基于值函数和策略梯度的深度强化学习综述

刘建伟 高 峰 罗雄麟
(中国石油大学(北京)自动化系 北京 102249)

摘 要 作为人工智能领域的热门研究问题,深度强化学习自提出以来,就受到人们越来越多的关注.目前,深度强化学习能够解决很多以前难以解决的问题,比如直接从原始像素中学习如何玩视频游戏和针对机器人问题学习控制策略,深度强化学习通过不断优化控制策略,建立一个对视觉世界有更高层次理解的自治系统.其中,基于值函数和策略梯度的深度强化学习是核心的基础方法和研究重点.该文对这两类深度强化学习方法进行了系统的阐述和总结,包括用到的求解算法和网络结构.首先,本文概述了基于值函数的深度强化学习方法,包括开山鼻祖深度 Q 网络和基于深度 Q 网络的各种改进方法.然后介绍了策略梯度的概念和常见算法,并概述了深度确定性策略梯度、信赖域策略优化和异步优势行动者-评论家这三种基于策略梯度的深度强化学习方法及相应的一些改进方法.接着概述了深度强化学习前沿成果阿尔法狗和阿尔法元,并分析了后者和该文概述的两种深度强化学习方法的联系.最后对深度强化学习的未来研究方向进行了展望.

关键词 深度学习;强化学习;深度强化学习;值函数;策略梯度;机器学习
中图法分类号 TP18 **DOI 号** 10.11897/SP.J.1016.2019.01406

Survey of Deep Reinforcement Learning Based on Value Function and Policy Gradient

LIU Jian-Wei GAO Feng LUO Xiong-Lin
(*Department of Automation, China University of Petroleum, Beijing* 102249)

Abstract As a hot research problem in the field of artificial intelligence, Deep Reinforcement Learning (DRL) has attracted more and more attention since it was proposed. At present, DRL can solve many problems that were previously difficult to solve such as learning how to play video games directly from raw pixels and learning a control strategy for robot problems. DRL builds an autonomous system with a higher level understanding of the visual world by a continous optimization of the control strategy. Among them, DRL based on value function and policy gradient is the core basic method and research focus. This paper systematically elaborates and summarizes two types of DRL methods including solving algorithms and network structures. Firstly, DRL methods based on value function are summarized, including Deep Q-Network (DQN) and improved methods based on DQN. DQN is a pioneering work in the field of DRL. This model trains Convolutional Neural Network (CNN) with a variety of Q learning. Before the emergence of DQN, the problem of instability or even non-convergence will appear when the action value function in Reinforcement Learning (RL) is approximated by neural network. To solve this problem, DQN uses two technologies: the experience replay mechanism and the target network. According to different emphasis on DQN improvement, various improved versions based on DQN can be divided into four categories: improvement of training algorithm, improvement of neural network structure, improvement of

收稿日期:2018-03-10;在线出版日期:2018-10-31.本课题得到国家自然科学基金(21676295)、中国石油大学(北京)2018 年度前瞻导向及培育项目(2462018QZDX02)资助.**刘建伟**,博士,副研究员,主要研究方向为机器学习、智能信息处理、复杂系统的分析、预测与控制、算法分析与设计.E-mail:liujw@cup.edu.cn.**高 峰**,硕士研究生,主要研究方向为机器学习.**罗雄麟**,博士,教授,主要研究领域为智能控制、复杂系统分析、预测与控制.

introduction of new learning mechanism and improvement based on new proposed RL algorithm. The research motivation, overall thinking, advantages and disadvantages, application scope and performance of DQN improvement are elaborated in detail. Then the concept and common algorithms of policy gradient are introduced. Policy gradient algorithm is widely used for RL problems in continuous space. Its main idea is to parameterize the policy, calculate the policy gradient about the action and the action is adjusted continuously along the direction of the gradient and the optimal policy is gradually obtained. The common policy gradient algorithm includes REINFORCE algorithm and Actor-Critic algorithm. Also, DRL methods based on policy gradient are summarized including Deep Deterministic Policy Gradient (DDPG), Trust Region Policy Optimization (TRPO), Asynchronous Advantage Actor-Critic (A3C) and some corresponding improved methods. Drawing on DQN technology, DDPG adopts the experience replay mechanism and a separate target network to reduce the correlation between data and increase the stability and robustness of the algorithm. The problem solved by TRPO is to select the appropriate step size by introducing the trust region constraint defined by Kullback-Leibler divergence so as to ensure that the optimization of the policy is always in the good direction. A3C uses a conceptually simple and lightweight DRL framework and optimizes the deep neural network controller using an asynchronous gradient descent method. Then, AlphaGo and Alpha Zero which represent advanced research achievements of DRL are summarized and the relationship between the latter and the two DRL methods summarized in this paper is analyzed. Then some common experimental platforms of DRL algorithms are introduced including ALE, OpenAI Gym, RLLab, MuJoCo and TORCS. Finally, the future research directions of DRL are prospected.

Keywords　deep learning; reinforcement learning; deep reinforcement learning; value function; policy gradient; machine learning

1　引　言

1.1　强化学习和多臂赌博机

强化学习(Reinforcement Learning,RL)是智能体和环境相互交互,通过“试错(trial and error)”的方式来学习最优策略的马尔科夫决策过程(Markov Decision Process,MDP). RL 系统由 4 个基本部分组成:状态集合 S、动作集合 A、状态转移概率矩阵 $\boldsymbol{P}$ 和奖励函数 R. 定义策略 π 为状态空间到动作空间的映射. 智能体在当前状态 s_t 下根据策略 π 选择动作 a_t 作用于环境,然后接收到环境反馈回来的奖励 r_t,并以转移概率 $p^{a}_{s_t s_{t+1}}$ 转移到下一个状态 s_{t+1}. RL 的目标是通过不断调整策略来最大化长期累积奖励. 此外定义两种函数来衡量每个状态或状态-动作对的好坏,它们分别为状态值函数 $V^{\pi}(s)$ 和动作值函数 $Q^{\pi}(s,a)$,两者的数学表达式中都用到了折扣因子 γ,用来表示未来某一时刻的奖励在累积奖励中所占的影响比重,即未来奖励的当前价值. 最优策略 π^{*} 则定义为使状态值函数和动作值函数取最大所对应的策略.

典型的 RL 问题具有以下 3 个特点:

(1) 不同的动作产生不同的奖励;

(2) 奖励在时间上具有延迟性;

(3) 某个动作的回报和具体所处环境有关.

由于在 RL 问题中经历多步动作之后才能观察到最终的奖励值,因此可将 RL 问题简化为多臂赌博机学习问题[1](Multi-armed Bandit). 多臂赌博机可以看作是 RL 问题的一个原型,该模型只满足上述的第 1 个特点,而不满足第 2 和第 3 个特点.

多臂赌博机模型假定臂个数 $K\geq 2$,每一个臂与未知奖励序列 $X_{i,1}, X_{i,2}, \cdots$ 相关联,其中 $i=1, 2, \cdots, K$,每一时刻 t 智能体选择某个臂 I_t,环境返回奖励 $R_{I_t,t}$,智能体学习的目标是使得后悔函数 $L_n=\max\limits_{i=1,\cdots,K}\sum\limits_{t=1}^{n}R_{i,t}-\sum\limits_{t=1}^{n}R_{I_t,t}$ 最小. 多臂赌博机模型按奖励类型分为随机性多臂赌博机模型、对手型多臂赌博机模型、马尔科夫多臂赌博机模型、线性多臂赌博机模型和非线性多臂赌博机模型.

RL 最基础的研究问题就是在学习中的每一步

要抉择是冒险探索(exploration),还是利用以前的经验获得当前最大的收益(exploitation)的问题.智能体可以通过尝试非最优的动作来探索环境中未见的可能带来更大预期收益的动作和状态,也可以利用当前最优的动作来取得短期当前最大的收益,因此需要在探索和利用之间权衡.在多臂赌博机问题中,"探索"对应估计每个臂的奖励期望,"利用"对应选择当前得到最多奖励的最优臂.同样地,由于用于探索和利用的总次数有限,因此需要对二者进行折中.用于解决探索-利用问题的多臂赌博机方法有[1]:ε-贪婪探索、贝叶斯探索、软最大探索、跟随随机置换领导者和乐观探索等.

1.2 深度强化学习

虽然 RL 在自然科学、社会科学和工程等领域取得了很大的成功[2-4],但是很多方法都缺少可扩展的能力,并局限于低维问题.这些限制的存在是因为 RL 和其它学习算法一样存在复杂性的问题:存储复杂性和计算复杂性.因此直接从如视觉、语音这样的高维感官输入学习控制智能体是 RL 一直以来面临的挑战.

深度学习(Deep Learning,DL)近来的发展已经使从原始感官输入中提取高水平特征成为可能,并引领了计算机视觉[5-6]、语音识别[7-8]和目标检测[9-10]等领域的突破性进展,其中利用了一系列的深度神经网络(Deep Neural Networks,DNN)结构,包括卷积神经网络[11](Convolutional Neural Network,CNN)、循环神经网络[12](Recurrent Neural Network,RNN)、深度残差网络[13](Deep Residual Network,DRN)和生成式对抗网络[14](Generative Adversarial Network,GAN)等.DL 最重要的特性是,DNN 可以自动学习高维数据(如图像、文本和音频等)的低维特征表示.基于以上事实,我们很自然地会考虑一个问题:类似的 DL 技术是否也对涉及感官数据的 RL 有效呢?

于是在过去几年,DL 已经成功地和 RL 相结合,两者的结合就叫做深度强化学习(Deep Reinforcement Learning,DRL).DL 使 RL 能够扩展到一些以前难以处理的问题,比如高维状态空间和高维动作空间设定下的决策问题,并利用 DL 提取高维特征的能力,使 DRL 模型只用原始输入而不用人工特征来生成输出,从而实现端到端(end-to-end)的学习.目前 DRL 已经广泛地应用于游戏[15-16]、机器人[17-18]、对话系统[19-20]、交通信号灯控制[21-22]、自动驾驶[23-24]和无线电[25-26]等领域.

本文和文献[27]、文献[28]都是关于 DRL 的综述性论文.其中刘全等人[27]总体上对 DRL 各类方法进行了简略阐述,包括基于值函数和基于策略梯度这两类最基本的方法,但对这两类方法只举例介绍了具体几个有代表性的算法和结构,赵冬斌等人[28]则是主要围绕阿尔法狗围棋程序进行阐述和总结.

本文则是重点针对上述两类方法进行详细地阐述,并囊括了很多最新改进和适用于不同场景的算法、机制和网络结构等.同时本文增加了实验的对比与分析部分,介绍 DRL 常用的几种实验平台,并分析实验结果.除此之外,本文对 DRL 领域最新的前沿成果,新一代围棋程序阿尔法元进行介绍,并分析两代围棋程序阿尔法狗、阿尔法元和本文阐述的两类 DRL 方法之间的联系.最后,本文还尝试提出 DRL 的几个未来研究方向.

2 基于值函数的深度强化学习

基于值函数的 DRL 是用 DNN 逼近奖励值函数.类似地,用 DNN 逼近策略并利用策略梯度方法求得最优策略,则被称为基于策略梯度的 DRL.下面我们将分别对二者进行概述.

值函数是 RL 中的基本概念,时间差分学习[29](TD learning)和 Q 学习[30](Q learning)则分别是学习状态值函数和动作值函数的经典算法.下面我们重点介绍开山鼻祖深度 Q 网络(Deep Q-network,DQN)和基于 DQN 的各种改进方法.

2.1 深度 Q 网络

Mnih 等人[15]提出的 DQN 模型是 DRL 领域的开创性工作,该模型用 Q 学习的一种变种训练 CNN.接着他们针对上述模型添加目标网络改进 DQN 的学习性能[16].

在 DQN 出现之前,当用神经网络来逼近 RL 中的动作值函数时,会出现不稳定甚至不收敛的问题.为了解决此问题,DQN 使用了两个技术:经验回放(experience replay)机制[31]和目标网络.DQN 训练过程示意图如图 1 所示.

(1)经验回放机制.经验回放机制的具体做法是在每个时间点存储智能体的经验 $e_t=(s_t,a_t,r_t,s_{t+1})$,形成回放记忆序列 $D=\{e_1,\cdots,e_N\}$.训练时,每次从 D 中随机提取小批量的经验样本,并使用随机梯度下降算法更新网络参数.经验回放机制通过重复采样历史数据增加了数据的使用效率,同时减少了数

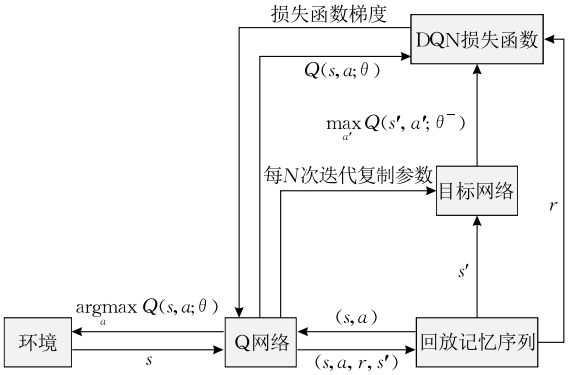

图 1　DQN 训练过程示意图

据之间的相关性.

(2) 目标网络. DQN 使用深度卷积网络(Q 网络)来逼近值函数,并将值函数参数化表示为 $Q(s,a;\theta_i)$,其中 θ_i 为 Q 网络在第 i 次迭代的参数(也就是神经元连接权重). Q 网络每次迭代的优化目标值为 $y=r+\gamma \max_{a'} Q(s',a';\theta_i^-)$,该目标值由另外一个单独的目标网络产生,其中 a' 为所有可能的动作,s' 为下一时刻的状态,θ_i^- 为目标网络的参数(也就是神经元连接权重).

Q 网络训练过程中,参数通过最小化以下损失函数进行第 i 次更新:

$$L_i(\theta_i)=E_{(s,a,r,s')}[(y-Q(s,a;\theta_i))^2] \tag{1}$$

上面的损失函数对权重求偏导得:

$$\nabla_{\theta_i} L_i(\theta_i)=E_{s,a,r,s'}[(y-Q(s,a;\theta_i))\nabla\theta_i Q(s,a;\theta_i)] \tag{2}$$

目标网络和 Q 网络都为深度卷积网络,二者结构相同. 并且目标网络的参数 θ_i^- 每经过 N 次迭代用 Q 网络的参数 θ_i 更新一次,在中间过程 θ_i^- 则保持不变.

