supported by the National Natural Science Foundation of China (61472262, 61502329, 61772355, 61876119), the Natural Science Foundation of Jiangsu (BK20181432), the Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University (93K172014K04, 93K172017K18), and the Suzhou Technology Development Plan (SYG201807).

These projects aim to enrich the reinforcement-learning theory and develop efficient approximate algorithms to expand the power and applicability of reinforcement learning on large scale problems.