具体的 DQN 网络结构如图 2 所示,输入是经过预处理后的最近的连续 4 帧游戏画面,然后经过 3 层卷积层和 2 层全连接层,最后输出每个动作所对应的 Q 值.

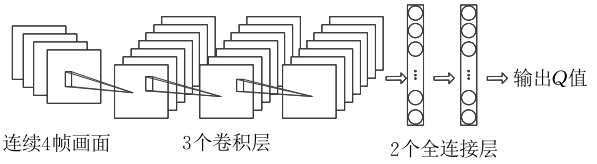

图 2　DQN 网络结构图

其中 DQN 所用的神经网络并没有池化层,这不同于很多传统的 CNN(如用于手写数字识别问题的 CNN).

DQN 的 3 个主要贡献是:

(1) 使用经验回放机制和目标网络来稳定训练过程.

(2) 设计了一种端到端的 RL 方法,即只以原始像素和游戏比赛得分作为输入,输出每个动作对应的 Q 值.

(3) 在不同的任务中使用相同的模型参数、训练算法和网络结构. 在 49 个 Atari 2600 游戏中的实验结果,超过以往所有算法的实验结果,并且可以媲美一位职业人类玩家的水平.

2.2　DQN 的改进方法

自 DQN 提出之后,出现了基于 DQN 的各种改进版本,按照对 DQN 改进侧重点的不同,可分为训练算法的改进、神经网络结构的改进、引入新的学习机制的改进和基于新提出的 RL 算法(归一化优势函数、多 Q 学习)的改进这 4 大类. 除此之外,还有一些其它改进方法. 接下来将对这些改进方法进行具体阐述,包括其研究动机、整体思路、优缺点、适用范围和性能好坏等. DQN 改进方法的具体分类如图 3 所示.

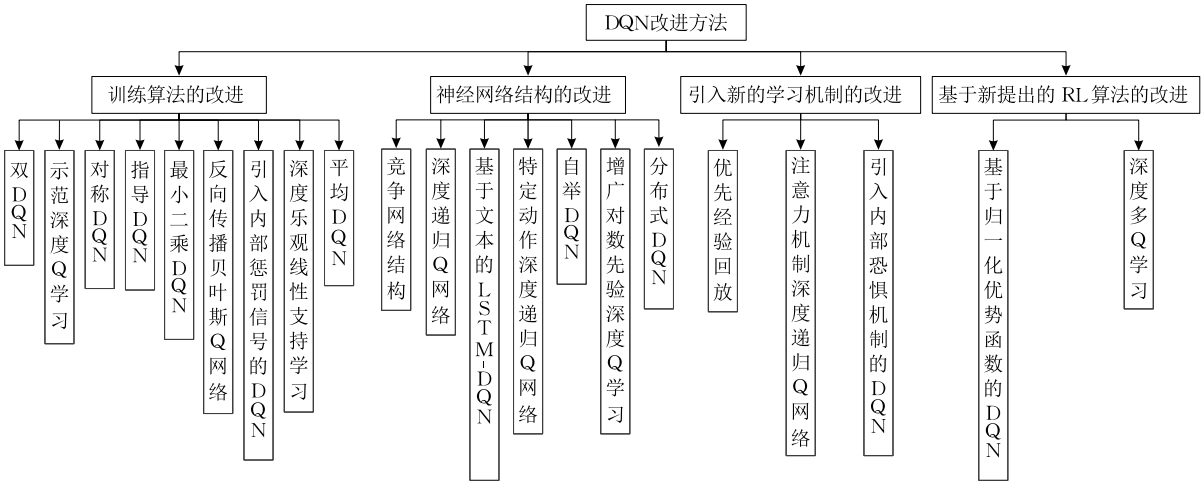

图 3　DQN 改进方法分类图

2.2.1 训练算法的改进

双 DQN. Hasselt 等人[32]基于 DQN 和双 Q 学习(Double Q-learning)算法提出双 DQN(Double DQN,DDQN)算法,解决了 Q 学习中的过高估计动作值函数的问题.

在标准 Q 学习以及 DQN 算法中,参数更新公式为

$$\theta_{t+1}=\theta_t+\alpha(y_t^Q-Q(s_t,a_t;\theta_t))\nabla_{\theta_t}Q(s_t,a_t;\theta_t) \tag{3}$$

其中优化目标值为

$$y_t^Q=r_{t+1}+\gamma\max_a Q(s_{t+1},a;\theta_t) \tag{4}$$

上式也可以写成:

$$y_t^Q=r_{t+1}+\gamma Q(s_{t+1},\operatorname{argmax}_a Q(s_{t+1},a;\theta_t);\theta_t) \tag{5}$$

从式(5)可以看出,选择和评价动作都是基于同一个参数 θ_t,这将更有可能造成过优的 Q 值估计. DDQN 算法借鉴双 Q 学习算法[33],对 DQN 算法进行小的改进:根据在线 Q 网络来估计策略,选择动作,用目标网络来估计 Q 值. DDQN 算法的更新和 DQN 一样,仅仅将 y_t^Q 替换为

$$y_t^{\text{DoubleDQN}}=r_{t+1}+\gamma Q(s_{t+1},\arg\max_a Q(s_{t+1},a;\theta_t);\theta_t^-) \tag{6}$$

其中,θ_t 是在线网络的参数,θ_t^- 是目标网络的参数. 实验结果显示,在一些 Atari 2600 游戏中,DDQN 比 DQN 有着更好的表现.

示范深度 Q 学习. DRL 算法通常需要在模拟环境中,经过和环境大量的交互才能达到令人满意的学习性能,当存在一个完全精确的模拟器时,这种情况是可以接受的. 然而在许多实际问题中,并没有这样的模拟器,这严重限制了 DRL 在许多实际任务中的适用范围. 因此智能体需要从学习一开始就有良好的性能做保障. 虽然实际问题中很难找到精确的模拟器,但是这些问题中的大多数都有在以前的控制器(人或是机器)下运行的系统的数据. 为了充分利用这些控制器的数据,即尽可能多地从示范数据中学习,Hester 等人[34]提出示范深度 Q 学习(Deep Q-learning from Demonstrations,DQfD)算法,它利用示范数据大大加快了学习过程. DQfD 算法分为两个阶段:

(1) DQfD 首先在示范数据上进行预训练,预训练阶段的目的是用值函数来学习模仿示范者,以便智能体和环境交互之后,可以用 TD 方法来进行更新. 网络更新结合了四种损失函数:单步双 Q 学习损失函数 $L_{DQ}(Q)$、n 步双 Q 学习损失函数 $L_n(Q)$、有监督大间隔分类损失函数 $L_E(Q)$ 和 L2 正则化损失函数 $L_2(Q)$. 两种 Q 学习损失函数 $L_{DQ}(Q)$ 和 $L_n(Q)$ 是为了保证网络满足贝尔曼方程,并为 TD 学习做铺垫. 监督损失函数是为了对示范者的动作进行分类. 正则化损失函数则是为了防止在相对较小的示范数据集上出现过拟合现象. 总的损失函数表达式为

$$L(Q)=L_{DQ}(Q)+\lambda_1 L_n(Q)+\lambda_2 L_E(Q)+\lambda_3 L_2(Q) \tag{7}$$

其中,λ 控制着损失函数之间的权重,$L_n(Q)$ 对应的 n 步回报为

$$r_t+\gamma r_{t+1}+\cdots+\gamma^{n-1}r_{t+n-1}+\max_a\gamma^n Q(s_{t+n},a) \tag{8}$$

大间隔分类损失函数定义为

$$L_E(Q)=\max_{a\in A}[Q(s,a)+l(a_E,a)]-Q(s,a_E) \tag{9}$$

其中,a_E 是专家示范者在状态 s 采取的动作,$l(a_E,a)$ 为间隔函数,$a=a_E$ 时为 0,其它情况为正数.

(2) 智能体开始和环境交互,DQfD 通过从自身训练中产生的数据和示范数据中采样来继续学习,这两种数据在采样过程中的比例由优先回放机制自动控制.

实验表明,在 42 个 Atari 游戏中的 41 个游戏上,DQfD 都比优先竞争双 DQN[32,35-36](Prioritized Dueling Double DQN,PDD DQN)得分更多.

对称 DQN. 复杂的 DRL 学习中需要大量的训练数据,因此利用 RL 过程中的对称性来保证抽样效率至关重要. Mahajan 等人[37]提出了一种新的框架来发现环境中的对称性,并应用到 DQN 中的函数逼近过程,这就是对称 DQN(Sym DQN)算法.

MDP 中的对称性:将 MDP 表示为元组 $M=\langle S,A,\Psi,T,R\rangle$,其中 S 是状态集合,A 是动作集合,$\Psi\subset S\times A$ 是状态-动作对集合,$R:\Psi\times S\to R$(R 是实数集合)和 $T:\Psi\times S\to[0,1]$分别是奖励函数和转移函数. MDP 中对称性的概念可以使用 MDP 同态映射[38]的概念来严格定义. 将从 $M=\langle S,A,\Psi,T,R\rangle$ 到 $M'=\langle S',A',\Psi',T',R'\rangle$ 的 MDP 同态映射 h 定义为满射 $h:\Psi\to\Psi'$,h 定义为满射元组 $\langle f,\{g_s,s\in S\}\rangle$. 特别地,$h((s,a))=(f(s),g_s(a))$,这里,$f:S\to S'$,$g_s:A_s\to A'_{f(s)}$.

基于以上定义,当 f 和 g_s 为满射映射函数时,对称性映射函数 $\chi:\Psi\to\Psi'$ 定义为 M 上的自同构映射关系,并且 $T(f(s),g_s(a),f(s'))=T(s,a,s')$ 和 $R(f(s),g_s(a),f(s'))=R(s,a,s')$ 这两个条件需要同时满足.

对称 DQN 提出的新框架主要包括两个部分:

(1) 对称性检测程序. 用来计算状态-动作对之间的对称性估计,找到对称的状态-动作对 χ_{sym}.

(2) 对称性实现机制. 将检测到的对称性应用到 Q 函数逼近，促进对称性策略学习过程.

找到一些对称的状态-动作对 χ_{sym} 之后，接下来就是利用这个信息来训练神经网络. 为了让对称的状态-动作对在神经网络中有相同的输出，在神经网络的顶层学习对称状态-动作对的相同的表示. 实现上述目标的方法就是直接用 χ_{sym} 给一步 TD 损失函数引入硬性约束：

$$L_{i,\text{TD}}(\theta_i)=$$
$$E_B[(r+\gamma\max_{a'}Q(s',a';\theta_{i-1})-Q(s,a;\theta_i))] \tag{10}$$

其中，B 是采取可以保证充分探索的 ε-贪婪策略的观测元组 (s,a,r,s') 的集合，因此问题变为

$$\min_{\theta_i}L_{i,\text{TD}}(\theta_i)$$
$$\text{满足 } Q(s,a;\theta_i)=Q(s',a';\theta_i)\forall(\langle s,a\rangle,\langle s',a'\rangle)\in\chi_{\text{sym}} \tag{11}$$

然而如果有太多的约束，很难优化上面的问题. 此外，因为检测到的对称的状态-动作对不能保证准确，所以引入对称约束作为附加的损失函数项是更好的选择：

$$L_{i,\text{sym}}(\theta_i)=E_{\chi_{\text{sym}}}[(Q(s',a';\theta_i)-Q(s,a;\theta_i))^2] \tag{12}$$

最后对称 DQN 的损失函数由两项组成：

$$L_{i,\text{total}}(\theta_i)=L_{i,\text{TD}}(\theta_i)+\lambda L_{i,\text{sym}}(\theta_i) \tag{13}$$

其中，λ 是对称损失的权衡参数.

实验表明，在 Cart-Pole 游戏中，对称 DQN 比 DQN 获得了更多的分数.

指导 DQN. 为了解决机器人学习空中冰球击打的控制策略问题，Taitler 等人[39]提出指导 DQN (Guided DQN)算法，它基于标准 DQN 算法做出以下两点改进：

(1) 在许多情况中，先验知识对应合理的动作，虽然通常它并不是最优策略. 先验知识可以作为指导策略，比如围棋中下棋规则能够指导智能体选择动作，在一些训练时间段中，智能体根据指导策略来选择动作，而不是贪婪策略，也就是指导 DQN 使用指导策略和贪婪策略的混合策略选择动作. 这里对应冰球问题，指导策略为让智能体朝着冰球的方向移动，而不是向左或向右移动.

(2) 在冰球和 Atari 游戏问题中，采用 DQN 会出现随着学习过程的进行，学习效果逐渐变差的问题. 指导 DQN 则通过调整目标网络参数的更新周期，来避免学习效果的下降. 在训练开始阶段，由于回放记忆序列是空的并且想要实现快速学习，所以先采用一个小的更新周期. 然后随着样本的增加，不断增大更新周期. 随着学习效果不断变好，样本分布逐渐稳定，也同样继续增大更新周期，这是为了保持训练过程的稳定并使智能体所选择的策略保持在好的学习策略附近波动. 更新周期表达式为 $C=C\cdot C_r$，其中 $C_r\geq 1$ 为每次周期增大的系数，当 $C_r=1$ 时，指导 DQN 算法就变为标准 DQN 算法.

实验表明，指导 DQN 在空中冰球击打的运动规划和控制问题上表现接近最优，指导 DQN 完全解决了采用 DDQN 算法会出现的得分降低问题.

最小二乘 DQN. 当 RL 用 DNN 做函数逼近时，叫 DRL；当 RL 用一些线性函数作为奖励值函数逼近器时，叫浅强化学习(Shallow RL，SRL). 一方面，DRL 可以学到丰富的特征表示. 另一方面，SRL 学习过程比较稳定. 为了将 DRL 和 SRL 相结合，同时利用两者的优点，Levine 等人[40]将 DQN 和线性最小二乘法相结合，提出最小二乘 DQN(Least Squares Deep Q-Network，LS-DQN).

拟合 Q 迭代(Fitted Q Iteration，FQI)算法是使用回归来迭代计算 Q 函数逼近值的一种批 SRL 算法. 在该算法的第 N 次迭代，用样本个数为 N_{SRL} 的数据集 $D=\{s_i,a_i,r_i,s_{i+1}\}_{i=1}^{N_{\text{SRL}}}$ 和上一次迭代的逼近奖励值函数 Q^{N-1} 构造监督学习的目标：$y_i=r_i+\gamma\max_{a'}Q^{N-1}(s_{i+1},a')$，$\forall i\in N_{\text{SRL}}$. 然后把 y_i 看作 $Q(s_i,a_i)$ 要逼近的真实值，使最小二乘损失函数 $(Q(s_i,a_i)-y_i)^2$ 最小，以监督学习的方法求解下一次迭代的逼近值 Q^N：

$$Q^N=\arg\min_Q\sum_{i=1}^{N_{\text{SRL}}}(Q(s_i,a_i)-$$
$$(r_i+\gamma\max_{a'}Q^{N-1}(s_{i+1},a')))^2 \tag{14}$$

根据表示理论，FQI 用线性函数表示 $Q(s_i,a_i)$ 的分量 $Q_n(s,a)=\boldsymbol{\varphi}^{\text{T}}(s,a)\boldsymbol{w}_n$，其中 $\boldsymbol{\varphi}^{\text{T}}(s,a)$ 是特征矩阵，相当于基向量矩阵，$\boldsymbol{w}_n$ 是权重向量，用式(14)学习控制策略就是学习权重向量 $\boldsymbol{w}_n$.

用最小二乘法可以得到 FQI 的解：$\boldsymbol{w}_n=\arg\min_{\boldsymbol{w}}\|\tilde{\boldsymbol{J}}\boldsymbol{w}-\tilde{\boldsymbol{b}}\|_2^2$，这里 $\tilde{\boldsymbol{J}}$ 和 $\tilde{\boldsymbol{b}}$ 分别为

$$\tilde{\boldsymbol{J}}=\frac{1}{N_{\text{SRL}}}\sum_{i=1}^{N_{\text{SRL}}}[\boldsymbol{\varphi}(s_i,a_i)^{\text{T}}\boldsymbol{\varphi}(s_i,a_i)] \tag{15}$$

$$\tilde{\boldsymbol{b}}=\frac{1}{N_{\text{SRL}}}\sum_{i=1}^{N_{\text{SRL}}}[\boldsymbol{\varphi}(s_i,a_i)^{\text{T}}y_i] \tag{16}$$

LS-DQN 学习过程则是基于 FQI 算法周期性地重新训练 DQN 的最后一个隐藏层. LS-DQN 学习过程的关键是引入用来做最小二乘更新的贝叶斯正则化项，它将 DQN 最后一个隐藏层的参数作为 FQI 算法的贝叶斯先验，防止了数据的过拟合. FQI

假定第 k 次迭代的先验权重概率分布为：$\boldsymbol{w}_{\text{prior}} \sim N(\boldsymbol{w}_k, \lambda^2)$，其中 $\boldsymbol{w}_k$ 是 FQI 算法第 k 次迭代 DQN 最后一个隐藏层连接边的权重. 根据 Box 等人[41]给出的 FQI 算法中回归问题的贝叶斯解，LS-DQN 算法最后一个隐藏层连接边的权重的更新公式为

$$\boldsymbol{w}=(\tilde{\boldsymbol{J}}+\lambda \boldsymbol{I})^{-1}(\tilde{\boldsymbol{b}}+\lambda \boldsymbol{w}_k) \tag{17}$$

其中，$\tilde{\boldsymbol{J}}$ 和 $\tilde{\boldsymbol{b}}$ 上面已经给出，$\boldsymbol{I}$ 是单位矩阵，λ 是正态分布的标准差.

实验表明，在一些 Atari 游戏中 LS-DQN 的表现比 DQN 和 DDQN 有着明显的提高，这种提高归功于批最小二乘更新方法. LS-DQN 的提出也为未来线性 RL 和 DRL 的结合打下坚实的基础.

反向传播贝叶斯 Q 网络. 当奖励值稀疏，动作空间很大时，使用 ε-贪婪冒险探索的 DQN 智能体学习性能并不好，这时就需要开发有效的冒险探索方法. 为了解决这个问题，Lipton 等人[42]提出反向传播贝叶斯 Q 网络(Bayes-by-Backprop Q-network，BBQN).

BBQN 的一个显著特征就是在进行动作选择时，量化了 Q 函数估计的不确定性，并用来指导冒险探索过程. 在 DQN 中，Q 函数由参数为 $\boldsymbol{w}$ 的神经网络表示，而在 BBQN 中则是把 $\boldsymbol{w}$ 看作随机向量，计算 $\boldsymbol{w}$ 的后验概率分布 $q(\boldsymbol{w}|\theta)$，其中 $\boldsymbol{w}$ 的后验概率分布 q 是具有对角协方差阵的多变量高斯分布，也就是说 $\boldsymbol{w}$ 是从 $N(\mu_i, \sigma_i^2)$ 中采样得到. 为了保证所有 σ_i 都保持正定，将 σ_i 参数化为 $\sigma_i=\log(1+\exp(\rho_i))$.

给定训练数据集 $D=\{x_i, y_i\}_{i=1}^N$，假定随机向量概率分布 q 的参数为 $\theta=\{(\mu_i, \rho_i)\}_{i=1}^D$. 根据反向传播贝叶斯方法[43]，给定先验概率 $p(\boldsymbol{w})$，求解最优 θ 的目标函数为

$$L(\theta)=\min_{\theta}(\log q(\boldsymbol{w}|\theta)-\log p(\boldsymbol{w})-\log p(D|\boldsymbol{w})) \tag{18}$$

BBQN 借鉴 DQN 的思想，同样使用目标网络来计算优化目标值 y. BBQN 目标网络的参数更新方式和 DQN 的目标网络更新方式相同，假定 BBQN 目标网络的参数为 $\tilde{\theta}=\{(\tilde{\mu}_i, \tilde{\rho}_i)\}_{i=1}^D$，$y$ 的产生有两种方法：

(1) 用蒙特卡洛抽样方法从目标网络中提取样本 $\tilde{\boldsymbol{w}} \sim q(\cdot|\tilde{\theta})$，使用 $y=r+\gamma \max_{a'} Q(s', a'; \tilde{\boldsymbol{w}})$ 来计算 y.

(2) 使用最大后验(Maximum a Posterior，MAP)估计来计算 y：$y=r+\gamma \max_{a'} Q(s', a'; \tilde{\mu})$. 实验表明这种方法效果更好，计算效率更高.

计算出 y 之后，得到数据集 $D=\{x_i, y_i\}_{i=1}^N$，其中 $x=(s, a)$. 然后用式(18)来优化 θ 直到收敛. θ 收敛之后，$\tilde{\theta}$ 被 θ 替代.

给定 q 之后，通过汤普森抽样方法实现冒险探索，其中对动作的抽取是根据当前状态最优的后验概率得到的. 给定 t 时刻状态 s_t 和参数 θ_t，首先抽样提取 $\boldsymbol{w}_t \sim q(\cdot|\theta_t)$，然后令 $a_t=\arg\max_a Q(S_t, a; \boldsymbol{w}_t)$.

BBQN 基于 DQN，并使用贝叶斯神经网络来逼近 Q 函数，解决了给定当前策略情况下 Q 值的不确定性问题. 实验表明，BBQN 比使用 ε-贪婪的 DQN 等传统方法学习速度更快.

引入内部惩罚信号的 DQN. 为了解决 DRL 累积奖励过高估计的问题并减少抽样复杂度，Leibfried 等人[44]通过将信息论中的有限理性原理[45](information-theoretic bounded rationality)推广到高维状态空间，在 DQN 中引入内部惩罚信号.

解决 Q 学习中过高估计 Q 值的一个方法就是通过给即时奖励加一个内部惩罚信号来注入“随机性”，以降低 Q 值的估计. 引入这种随机性的原理参考自信息论中的有限理性原理，其中智能体旨在找到使回报 $R(s, a)$ 最大的最优行为策略. 有限理性是给行为策略和先验策略 π_{prior} 的差加一个界限，这可以通过约束两者之间的 KL 散度来实现：

$$\begin{aligned}&\max_{\pi_{\text{behave}}} \sum_{a\in A} \pi_{\text{behave}}(a|s) R(s, a)\\ &\text{满足 } KL(\pi_{\text{behave}} \| \pi_{\text{prior}}) \leq K\end{aligned} \tag{19}$$

通过引入拉格朗日乘子 λ，上面的约束优化问题可以转化为等价的无约束优化问题：

$$\begin{aligned}L(s, \pi_{\text{prior}}; \lambda)=&\max_{\pi_{\text{behave}}} \sum_{a\in A} \pi_{\text{behave}}(a|s) R(s, a)-\\ &\frac{1}{\lambda} KL(\pi_{\text{behave}} \| \pi_{\text{prior}})\end{aligned} \tag{20}$$

其中，λ 是权衡前后两项之间权重的参数. 由于无约束问题(20)是关于优化参数的凹性问题，因此可以给出解析形式的最优解：

$$\pi^*_{\text{behave}}(a|s)=\frac{\pi_{\text{prior}}(a|s)\exp(\lambda R(s, a))}{\sum_{a'} \pi_{\text{prior}}(a'|s)\exp(\lambda R(s, a'))} \tag{21}$$

将式(21)代回式(20)中得

$$L^*(s, \pi_{\text{prior}}; \lambda)=\frac{1}{\lambda}\log \sum_{a\in A} \pi_{\text{prior}}(a|s)\exp(\lambda R(s, a)) \tag{22}$$

将以上思想引入到 DQN 中得

$$L^*(s, \pi_{\text{prior}}; \lambda, \theta)=\frac{1}{\lambda}\log \sum_{a\in A} \pi_{\text{prior}}(a|s)\exp(\lambda Q(s, a; \theta)) \tag{23}$$

然后利用上面的目标函数来定义 DQN 的优化目标函数为

$$L_\lambda(\theta)=$$

$$E_{s,a,r,s'}[(r+\gamma L^*(s,\pi_{\text{prior}};\lambda,\theta^-)-Q(s,a;\theta))^2] \quad (24)$$

其中，θ^-表示较早阶段学习得到的参数. 上式可以被认为是 DQN 的一种推广，换句话说，DQN 是上式的一个特例，因为当$\lambda\to\infty$时，上式变为

$$L_{\lambda\to\infty}(\theta)=$$

$$E_{s,a,r,s'}[(r+\gamma \max_a Q(s,a;\theta^-)-Q(s,a;\theta))^2] \quad (25)$$

实验表明在一些 Atari 2600 游戏中，引入内部惩罚信号的 DQN 在游戏表现和采样效率上都要优于 DQN 和 DDQN.

深度乐观线性支持学习. 为了解决多个目标之间相对重要性未知的高维多目标决策问题，Mossalam 等人[46]利用乐观线性支持[47](Optimistic Linear Support，OLS)框架，并结合深度 Q 学习算法，提出深度乐观线性支持学习(Deep Optimistic Linear Support Learning，DOL)算法，对 DOL 做了进一步的改进，进而提出另外两种新的算法：全部再用深度乐观线性支持学习(Deep OLS Learning with Full Reuse，DOL-FR)和部分再用深度乐观线性支持学习(Deep OLS Learning with Partial Reuse，DOL-PR).

OLS 是一种解决多目标决策问题的方法，能够得到决定目标之间相对重要性大小的向量 $\boldsymbol{w}$[48]. 给定通过 OLS 得到的 $\boldsymbol{w}$，DOL 通过优化参数 θ 来使网络输出的多目标 Q 值向量 $\boldsymbol{Q}(s,a;\theta)$ 和 $\boldsymbol{w}$ 之间的内积最小，即令 $f(\boldsymbol{Q}(s,a;\theta),\boldsymbol{w})=\boldsymbol{w}\cdot\boldsymbol{Q}(s,a;\theta)$ 最小，其中函数 f 将 Q 值向量映射为标量值. 将上述学习过程定义为标量化深度学习(scalarised deep Q-learning)，然后将标量化深度学习和 OLS 结合，就得到了提出的 DOL 算法. 为了满足 OLS 框架的要求，需要调整 DQN 的网络结构，将 DQN 的输出变为 n 个 Q 值，n 是目标的个数. 也就是说神经网络输出的是 n 个目标所对应的各个动作的 Q 值.

随着学习过程的进行，各个目标越来越相似，这体现在各个目标所对应的最优 Q 值向量越来越相似. 因为 DQN 可以用来提取 Q 值相关的特征，因此可以再用之前一个目标的网络参数来加快下一目标的学习进程. 根据再用网络参数方式的不同，分为 DOL-FR 和 DOL-PR 算法两种.

当学习一个新的标量权重 $\boldsymbol{w}'$时，DOL-FR 在目前 OLS 生成的所有标量权重中找到和 $\boldsymbol{w}'$最相近的一个权重 $\boldsymbol{w}$，再使用之前优化 $\boldsymbol{w}$ 的神经网络的所有层网络参数. 与 DOL-FR 方法不同的是，DOL-PR 方法随机对网络的最后一层参数重新进行初始化，而不是直接使用之前优化 $\boldsymbol{w}$ 的神经网络的最后一层参数，这个做法是为了避免陷入局部最优.

实验表明，提出的三种方法对于学习多目标 MDP 问题都有着很高的准确率，是 DRL 首次在学习多目标策略问题中取得成功. 并且 DOL-PR 比 DOL 和 DOL-FR 效果都要好，这表明：

(1)使用网络参数的方法是有效的.

(2)对网络参数进行部分再用而不是全部再用能有效地防止模型陷入局部最优，即陷入之前权重 $\boldsymbol{w}$ 所对应的的最优策略.

平均 DQN. 为了降低 DRL 算法的不稳定性和可变性对其产生的负面影响，Anschel 等人[49]对 DQN 算法进行扩展，提出平均 DQN(Averaged-DQN)算法. 该算法将之前的 Q 值估计取平均，使训练过程更加稳定，并通过减少目标逼近误差(Target Approximation Error，TAE)来提高性能.

平均 DQN 将之前学习的 K 个 Q 值取平均值：

$$Q^A_{i-1}(s,a)=\frac{1}{K}\sum_{k=1}^{K}Q(s,a;\theta_{i-k}) \quad (26)$$

将上式替换 DQN 中优化目标值中的 Q 值得到平均 DQN 的优化目标值：

$$y=r+\gamma \max_{a'} Q^A_{i-1}(s',a') \quad (27)$$

实验表明，在一些 Atari 游戏和 Gridworld 问题中，增加 K 的大小可以在学习到更好的策略的同时减少 Q 值的过高估计，显著提高了稳定性和游戏表现. 平均 DQN 可以和很多 DQN 的改进版本相结合，比如双 DQN、竞争网络结构和优先经验回放等.

2.2.2　神经网络结构的改进

竞争网络结构. Wang 等人[31]针对无模型 RL 提出一种新的神经网络结构：竞争网络结构(Dueling Network Architecture).

如图 4 所示，图 4 中上面是传统 DQN 的示意图，结构为输入层、卷积层、全连接层和输出每个动作的 Q 值输出层. 图 4 中下面则是竞争 Q 网络示意图，它把卷积层提取到的抽象特征分流到全连接

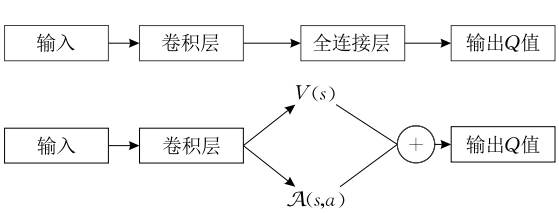

图 4　传统 DQN 和竞争 DQN 网络结构对比图

层的两条支路中.一条支路代表标量状态值函数 $V(s)$,另一条支路代表某个状态下的动作优势函数(advantage function)$\mathcal{A}(s,a)$.最后输出模块将两条支路组合起来得到每个动作的 Q 值.

优势函数定义如下:

$$\mathcal{A}^{\pi}(s,a)=Q^{\pi}(s,a)-V^{\pi}(s) \tag{28}$$

其中,奖励值函数 V 衡量了状态 s 的好坏程度,Q 函数衡量了在这个状态下选择某个特定动作 a 的奖励值的大小,优势函数 $\mathcal{A}$ 则衡量了这个状态下各个动作的相对好坏程度.

根据优势函数的定义构造的 Q 函数如下:

$$Q(s,a;\theta,\alpha,\beta)=V(s;\theta,\beta)+\mathcal{A}(s,a;\theta,\alpha) \tag{29}$$

其中 α 和 β 分别是两条支路的参数,θ 为 DQN 的参数.然而按以上优势函数构造的 Q 函数会导致解不唯一的问题:给定一个 Q 值,无法得到唯一的 V 和 $\mathcal{A}$,比如在 V 上加上一个值,在 $\mathcal{A}$ 上再减去同一个值,得到的是相同的 Q 值.因此由于这种问题的存在,在实际学习中直接使用式(29)时,学习的效果较差.

为了解决这个问题,令贪婪动作选择时的优势函数为零:

$$Q(s,a;\theta,\alpha,\beta)=V(s;\theta,\beta)+\left(\mathcal{A}(s,a;\theta,\alpha)-\max_{a'\in|A|}\mathcal{A}(s,a';\theta,\alpha)\right) \tag{30}$$

为了进一步改进学习效果,将上式的最大算子替换成优势函数的平均值:

$$Q(s,a;\theta,\alpha,\beta)=V(s;\theta,\beta)+\left(\mathcal{A}(s,a;\theta,\alpha)-\frac{1}{|A|}\sum_{a'}\mathcal{A}(s,a';\theta,\alpha)\right) \tag{31}$$

虽然这样的改进失去了 V 和 $\mathcal{A}$ 的原始语义,但是提高了优化过程的稳定性,这是因为式(31)中的优势函数只需要和其平均值变化一样快,而不需要和其最大值变化速度相匹配.

实验表明,结合 DDQN 训练算法和优先经验回放技术实现的竞争网络结构,在 Atari 游戏中的表现比以往只采用优先经验回放的 DQN 和 DDQN 都要更好.

除此之外,Raghu 等人[50]通过使用连续状态空间模型和 DRL 算法来解决败血症的最佳治疗策略学习问题,其中所用的网络结构结合了 DDQN 和竞争网络,并命名为竞争双深度 Q 网络(Dueling Double-Deep Q Network,Dueling DDQN).

深度循环 Q 网络. 由于实际应用中状态存在部分可观测性,一些复杂任务的状态信息经常是不完整且具有噪声干扰的.为了解决这个问题,Hausknecht 等人[51]将长短期记忆网络[52](Long Short Term Memory,LSTM)和 DQN 相结合,形成的网络叫深度循环 Q 网络(Deep Recurrent Q-Network,DRQN),用 LSTM 替换掉 DQN 中第一个全连接层来实现 DRQN.

DQN 有两个缺点:(1)记忆能力有限;(2)依赖于游戏画面的完整性.DQN 是将最近 4 帧的视频信息作为输入,虽然取得了不错的效果,但是也只能记住这 4 帧的信息.DRQN 虽然在每个时间点只能看见 1 帧,但却能够整合帧间的相关信息.实验结果表明,在标准 Atari 游戏中 DRQN 取得了和 DQN 相媲美的学习效果.此外,当用部分观测信息训练时,用逐渐更完整的观测信息评价时,DRQN 的性能和观测质量(游戏画面完整程度的大小)成一定的函数关系.相反,当用完整的观测信息进行训练,用部分观测信息评价时,DRQN 比 DQN 效果要差.

基于文本的 LSTM-DQN. 在基于文本的游戏中,虚拟世界中的所有交互都是通过文本进行的,并且底层状态是无法观测到的.由此造成的语言障碍使得这种环境对自动游戏玩家来说具有挑战性.Narasimhan 等人[53]提出基于文本的 LSTM-DQN 框架,以游戏奖励作为反馈,共同学习状态表示和动作策略.这个框架可以实现将文本描述映射为游戏状态语义的向量表示.

如图 5 所示,LSTM-DQN 模型分为两个部分,下面的部分是表示生成器(Representation Generator)ϕ_R,以观察到的当前状态的词嵌入向量 $\boldsymbol{w}$ 作为输入,经过 LSTM 和均值池化(Mean Pooling)处理,最终输出向量表示 $v_s=\phi_R(s)$.上面的部分是动作得分器(Action Scorer)ϕ_A,以 v_s 为输入,分别经过线性单元(Linear Unit)和线性整流单元(Rectified Linear

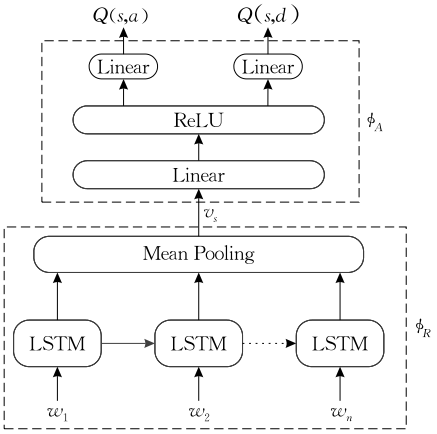

图 5 LSTM-DQN 网络结构图

Unit，ReLU)，最后再经过一个线性单元分别输出动作 a(如“吃”)和对象 d(如“苹果”)的分数. 因此最后动作 a 的 Q 函数的逼近值为 $Q(s,a)\approx \phi_A(\phi_R(s))[a]$，同样，可以得到对象 d 的 Q 函数.

实验表明，在两个游戏中提出的 LSTM-DQN 都要优于使用单词袋子模型、二元语法袋子模型做状态表示的基准方法，证明学习良好的文本表示的重要性.

特定动作深度循环 Q 网络. Zhu 等人[54]在 DRQN 的基础上，将过去动作的影响也考虑进去，提出特定动作深度循环 Q 网络(Action-specific Deep Recurrent Q-Network，ADRQN)，提高了解决 RL 学习部分可观测问题的性能. 具体地，将过去的动作和获得的观测一起输入到 Q 网络中，模型的结构如图 6 所示.

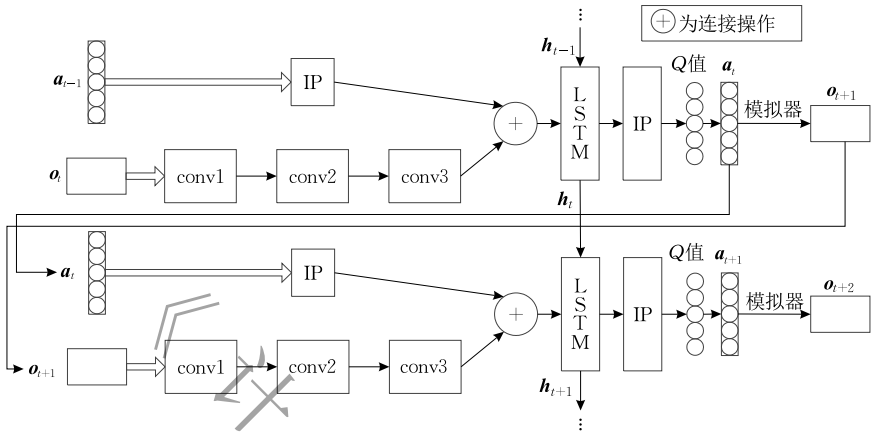

图 6　ADRQN 模型结构图

用独热向量来表示每个动作 $\boldsymbol{a}$，但直接将动作向量 $\boldsymbol{a}$ 和观测向量 $\boldsymbol{o}$ 结合效果并不好，因为两者之间的长度相差很多. 在 Atari 以及其它部分可观测问题中，动作的维数远远小于状态表示的维数. 为了解决这个不平衡问题，通过一个全连接层(图 6 中表示为 IP)将独热动作向量 $\boldsymbol{a}_{t-1}$ 嵌入到更高维的向量中. 视频帧 $\boldsymbol{o}_t$ 经过三个卷积层 conv1、conv2、conv3 输出的卷积观测和 IP 输出的动作向量连接形成动作-观测对，然后经过 LSTM，整合上一时刻 LSTM 的输出 $\boldsymbol{h}_{t-1}$ 得到 $\boldsymbol{h}_t=\text{LSTM}(\boldsymbol{h}_{t-1},\boldsymbol{o}_t)$，再经过 IP 输出各个动作对应的 Q 值. 然后经过模拟器(Simulator)产生下一时刻的观测值 $\boldsymbol{o}_{t+1}$.

和 DQN 及 DRQN 相比，ADRQN 能够记住过去的动作，尤其是上一个动作. 因此，传统 DQN 经验回放机制中的转移元组 (s_t,a_t,r_t,s_{t+1}) 改进为 $(\{a_{t-1},o_t\},a_t,r_t,s_{t+1})$，以便让框架更方便地提取动作-观测对. 实验表明，在几个 Atari 游戏上，ADRQN 的得分要高于 DRQN.

自举 DQN. 如何实现有效的冒险探索仍然是 RL 的主要挑战. 冒险探索是智能体不利用以前学习到的经验，在某个状态下，冒险选择不熟悉或未经历的动作，以便探测状态-动作的奖励值，从而获取经历的过程. 一方面，策略需要一定的冒险探索来发现新的策略；另一方面，策略也需要充分利用之前学到的知识以保证更有效率地探索. 常见的随机扰动策略，如 ε-贪婪 DRL 算法，试图通过不选择当前最优的 Q 值对应的策略和动作，即不选择贪婪策略，以一定比例加入随机选择的动作，即采用保守冒险探索策略. 其中保守冒险探索策略和贪婪策略之间的比例可以通过 ε 的大小来调整.

但是在复杂问题中，ε-贪婪算法十分低效，因为在 ε 大小一定的情况下，简单的随机策略很难到达离初始状态很远的状态，智能体到达某个状态的概率，是随着该状态离初始状态距离的增加而成指数减少的，这样的探索被称为很“浅”的探索，并没有实现深度探索. 采用随机值函数为进行有效的冒险探索提供了一种有效的方法[55]，但是现有的算法并不能实现非线性参数化的值函数上的学习.

为了解决这个问题，Osband 等人[56]提出自举 DQN(Bootstrapped DQN)算法，它也是第一个结合 DL 与深度冒险探索过程的实用 RL 算法.

自举是用抽样分布来估计总体分布的过程. 它通常的形式是，从原来的样本集 D 中有放回地随机抽取和 D 中样本相同数量的样本，放在集合 $\widetilde{D}$ 中，然后根据需要计算需要估计的总体分布统计量 $\varphi(\widetilde{D})$. 自举算法被广泛誉为 20 世纪应用统计学的一大进步，理论上保障了估计的一致性[57].

自举 DQN 网络结构图如图 7 所示，包括一个

共享网络和 K 个自举头(Head)分支,共享网络学习所有数据的联合特征表示,每个自举头在它的自举样本数据上进行训练. 自举 DQN 改进了 DQN,通过自举过程来估计 Q 值的概率分布. 在每个训练时间段(episode)的开始,自举 DQN 对 Q 值样本集进行自举采样,每个自举头代表采样后的一组 Q 值. 然后智能体根据此自举头的 Q 值选择最优策略进行探索,直到此时间段结束,然后在下一个时间段中,再随机选择一个自举头进行探索. 其它训练过程则和 DQN 一样.

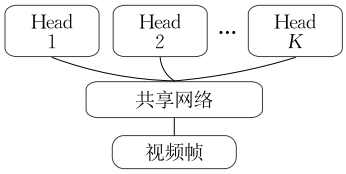

图 7 自举 DQN 网络结构图

自举 DQN 可以将深度冒险探索与 DNN 相结合,对 DNN 的不确定性进行有效的估计,并可实现与任何随机扰动策略学习过程的收敛速度相比,呈更快的指数级速度学习. 自举 DQN 在计算上是易处理的,并且可以自然地扩展到大规模并行系统. 在 Arcade 学习环境中,自举 DQN 显著提高了大多数游戏的学习速度和表现.

增广对数先验深度 Q 学习. Jaques 等人[58]为了改进 RNN 生成序列的结构和质量,同时保留从数据中得到的信息以及样本的多样性,提出增广对数先验深度 Q 学习(Deep Q-learning with log prior augmentation)算法.

具体地,先用最大似然估计(Maximum Likelihood Estimation)对 LSTM 进行预训练,然后为模型中的三个神经网络赋予初始权重:Q 网络、目标网络以及奖励 RNN. 其中奖励 RNN 的连接权重矩阵和偏置在训练过程中保持不变,同时输出先验策略 $\pi(a_t|s_t)$.

为了将 RL 应用到序列生成过程中,生成序列中的下一个时间段被看作是一个动作 a,环境状态包括目前生成的所有时间段,即 $s_t=\{a_1,a_2,\cdots,a_t\}$. 给定动作 a_t,为了将 $\pi(a_t|s_t)$ 和特定任务的奖励值 r_T 结合作为总奖励值,将总奖励定义为

$$r(s,a)=\log\pi(a|s)+r_T(s,a)/c \tag{32}$$

其中,c 为一个常数,用来控制 r_T 占总奖励的比例. 再将上式代入 DQN 的目标函数中得

$$L(\theta)=E_\beta[(r(s,a)+\gamma\max_{a'}Q(s',a';\theta^-)-Q(s,a;\theta))^2] \tag{33}$$

这个就是增广对数先验深度 Q 学习的目标函数. 预训练后的 LSTM 为 Q 网络、目标网络以及奖励 RNN 这三个神经网络提供初始权重的过程示意图如图 8 所示.

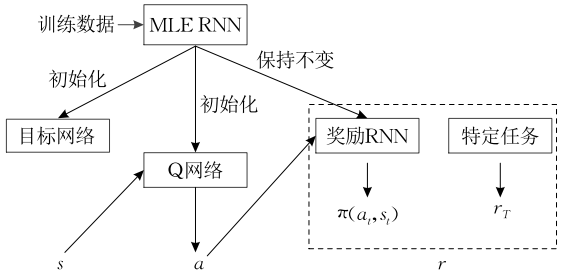

图 8 LSTM 提供初始权重的过程示意图

实验表明,增广对数先验深度 Q 学习改进了生成序列所需的属性和结构,同时保留了从数据中学习到的信息.

Gorila DQN. 为了能够通过开发大量的计算资源来扩展如 DQN 的 DRL 算法,Nair 等人[59]提出了 DRL 的第一个大规模分布式结构,将新提出的分布式结构 Gorila(General Reinforcement Learning Architecture)应用到 DQN 算法上,提出一种新的分布式算法 Gorila DQN,Gorila 结构如图 9 所示.

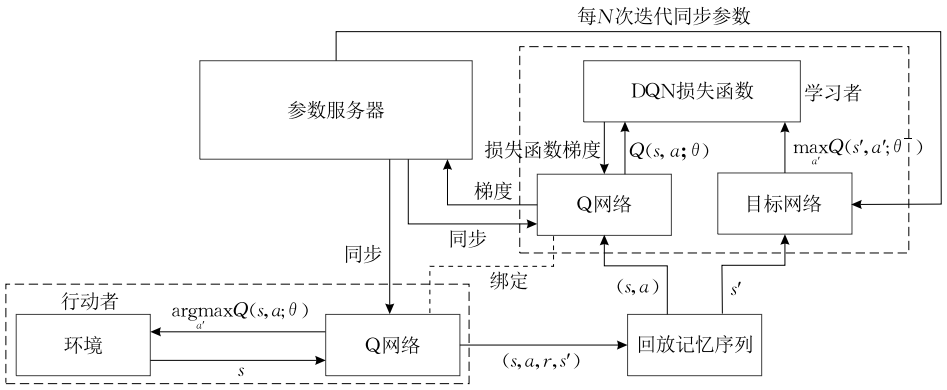

图 9 Gorila 结构图

该结构主要由四部分组成：

(1)用于产生新行为的行动者. 行动者包含一个 Q 网络的复制品，用于选择动作和环境交互，Q 网络的参数周期性地从参数服务器同步.

(2)分布式存储经验的回放记忆序列. 存储由行动者产生的经验元组.

(3)用存储的经验来训练的学习者. 学习者也包含一个 Q 网络的复制品，从回放记忆序列中采样数据，计算损失函数梯度，并发送给参数服务器，其中 Q 网络的参数周期性地从参数服务器同步.

(4)表示 Q 网络的分布式神经网络参数服务器. 参数服务器收到来自学习者发送的梯度，并用随机梯度下降法进行 Q 网络的参数更新.

Gorila DQN 分布式算法使用相同的参数，应用到街机环境下的 Atari 2600 游戏中的 49 个游戏中，其中 41 个游戏的表现都超过非分布式的 DQN(单 GPU 的 DQN)，25 个游戏更是超过人类职业玩家的水平，同时也大大减少实现这些结果所需要的时间. Gorila 是一个随着计算复杂性和存储复杂性的增加性能都更好的可扩展结构.

2.2.3　引入新机制的改进

优先经验回放. 在线 RL 智能体通过经验回放技术记住和重复使用过去的经验. 以前的研究工作中，经验转移都从回放记忆中采样. 然而，这个方法仅仅同等频率地回放之前经历的转移元组(s_t，a_t，r_t，s_{t+1})，却没有考虑序列中各个元组的重要程度不同. 基于此，Schaul 等人[35]提出优先经验回放(Prioritized Experience Replay)，以便更频繁地回放重要的转移元组，从而更有效地实现在线 RL.

经验转移元组的重要性由时间差分(Temporal Difference，TD)误差来衡量，TD 误差绝对值越大的样本被抽取用来训练的概率越大. Schaul 等人[35]基于 TD 设计了一种随机采样方法：随机优先级(stochastic prioritization)采样，即使用重要性采样机制来避免更新过程引起的偏置，并且把优先经验回放应用到 DQN 和 DDQN 中，提高了其在 Atari 游戏中的表现.

注意力机制深度循环 Q 网络. 视觉注意力模型在字幕生成、目标跟踪和机器翻译领域取得了很大的成功，因此 Sorokin 等人[60]在 DRQN 结构中加入注意力机制(attention mechanism)，提出注意力机制深度循环 Q 网络(Deep Attention Recurrent Q-Network，DARQN). 其中采用的注意力机制分为软注意力机制(soft attention mechanism)和硬注意力机制(hard attention mechanism)两种.

DARQN 的结构如图 10 所示，它包括三种类型的网络：CNN、注意力机制网络 g 和循环神经网络(LSTM). 在每一个时间点 t，CNN 以视频帧的形式收到当前状态 s_t 的表示，根据这个表示产生一组特征映射. 注意力机制网络把这些映射转化为向量集 $\boldsymbol{v}_t=\{\boldsymbol{v}_t^1,\cdots,\boldsymbol{v}_t^L\}$ 并输出它们的线性组合 $\boldsymbol{z}_t$，称为上下文向量. LSTM 把上下文向量 $\boldsymbol{z}_t$ 作为输入，再根据之前的隐状态 $\boldsymbol{h}_{t-1}$ 和记忆状态 $\boldsymbol{c}_{t-1}$，产生隐状态 $\boldsymbol{h}_t$. $\boldsymbol{h}_t$ 被用于：(1)线性层中，评价智能体在状态 s_t 选择的动作 a_t 的 Q 值；(2)注意力机制网络在下一时间点 $t+1$ 生成上下文向量.

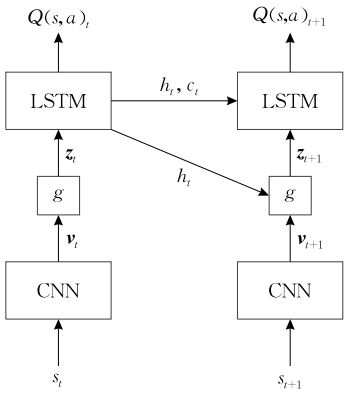

图 10　DARQN 网络结构图

利用注意力机制的优势在于，DRQN 获得了选择和集中处理较小的输入图像信息区域的能力，因此能够减少 DNN 中的参数数量和用于训练及测试网络的计算成本. 和 DRQN 相比，加入注意力机制后，LSTM 存储的数据不仅用于下一个动作的选择，也用于选择下一个注意力的区域. 除了提高计算速度，基于注意力机制的模型也增加了深度 Q 学习的可解释性，为研究者提供一个机会去观察到智能体的注意力集中在“哪”以及“什么”上. 实验表明，在一些 Atari 2600 游戏中 DARQN 的表现超过了 DQN 和 DRQN.

引入内部恐惧机制的 DQN. 在实际学习过程中，甚至在一些简单的学习环境中，DQN 会由于使用函数逼近而最终忘记经验，发生灾难性的错误. 同时只要继续训练，DQN 可能会周期性地重复那些灾难性错误，而这些错误是可以避免的. Lipton 等人[61]提出引入内部恐惧(Intrinsic Fear，IF)机制的 DQN，通过避免危险状态来缓解上述问题.

实现 IF 机制的关键，是通过监督学习来训练危

险模型(danger model),危险模型是一个二分类的神经网络,其结构除了输出层以外和 DQN 相同.设计危险模型的目的是为了预测在短时间内发生灾难的可能性.

引入 IF 机制后,DQN 的优化目标值变为

$$y^{\mathrm{IF}}=r+\max_{a'}Q(s',a';\theta_Q)-\lambda\cdot d(s';\theta_d) \quad (34)$$

其中,$d(s;\theta_d)$是危险模型,λ 是恐惧因子(fear factor),决定 IF 对 Q 函数更新的影响大小.

这项工作是解决源自 DRL 中使用函数逼近引起的人工智能(Artificial Intelligence,AI)安全问题迈出的第一步,但缺点是并没有预先假设可能会在状态空间中发生的危险的精确形式.

2.2.4 基于新提出的 RL 算法(归一化优势函数、多 Q 学习)的改进

基于归一化优势函数的 DQN. 无模型 RL 已经成功地应用于一系列具有挑战性的问题,并且能够扩展到解决大规模神经网络策略和值函数的学习问题.然而,无模型算法具有很高的抽样复杂度,特别是在使用高维函数逼近器时,往往会限制其对实际物理系统的适用性.所谓的抽样复杂度就是需要收集大量的数据和样本才能实现满意的训练效果.为了在连续控制任务中降低 DRL 的抽样复杂度和提高算法效率,Gu 等人[62]提出 Q 学习算法的一个变换形式:归一化优势函数(Normalized Advantage Function,NAF),并将它和 DNN 相结合,形成基于 NAF 的 DQN.

NAF 中的 Q 函数和优势函数分别定义如下:

$$Q(s,a;\theta^Q)=A(s,a;\theta^A)+V(s;\theta^V) \quad (35)$$

$$A(s,a;\theta^A)=-\frac{1}{2}(a-\mu(s;\theta^a))^{\mathrm{T}}\boldsymbol{G}(s;\theta^P)(a-\mu(s;\theta^a)) \quad (36)$$

其中,$\mu(s;\theta^a)=\arg\max_a Q(s,a;\theta^a)$,$\boldsymbol{G}(s;\theta^P)$是和状态有关的正定方阵,并由 $\boldsymbol{G}(s;\theta^P)=\boldsymbol{U}(s;\theta^P)\boldsymbol{U}(s;\theta^P)^{\mathrm{T}}$参数化表示,其中 $\boldsymbol{U}(s;\theta^P)$是元素来自于神经网络线性输出层的下三角矩阵.

实验表明,NAF 可以在一组模拟的机器人控制任务上显著提高性能,更快地学习并获得更精确的策略.

深度多 Q 学习. Q 学习是一种常用的 TD RL 算法,但当用 DNN 来做值函数逼近时,会出现不稳定的情况.为了克服这种不稳定性,Duryea 等人[63]基于 Q 学习和双 Q 学习算法提出一种新的 TD 算法:多 Q 学习(Multi Q-learning),并把它延伸到 DRL 领域,基于 DDQN 算法进一步提出深度多 Q 学习(Deep Multi Q-learning)算法.

深度多 Q 学习和 DDQN 不同的是,DDQN 中每经过一定次数迭代目标网络的参数会从在线 Q 网络中复制过来,而深度多 Q 学习则针对每个 Q 值使用一个独立的神经网络.这完全将动作的选择和评价独立起来(解耦),而 DDQN 则是实现了选择和评价过程之间的部分解耦.深度多 Q 学习的优化目标估计值为

$$Y_t^{\mathrm{DeepMultiQ}}=r+\gamma\frac{1}{n-1}\sum_{b=1,i\neq A}^{n}Q(s_{t+1},\arg\max_a Q(s_{t+1},a;\theta_t^A);\theta_t^b) \quad (37)$$

其中,θ_t^A是在线 Q 网络的参数,θ_t^b是每个 Q 值对应的独立神经网络的参数.

实验表明,结合深度多 Q 学习算法的神经网络比结合 Q 学习和双 Q 学习算法的神经网络稳定性和效率都要更好,深度多 Q 学习更好的稳定性也使它成为一个更具有普适性的算法,不需要调整参数就能完成不同种类的任务.

2.2.5 DQN 其它改进方法

DDQN[32]、优先经验回放[35]、竞争网络结构[36]、多步自举目标学习[64]、分类 DQN[65](Categorical DQN)和噪声 DQN[66](Noisy DQN)这些方法各自都可以提升 DQN 性能的某个方面.由于它们都着力于解决不同的问题,而且都构建在同一个框架上,所以能够被整合起来.Hessel 等人[67]对以上基于 DQN 的 6 种改进方法进行整合,提出一个新的变种:单智能体系统 Rainbow.

实验结果表明,在 57 个 Atari 游戏中,无论是数据效率还是最终性能,Rainbow 都超过 DQN 以及基于 DQN 的 6 种改进方法,达到新的最优性能.其中,用 700 万帧数据训练后 Rainbow 就已经媲美了 DQN 的最佳性能,用 4400 万帧数据训练后 Rainbow 就已超越所有 6 种改进方法的最终性能,并最终趋于平稳.此外,实验结果还显示,去除竞争网络结构和 DDQN 对 Rainbow 的性能影响不大,去除优先经验回放和 A3C 则对 Rainbow 性能下降的影响最大.

除了上面提到的,还有一些其它基于 DQN 的改进方法.Lee 等人[68]提出稀疏 Q 学习(sparse Q-learning)算法,并进一步把它扩展为稀疏深度 Q 学习(Sparse Deep Q-learning)算法.He 等人[69]基于 DQN 提出深度强化对手网络(Deep Reinforcement

Opponent Network，DRON). DRON 有两个学习模块，一个用来预测 Q 值，另一个用来推断对手策略. 实验表明，在模拟的足球游戏和流行的问答游戏中，DRON 比 DQN 展现了更优越的性能. Palmer 等人[70]基于宽容学习[71]的思想，将其应用到 DQN 中，提出宽容深度 Q 网络(Lenient Deep Q-Network，LDQN)，用来控制哪些状态转移样本更新 DQN. 他们还提出温度衰减计划(Temperature Decay Schedule，TDS)，将其应用到滞后深度 Q 网络[72](Hysteretic DQN，HDQN)中，提出计划滞后深度 Q 网络(Scheduled Hysteretic-DQN，SHDQN). 实验表明，在产生随机奖励的合作任务中 LDQN 和 SHDQN 都比 HDQN 性能要好. Qureshi 等人[73]提出多模态 DQN(Multimodal Deep Q-Network，MDQN)，让机器人能够通过试错的方法学习到类人的交往技巧. Bellemare 等人[74]提出新的具有局部一致性的 Q 函数优化保护算子，并用它代替 DQN 中的学习规则. 该算子的局部一致性可以使每个状态的动作之间的间隔增加，进而减小逼近和估计误差对贪婪策略的不良影响. 实验表明，加入该算子的 DQN 在 Atari 游戏中可以显著提高表现. Bellemare 等人[65]还提出分类 DQN，用 KL 散度(Kullback-Leibler divergence)的交叉熵项代替 DQN 中的损失函数，网络输出也不是每个动作的 Q 值，而是原子概率(atom probability). He 等人[75]将深度 Q 学习算法和约束优化方法相结合，以便实现更快的奖励传播. 在 Atari 游戏环境中该方法在训练时间和准确性方面有着显著的提高.

2.3　小结与分析

本节首先详细介绍了 DRL 领域的开山鼻祖 DQN，包括 DQN 所采用的两个关键技术：经验回放机制和目标网络以及 DQN 贡献的三个重要成果. 接着按照对 DQN 改进方式的不同，将基于 DQN 的各种改进版本分为训练算法的改进、神经网络结构的改进、引入新的学习机制的改进、基于新提出的 RL 算法的改进和其它改进. 表 1 总结了各种 DQN 改进方法的改进类别、解决的问题及意义和它们的实验结果.

表 1　DQN 改进方法的改进类别、解决问题和优缺点及意义

方法名称	改进类别	解决问题及意义	实验结果
双 DQN	训练算法的改进	解决 Q 学习中过高估计动作值函数的问题	Atari 游戏中的表现优于 DQN
示范深度 Q 学习	训练算法的改进	减少和环境的交互次数	大大加快学习过程，Atari 游戏中的表现优于 PDD DQN
对称 DQN	训练算法的改进	保证抽样效率	Cart-Pole 游戏中比 DQN 获得更多的分数
指导 DQN	训练算法的改进	解决机器人学习空中冰球击打的控制策略问题	在空中冰球击打的运动规划和控制问题上表现接近最优，完全解决了采用 DDQN 算法会出现的得分降低问题
最小二乘 DQN	训练算法的改进	学到丰富特征表示的同时保证学习过程的稳定，为未来线性 RL 和 DRL 的结合打下坚实的基础	Atari 游戏中的表现明显优于 DQN 和 DDQN
反向传播贝叶斯 Q 网络	训练算法的改进	开发了有效的冒险探索方法，解决了给定当前策略情况下 Q 值的不确定性问题	比使用 ε-贪婪的 DQN 等传统方法学习速度更快
引入内部惩罚信号的 DQN	训练算法的改进	解决 DRL 累积奖励过高估计的问题并减少抽样复杂度	Atari 游戏中的表现和采样效率上都优于 DQN 和 DDQN
深度乐观线性支持学习	训练算法的改进	解决多个目标之间相对重要性未知的高维多目标决策问题，是 DRL 首次在学习多目标策略问题中取得成功	学习多目标 MDP 问题有着很高的准确率
平均 DQN	训练算法的改进	减少 DRL 算法的不稳定性和可变性对其产生的负面影响，使训练过程更加稳定，并通过减少目标逼近误差来提高性能	Atari 游戏和 Gridworld 问题中增加 K 的大小可以在学习到更好的策略的同时减少 Q 值的过高估计
竞争网络结构	神经网络结构的改进	针对无模型 RL 提出一种新的神经网络结构	结合 DDQN 训练算法和优先经验回放技术实现的竞争网络结构的 Atari 游戏中的表现优于只采用优先经验回放的 DQN 和 DDQN
深度循环 Q 网络	神经网络结构的改进	解决实际应用中状态存在部分可观测性和噪声干扰的问题	Atari 游戏中取得和 DQN 相媲美的学习效果，但当用完整的观测信息进行训练，用部分观测信息评价时，DRQN 比 DQN 表现要差
基于文本的 LSTM-DQN	神经网络结构的改进	解决基于文本的游戏中存在语言障碍的问题，实现将文本描述映射为游戏状态语义的向量表示	两个游戏中的表现优于使用单词袋子模型、二元语法袋子模型做状态表示的基准方法
特定动作深度循环 Q 网络	神经网络结构的改进	提高了解决 RL 学习部分可观测问题的性能	Atari 游戏中的表现优于 DRQN

（续 表）

方法名称	改进类别	解决问题及意义	实验结果
自举 DQN	神经网络结构的改进	实现有效的冒险探索，将深度冒险探索与 DNN 相结合，对 DNN 的不确定性进行有效的估计并加快学习速度.计算上易处理，并且可以扩展到大规模并行系统	显著提高很多 Atari 游戏的学习速度和表现
增广对数先验深度 Q 学习	神经网络结构的改进	改进 RNN 生成序列的结构和质量，同时保留从数据中得到的信息以及样本的多样性	改进了生成序列所需的属性和结构，同时保留了从数据中学习到的信息
Gorila DQN	神经网络结构的改进	通过开发大量的计算资源来扩展如 DQN 的 DRL 算法	41 个 Atari 游戏中的表现都超过非分布式的 DQN，25 个游戏超过人类职业玩家的水平，同时也大大减少实现这些结果所需要的时间
优先经验回放	引入新机制的改进	考虑序列中各个转移元组的重要程度，更频繁地回放重要的转移元组	提高 DQN 和 DDQN 在游戏中的表现
注意力机制深度循环 Q 网络	引入新机制的改进	在 DRQN 结构中加入注意力机制	使 DRQN 获得了选择和集中处理较小的输入图像信息区域的能力，减少 DNN 中的参数数量和用于训练及测试网络的计算成本
引入内部恐惧机制的 DQN	引入新机制的改进	避免 DQN 训练过程中周期性地重复灾难性错误，这项工作是解决源自 DRL 中使用函数逼近引起的 AI 安全问题迈出的第一步	Cart-Pole 游戏中比 DQN 获得更多的分数
基于归一化优势函数的 DQN	基于新提出的 RL 算法的改进	在连续控制任务中降低 DRL 的抽样复杂度和提高算法效率	在一组模拟的机器人控制任务上显著提高 DQN 的性能，更快地学习并获得更精确的策略
深度多 Q 学习	基于新提出的 RL 算法的改进	克服 DNN 来做值函数逼近时出现的不稳定性	结合深度多 Q 学习算法的 DNN 比结合 Q 学习和双 Q 学习算法的 DNN 稳定性和效率都要更好

3 基于策略梯度的深度强化学习

基于策略梯度的 DRL 主要分为深度确定性策略梯度[76]（Deep Deterministic Policy Gradient，DDPG）、信赖域策略优化[77]（Trust Region Policy Optimization，TRPO）和异步优势行动者-评论家[64]（Asynchronous Advantage Actor-Critic，A3C）这三大类方法以及基于它们的一些改进方法.基于策略梯度的 DRL 具体分类如图 11 所示.

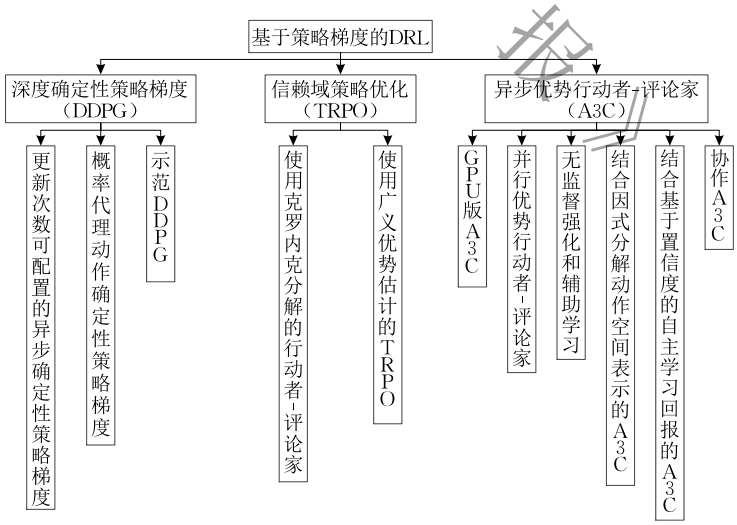

图 11 基于策略梯度的 DRL 分类示意图

3.1 策略梯度

策略梯度算法广泛应用在连续空间中的 RL 问题中，其主要思想是将策略 π 参数化表示为 π_θ，并计算出关于动作的策略梯度，然后沿着梯度的方向，不断地调整动作，逐渐得到最优策略.常见的策略梯度算法有 REINFORCE[78] 算法和行动者-评论家[79-81]（Actor-Critic，AC）算法等.

策略梯度中，策略分为随机性策略 $\pi_\theta(a|s)=P[a|s;\theta]$ 和确定性策略 $a=\mu_\theta(s)$.随机性策略对应的含义为，当前状态为 s 时，动作 a 满足参数为 θ 的

某个概率分布，因此相同状态也会对应不同的动作. 而确定性策略和随机性策略不同的是，对于一个确定性的策略 μ_θ 来说，每个状态对应唯一的动作.

和策略相对应的，策略梯度也分为随机性策略梯度(Stochastic Policy Gradient，SPG)和确定性策略梯度(Deterministic Policy Gradient，DPG). Sutton 等人[82]推导出 SPG 的公式：

$$\nabla_\theta L(\pi_\theta)=\int_S \rho^\pi(s)\int_A \nabla_\theta \pi_\theta(a|s)Q^\pi(s,a)\mathrm{d}a\mathrm{d}s$$
$$=E_{s\sim\rho^\pi,a\sim\pi_\theta}[\nabla_\theta \log\pi_\theta(a|s)Q^\pi(s,a)] \quad (38)$$

上述公式实际上是在假设条件 $s\sim\rho^\pi$ 和 $a\sim\pi_\theta$ 下，计算策略梯度函数在 $Q^\pi(s,a)$ 概率分布上的数学期望. 自上述公式提出以后，SPG 成为研究的重点，并且一般认为 DPG 是不存在的. 直到 Silver 等人[83]提出并证明了 DPG 的公式：

$$\nabla_\theta L(\mu_\theta)=\int_S \rho^\mu(s)\nabla_\theta \mu_\theta(s)\nabla_a Q^\mu(s,a)|_{a=\mu_\theta(s)}\mathrm{d}s$$
$$=E_{s\sim\rho^\mu}[\nabla_\theta \mu_\theta(s)\nabla_a Q^\mu(s,a)|_{a=\mu_\theta(s)}] \quad (39)$$

DPG 方法才得以流行.

3.2　AC 算法

3.2.1　随机性 AC 算法

AC 是得到广泛应用的基于策略梯度理论的 RL 算法. AC 网络结构如图 12 所示.

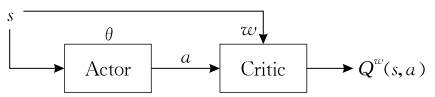

图 12　AC 网络结构图

AC 网络结构包括两个组成部分：行动者网络(Actor)和评论家网络(Critic). 输入状态 s，Actor 根据式(38)采用随机梯度下降方法更新随机策略 $\pi_\theta(s)$ 的参数 θ，并用 Critic 得到的估计的动作值函数 $Q^w(s,a)$ 代替式(38)中未知的真实动作值函数 $Q^\pi(s,a)$ 进行更新. Actor 输出的动作 a 和状态 s 一起输入到 Critic 中，Critic 使用如 TD 学习的策略评价算法来估计动作值函数 $Q^w(s,a)\approx Q^\pi(s,a)$，网络参数为 w. 随机性 AC 算法的整个学习过程即可描述为 Actor 产生动作，Critic 来评价 Actor 产生的动作的好坏，并生成 TD 估计误差来同时指导 Actor 和 Critic 的更新过程.

3.2.2　离策略随机性 AC 算法

经常采用从一个不同的行为策略 $\beta(a|s)\neq\pi_\theta(a|s)$ 中采样的方法来离策略估计 SPG. 当策略为离策略时，目标函数修改为目标策略的值函数 $V^\pi(s)$ 在行为策略的状态概率分布 $\rho^\beta(s)$ 上取平均的形式[84]：

$$L_\beta(\pi_\theta)=\int_S \rho^\beta(s)V^\pi(s)\mathrm{d}s$$
$$=\int_S\int_A \rho^\beta(s)\pi_\theta(a|s)Q^\pi(s,a)\mathrm{d}a\mathrm{d}s \quad (40)$$

对上式两边求微分，并丢弃和 $\nabla_\theta Q^\pi(s,a)$ 相关的一项，得到离策略 SPG[84]：

$$\nabla_\theta L_\beta(\pi_\theta)\approx\int_S\int_A \rho^\beta(s)\nabla_\theta \pi_\theta(a|s)Q^\pi(s,a)\mathrm{d}a\mathrm{d}s$$
$$=E_{s\sim\rho^\beta,a\sim\beta}\left[\frac{\pi_\theta(a|s)}{\beta_\theta(a|s)}\nabla_\theta \log\pi_\theta(a|s)Q^\pi(s,a)\right] \quad (41)$$

离策略随机性 AC 算法使用行为策略 $\beta_\theta(a|s)$ 来产生轨迹样本. Critic 利用这些样本并采用梯度时间差分学习[85](gradient temporal-difference learning)方法，以离策略的方式估计状态值函数 $V^v(s)\approx V^\pi(s)$. Actor 根据式(41)采用随机梯度下降法来更新策略参数 θ，计算时，用 TD 误差 $\delta_t=r_{t+1}+\gamma V^v(s_{t+1})-V^v(s_t)$ 代替未知的动作值函数 $Q^\pi(s,a)$，可以证明 δ_t 能够逼近真实的梯度. Actor 和 Critic 都使用重要性采样权重 $\frac{\pi_\theta(a|s)}{\beta_\theta(a|s)}$ 来调整目标函数，因为事实上动作是根据 π 而不是 β 来选择的.

3.2.3　离策略确定性 AC 算法

类似于离策略随机性 AC 算法，离策略确定性 AC 算法的目标函数的形式为目标策略的值函数 $V^\mu(s)$ 在行为策略 $\rho^\beta(s)$ 的概率分布上取平均[83]：

$$L_\beta(\mu_\theta)=\int_S \rho^\beta(s)V^\mu(s)\mathrm{d}s$$
$$=\int_S \rho^\beta(s)Q^\mu(s,\mu_\theta(s))\mathrm{d}s \quad (42)$$

则离策略 DPG 为

$$\nabla_\theta L_\beta(\mu_\theta)\approx\int_S \rho^\beta(s)\nabla_\theta \mu_\theta(a|s)Q^\mu(s,a)\mathrm{d}s$$
$$=E_{s\sim\rho^\beta}[\nabla_\theta \mu_\theta(s)\nabla_a Q^\mu(s,a)|_{a=\mu_\theta(s)}] \quad (43)$$

在计算 $\nabla_\theta J_\beta(\mu_\theta)$ 时，和处理随机性策略时一样，用可微的动作值函数 $Q^w(s,a)$ 来代替上式中的真实动作值函数 $Q^\mu(s,a)$. Critic 利用 $\beta(a|s)$ 产生的样本来估计动作值函数 $Q^w(s,a)\approx Q^\mu(s,a)$. 综上所述可以得到离策略确定性 AC 算法的更新过程：

$$\delta_t=r_t+\gamma Q^w(s_{t+1},\mu_\theta(s_{t+1}))-Q^w(s_t,a_t) \quad (44)$$
$$w_{t+1}=w_t+\alpha_w\delta_t\nabla_w Q^w(s_t,a_t) \quad (45)$$
$$\theta_{t+1}=\theta_t+\alpha_\theta\nabla_\theta \mu_\theta(s_t)\nabla_a Q^w(s_t,a_t)|_{a=\mu_\theta(s)} \quad (46)$$

和离策略随机性 AC 算法的 Actor 和 Critic 都采用重要性采样的方法不同，与 SPG 相比，DPG 没有对动作的积分，因此在 Actor 更新中可以避免使用重要性采样过程. 同时通过使用 Q 学习来估计动

作值函数,Critic 同样避免了使用重要性采样过程.

3.3 DDPG

Lillicrap 等人[76]将 DQN 的思想应用到连续动作域中,提出一种基于 DPG 和 AC 的无模型算法:DDPG.

DQN 只能处理离散和低维的动作空间,但许多情况尤其是物理控制任务有着连续和高维的动作空间. DQN 不能直接应用到连续域,因为它需要找到使动作值函数最大的动作,在动作取连续值的情况下,每一步都需要迭代优化过程.

将像 DQN 这样的 DRL 方法运用于连续域的一个有效的方法就是把动作空间进行离散化,但这会带来一个显著的问题就是维数灾难:动作的数量随着自由度的增加而呈指数倍增长,这将会给训练带来很大困难. 此外,单纯地对动作空间进行离散化将会去除掉关于动作域结构的信息.

DDPG 借鉴 DQN 技术,采用经验回放机制和单独的目标网络,减少数据之间的相关性,增加算法的稳定性和鲁棒性. 虽然 DDPG 借鉴了 DQN 的思想,但要直接将 Q 学习应用到连续动作空间是不可能的,因此 DDPG 采用的是基于 DPG 算法的 AC 方法.

DDPG 采用的经验回放机制和 DQN 完全相同,但目标网络的更新方式和 DQN 相比略有差异. DQN 的目标网络是隔 N 步和 Q 网络同步一次,DDPG 中 Actor 和 Critic 各自的目标网络参数 θ^- 和 w^- 则是通过变化较慢的方式更新,而不是直接复制参数,以此进一步增加学习过程的稳定性:

$$\theta^- = \tau\theta + (1-\tau)\theta^- \tag{47}$$

$$w^- = \tau w + (1-\tau)w^- \tag{48}$$

在连续动作空间中学习的主要挑战是有效地实现冒险探索,考虑到 DDPG 是离策略算法,通过额外增加一个噪声项 N 来构建一个探索策略 μ':

$$\mu'(s_t) = \mu_\theta(s_t) + N \tag{49}$$

综上所述,DDPG 算法 Actor网络参数 θ 和 Critic 网络参数 w 的更新公式分别如式(45)、(46)和式(50)所示:

$$\delta_t = r_t + \gamma Q^{w^-}(s_{t+1}, \mu'^{-}_{\theta}(s_{t+1})) - Q^{w}(s_t, a_t) \tag{50}$$

如图 13 所示,DDPG 的网络结构和 DQN 相比除了值网络之外还多了一个策略网络,策略网络的输出为 $\pi(s)$. 同时 DQN 输入仅是视频帧而不需要额外输入动作,每个离散动作都有一个单独的输出单元,其值网络的输出是每个动作所对应的 Q 值. 而 DDPG 的值网络则是输入视频帧后再通过 CNN 得到特征,再输入动作 a,最后输出 Q 值.

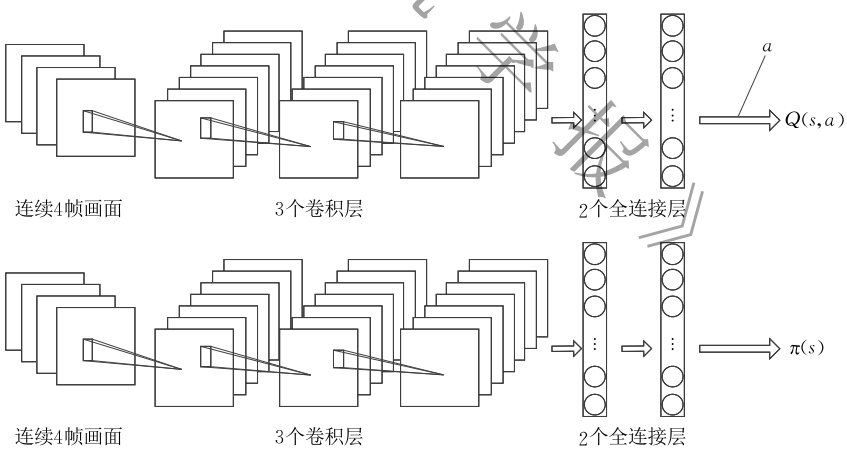

图 13 DDPG 网络结构图

实验表明使用相同的学习算法、网络结构和参数,DDPG 能够鲁棒地解决超过 20 个物理仿真任务,包括一些经典问题,比如灵巧性操纵、腿部运动和汽车驾驶等. DDPG 能够发现一些好的策略,这些策略的性能能够与传统的规划算法相媲美. 同时对于许多任务来说,DDPG 能够直接以原始像素作为输入端到端地学习策略. DDPG 在这些任务中求解的时间要比 DQN 在 Atari 问题中所用时间更少. 这表明,如果给更多的计算时间,DDPG 可能会解决更大规模的问题.

DDPG 一个关键的优点就是它的简洁性,这使它能够很容易地应用到更复杂的问题和更大的网络. 不过 DDPG 还有一些局限性,最明显的就是和大多数无模型的 RL 方法一样,DDPG 需要大量的训练时间段才能收敛.

3.4 DDPG 的改进方法

3.4.1 更新次数可配置的异步 DPG

DL 和 RL 方法近来被用于解决各种连续控制

问题.这些技术的一个最显著的应用场景就是机器人的灵活操纵任务,这很难用传统的控制理论或手动方法解决.这类任务的一个例子就是抓住一个对象,并将其精确地堆叠在另一个对象上.在现实世界中解决这一和实际相关的问题是机器人研究领域的一个重要和长期目标.针对此问题,Popov 等人[86]从两方面扩展 DDPG 算法,提出更新次数可配置的异步 DPG(Asynchronous DPG-R,ADPG-R).

首先,在 DDPG 算法上,增加算法执行过程中智能体和环境每一次交互的更新迭代学习次数,并将这种更新次数可配置的 DDPG 命名为 DPG-R.实验表明这一方法可以显著减少学到成功策略所需要的经验数据即与环境交互的次数.对于更改智能体和环境每一次交互的更新次数会有如此显著的效果的原因,一种可能的解释认为:这类似于监督学习中不充分的训练会导致对已经收集训练数据的策略网络和值网络的欠拟合.和监督学习中数据集通常是固定的不同的是,由于策略网络用于选择冒险探索策略,因此选择冒险策略的质量直接反馈到数据获取过程中,进而影响了未来网络训练所使用的数据的质量.

基于 DPG-R,然后借鉴 A3C 的思想,通过并行执行训练过程,并使 Actor 和 Critic 网络共享训练参数,即得到 ADPG-R 算法.实验表明,利用该算法可以学到控制策略来鲁棒地抓取和堆叠物体.

3.4.2　概率代理动作 DPG

Wang 等人[87]基于替代智能体-环境接口(Surrogate Agent-Environment Interface,SAEI)提出概率替代动作确定性策略梯度(Probability Surrogate Action Deterministic Policy Gradient,PSADPG)算法,它是 DDPG 算法的改进版本,能够连续控制离散的动作.

PSADPG 和 DDPG 的不同在于它用随机向量 $\boldsymbol{p}$ 来替代 DDPG 中的动作 a,再由这个随机向量 $\boldsymbol{p}$ 上抽样得到的样本用于和环境交互的动作计算.实验表明,在离散控制任务 Acrobot 中 PSADPG 媲美了 DQN 的表现.

3.4.3　示范 DDPG

要解决高维连续控制问题,RL 需要为智能体的每个状态提供一个平滑可变的奖励信号,奖励信号通常通过编码函数实现,例如到一个目标点的笛卡尔距离.虽然许多任务可以很容易地由终止目标状态定义奖励值函数,但是如何根据问题给出具有很好轨迹的奖励值函数是非常困难的.为了解决上述问题,Večerík 等人[88]用示范来替代具有很好轨迹的奖励值函数,提出示范 DDPG(DDPG from Demonstrations,DDPGfD)算法,它是一种通用的无模型离策略算法,实验表明在 4 个模拟机器人按指定位置插入物体的任务中,DDPGfD 比 DDPG 效果更好.

DDPGfD 和 DDPG 有以下 5 点不同:

(1)训练开始之前,示范转移(demonstration transitions)数据被放到回放缓冲区,并且在回放缓冲区中永远保留这些示范转移数据.

(2)示范转移数据和智能体数据之间的采样比率通过优先重放机制自动调整,该机制能够有效地传播奖励信息,对奖励值稀疏问题很有效.

(3)单步和 n 步奖励值都用来更新 Critic 的损失函数.

(4)每一次和环境交互的步骤,做多次学习更新.如果每一次和环境交互的步骤只更新一次,那么每一次示范转移数据被抽样的次数只是和 mini-batch 的次数一样多,因此需要在收集更新的数据和做更多的学习之间求得一个平衡.如果数据陈旧,来自回放缓冲区的样本不再代表当前策略所要经历的状态分布,这可能导致学习的策略和 Q 值的发散.但由于需要数据效率即从之前的交互中学习,因此采取多次使用每个转移的方法.

(5)Actor 和 Critic 两个网络的损失函数中,分别增加两个网络参数的 L2 正则化项,因此最终的 Critic 损失函数和 Actor 梯度为

$$L_{\text{Critic}}(w)=L_1(w)+\lambda_1 L_n(w)+\lambda_2 L_{\text{reg}}^{\text{C}}(w) \tag{51}$$

$$\nabla_\theta L_{\text{Actor}}(\theta)=-\nabla_\theta J(\theta)+\lambda_2\nabla_\theta L_{\text{reg}}^{\text{C}}(\theta) \tag{52}$$

3.4.4　DDPG 其它改进方法

虽然 DRL 在连续状态和动作空间中应用广泛,但是直到 Hausknecht 等人[89]对 DDPG 算法进行扩展,才第一次成功地在结构化(参数化)的连续动作空间中使用 DNN. Hausknecht 等人对 DDPG 算法进行扩展,提出三种限制 Critic 动作空间梯度的上下界的新的方法,以保证参数在一定范围内变化,并着重于研究在模拟的机器人足球世界杯中的 DRL 问题,其具有一组离散动作类型,每一个动作类型都用连续变量进行参数化.实验表明,效果最好的智能体比 2012 年机器人足球世界杯冠军更可靠地取得进球.虽然没能解决更具挑战性的任务,比如在守门员面前进球或和队友配合,但仍旧在学习复

杂的机器人世界杯智能体研究方面迈出重要一步.

Bruin 等人[90]将 DDPG 和经验回放方法结合,提出一种在两种经验回放数据库上使用不同重写策略的方法,以确保用于训练的经验分布处在策略分布和均匀分布之间.其中一个数据库以先进先出(First In First Out)的方式重写经验,另一个数据库以确保在状态-动作空间上得到的经验分布尽可能地接近均匀分布的方式重写经验.通过在两个数据库上对经验数据进行抽样,可以更安全地使用 DNN 作为 RL 的函数逼近器.该方法减少了持续进行冒险探索的需要,并提高了训练策略的泛化性能.

3.5 TRPO

Schulman 等人[77]提出一种保证单调改进的策略优化迭代规划算法,并通过一系列的近似来推出理论公式和实际可行的算法:TRPO. TRPO 要解决的问题就是通过引入由 KL 散度定义的信赖域约束,来选取合适的步长,保证策略的优化总是朝着不变坏的方向进行.

MDP 过程的无限时域奖励数学期望定义为

$$\eta(\tilde{\pi})=E_{s_0,a_0,\cdots}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t)\right] \tag{53}$$

新的和旧的策略的无限时域奖励函数之间的关系为[91]

$$\eta(\tilde{\pi})=\eta(\pi)+E_{s_0,a_0,\cdots\sim\tilde{\pi}}\left[\sum_{t=0}^{\infty}\gamma^t A_\pi(s_t,a_t)\right] \tag{54}$$

其中,$E_{s_0,a_0,\cdots\sim\tilde{\pi}}[\cdots]$表示动作以$a_t\sim\tilde{\pi}(\cdot|s_t)$方式采样.令 ρ_π为状态 s 折扣访问频率:

$$\rho_\pi(s)=P(s_0=s)+\gamma P(s_1=s)+\gamma^2 P(s_2=s)+\cdots \tag{55}$$

其中,ρ_0为初始状态 s_0的概率分布,假定初始状态之后的动作都是根据策略 π 来选择.则式(54)可以重新表示为

$$\begin{aligned}\eta(\tilde{\pi})&=\eta(\pi)+\sum_{t=0}^{\infty}\sum_s P(s_t=s|\tilde{\pi})\sum_a\tilde{\pi}(a|s)\gamma^t A_\pi(s,a)\\&=\eta(\pi)+\sum_s\sum_{t=0}^{\infty}\gamma^t P(s_t=s|\tilde{\pi})\sum_a\tilde{\pi}(a|s)A_\pi(s,a)\\&=\eta(\pi)+\sum_s\rho_{\tilde{\pi}}(s)\sum_a\tilde{\pi}(a|s)A_\pi(s,a)\end{aligned} \tag{56}$$

由于 $\tilde{\pi}$ 不知道,状态 s 折扣访问频率$\rho_{\tilde{\pi}}(s)$对 $\tilde{\pi}$ 的依赖性将导致很难对式(56)直接求解优化,因此引入对 $\eta(\tilde{\pi})$的局部近似 $L_\pi(\tilde{\pi})$:

$$L_\pi(\tilde{\pi})=\eta(\pi)+\sum_s\rho_\pi(s)\sum_a\tilde{\pi}(a|s)A_\pi(s,a) \tag{57}$$

为了使用便于优化的 $L_\pi(\tilde{\pi})$,引入下面的不等式:

$$\eta(\tilde{\pi})\geq L_\pi(\tilde{\pi})-CD_{KL}^{\max}(\pi,\tilde{\pi}) \tag{58}$$

$$C=\frac{4\varepsilon\gamma}{(1-\gamma)^2} \tag{59}$$

其中,$D_{KL}^{\max}(\pi,\tilde{\pi})=\max_s D_{KL}(\pi(\cdot|s)\|\tilde{\pi}(\cdot|s))$,$\varepsilon=\max_s|E_{a\sim\pi'(a|s)}[A_\pi(s,a)]|$.

令 $M_i(\pi)=L_{\pi_i}(\pi)-CD_{KL}^{\max}(\pi_i,\pi)$,则如果在每一次迭代都取 M_i的最大值,可以保证 η 是单调非减的.以上思想可以总结为

$$\max_\theta[L_{\theta_{\text{old}}}(\theta)-CD_{KL}^{\max}[\theta_{\text{old}},\theta]] \tag{60}$$

实际使用中,若使用惩罚因子 C,则迭代步长很小,因此 TRPO 给新策略和旧策略之间的 KL 散度施加一个约束,即信赖域约束,因此上述思想可以进一步转化为求解问题:$\max_\theta L_{\theta_{\text{old}}}(\theta)$,并满足约束条件 $D_{KL}^{\max}[\theta_{\text{old}},\theta]\leq\delta$.用平均 KL 散度 $\overline{D}_{KL}^{\rho}(\theta_1,\theta_2)=E_{s\sim p}[D_{KL}(\pi_{\theta_1}(\cdot|s)\|\pi_{\theta_2}(\cdot|s))]$近似最大 KL 散度,约束条件变为 $\overline{D}_{KL}^{\rho_{\theta_{\text{old}}}}(\theta_{\text{old}},\theta)\leq\delta$.

然后继续对目标函数和约束函数做进一步的近似,对 $L_{\theta_{\text{old}}}$展开后,目标函数变为

$$\max_\theta\sum_s\rho_{\text{old}}(s)\sum_a\pi_\theta(a|s)A_{\text{old}}(s,a).$$

首先用 $\frac{1}{1-\gamma}E_{s\sim\rho_{\theta_{\text{old}}}}[\cdots]$代替目标函数中的$\sum_s\rho_{\theta_{\text{old}}}(s)[\cdots]$,然后用 $Q_{\theta_{\text{old}}}$代替 $A_{\theta_{\text{old}}}$,最后利用重要性抽样过程对动作概率分布进行变换,即 $\sum_a\pi_\theta(a|s_n)A_{\theta_{\text{old}}}(s_n,a)=E_{a\sim q}\left[\frac{\pi_\theta(a|s)}{q(a|s_n)}\right]$,其中 q 表示抽样概率分布.综上所述,最后 TRPO 算法的优化问题变为

$$\max_\theta E_{s\sim\rho_{\theta_{\text{old}}},a\sim q}\left[\frac{\pi_\theta(a|s)}{q(a|s)}Q_{\theta_{\text{old}}}(s,a)\right]$$

$$\text{满足 } E_{s\sim\rho_{\theta_{\text{old}}}}[D_{KL}(\pi_{\theta_{\text{old}}}(\cdot|s)\|\pi_\theta(\cdot|s))\leq\delta] \tag{61}$$

然后需要解决的问题就是如何通过蒙特卡洛抽样计算来代替数学期望求解,以及估计 Q 值的问题.有两种抽样方法:一种是单路径法(single path),该方法经常用于无限时域策略梯度估计[92];另一种方法是多路径法(vine),该方法大多用于策略迭代[93-94].

通过分析和证明,策略迭代、策略梯度以及自然策略梯度[95](natural policy gradient)都是 TRPO 的特例.

实验表明,TRPO 对于优化像神经网络这样的大型非线性策略十分有效.TRPO 在机器人学习游

泳、跳跃、行走和 Atari 游戏等很多任务中都有着鲁棒的性能.

3.6 TRPO 的改进方法

Wu 等人[96]利用最近提出的克罗内克因式分解曲率近似[97-98](Kronecker-Factored Approximated Curvature,K-FAC)方法近似求取 AC 更新的自然梯度,基于 TRPO 提出一种高效抽样、计算成本低的信赖域优化方法:使用克罗内克分解的行动者-评论家方法(Actor Critic using Kronecker-Factored Trust Region,ACKTR).它也是一种从原始的像素输入中直接学习连续控制任务和离散控制策略的方法.实验结果表明,该方法在 Atari 游戏和 MuJoCo 环境中,比优势行动者-评论家[64](Advantage Actor-Critic,A2C)和 TRPO 获得更好的奖励和更高的抽样效率.这里,抽样效率是指抽得的样本被接受的概率高低,如果蒙特卡洛抽得的样本大多被拒绝,则抽样过程无效.

策略梯度是累积奖励梯度的无偏估计,因此策略梯度方法可直接通过优化累积奖励来求解最优参数,并且由于可以直接与神经网络等非线性函数逼近器一起使用,策略梯度方法是 RL 领域一种很有吸引力的方法.不过使用该方法有两个主要问题:一是要获得满意的估计结果,所需的样本数量很大;二是在输入数据不平稳的情况下很难获得稳定的策略改进.针对第一个问题,Schulman 等人[99]提出一种新的策略梯度估计方法:广义优势估计(Generalized Advantage Estimation,GAE),大幅度地降低了策略梯度估计的方差.针对第二个问题,Schulman 等人[99]应用 TRPO 方法来训练神经网络.实验结果表明此方法在一些极具挑战性的 3D 运动任务上取得很好的效果,包括学习两足和四足模拟机器人的跑步步法等.

3.7 A3C

Mnih 等人[64]提出一个概念上简单且轻量级的 DRL 框架,并使用异步梯度下降法优化 DNN 控制器.其中将该框架和四种标准的 RL 算法:单步 Sarsa、单步 Q 学习、n 步 Q 学习和优势行动者-评论家相结合形成各自的异步变种,其中效果最好的方法是 A3C 算法.

DNN 为有效地执行 RL 算法提供丰富的表示学习能力.由于将二者结合在一起算法不稳定,人们提出很多的解决方案,这些方法有一个共同思想是:在线 RL 智能体所用的观测数据序列是不稳定的,并且在线 RL 更新序列存在很大的相关性的.通过将智能体的数据存储在经验回放记忆中,数据可以成批处理或者从不同的时间步长中随机抽样得到.这种方法减少了学习过程的不平稳性,并且去除了更新序列的相关性,但是与此同时也将方法限制为只能用于离策略 RL 中.

基于经验回放的 DRL 算法已经在具有挑战性的领域中,如 Atari 游戏中,取得了空前的成功.然而,经验回放机制有两个缺点:在每次的实际交互中,基于经验回放的 DRL 算法都要使用很大的记忆存储单元和计算资源,并且它只能用于离策略 RL 学习,但离策略 RL 学习只能用旧策略生成的数据进行参数更新.

针对上述问题,Mnih 等人提出另一种思路来代替经验回放机制,即创建多个智能体,在多个环境中异步、并行地执行和学习.具体地,首先借鉴 Gorila[59]框架使用异步的 actor-learner 对,但是不再使用不同的机器节点和参数服务器,而是在一台机器上使用多核 CPU 的多线程,这样也就节省了用于发送梯度和参数的通讯费用,并能够使用无锁的方式[100](Hogwild!)进行训练.其次采用并行的多个 actor-learner 探索了环境的不同部分,而且可以在每个 actor-learner 中采用不同的探索策略,使探索策略具有多样性.通过在不同的线程上采用不同的探索策略,不同的 actor-learner 对通过在线并行参数更新过程,减少了数据在时间上的相关性.因此不需要经验回放机制也可以实现稳定的学习过程.

这种方法除了使学习过程稳定之外,还有两个好处.一是减少训练时间,这种方法和线程数近似成线性关系;二是由于不再依赖于经验回放,就可以采用如 Sarsa 和行动者-评论家在线 RL 算法来稳定地训练神经网络.

和 DDPG 算法类似,A3C 算法中每执行 $t_{\max}$ 个动作之后或者到达终止状态策略时,策略和值函数更新一次.Actor 网络的梯度为 $\nabla_{\theta'}\log\pi(a_t\mid s_t;\theta')A(s_t,a_t;\theta',\theta'_v)$,其中优势函数的估计值为

$$\mathcal{A}(s_t,a_t;\theta',\theta'_v)=\sum_{i=0}^{k-1}\gamma^i r_{t+i}+\gamma^k V(s_{t+k};\theta'_v)-V(s_t;\theta'_v) \tag{62}$$

这里,θ' 是策略参数,θ'_v 是状态值函数的参数,不同状态对应不同的 k,k 取值的上界为 $t_{\max}$.

A3C 网络结构中使用了 CNN,其中 CNN 的一个软最大输出为 $\pi(a_t\mid s_t;\theta)$,另一个线性输出为值

函数 $V(s_t;\theta_v)$，其余层由值网络和策略网络共享.除此以外，在目标函数关于策略参数 θ' 的梯度中增加策略 π 的熵正则化项 $\beta\nabla_{\theta'}H(\pi(s_t;\theta))$，其中 H 是熵避免过早收敛到次优确定性策略，改进 A3C 的冒险探索效果.在优化方法上，采用基于均方根传播[①](Root Mean Square Propagation，RMSProp)方法的一种变体.上述整个过程大致如图 14 所示.

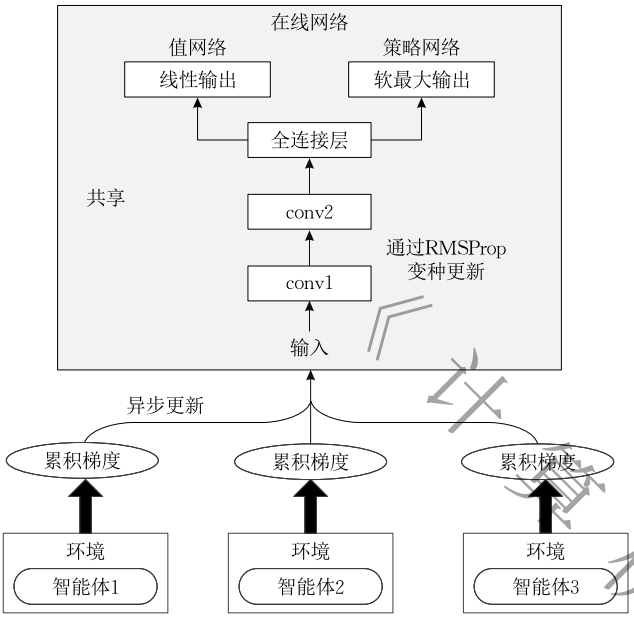

图 14 A3C 训练过程示意图

以前的 DRL 方法依赖于像 GPU 这样专门的硬件或者大规模分布式结构，A3C 则是运行在一台拥有标准多核 CPU 的机器上.实验表明，当把 A3C 应用到 Atari 2600 的各种游戏中时，A3C 比以前基于 GPU 的算法用了更少的时间，比大规模分布式方法用了更少的存储和计算资源，却取得了更好的效果.A3C 也在连续运动控制问题上取得了成功，包括赛车游戏 TORCS、连续电机控制任务物理模拟器 MoJoCo 和随机 3D 迷宫导航任务 Labyrinth.A3C 在以上任务中取得的成功以及它训练前馈和循环神经网络中的智能体的能力，使 A3C 成为迄今为止最通用和最成功的 DRL 算法.

虽然不用经验回放机制也能使训练过程稳定，但是这不意味着经验回放机制没用.将回放机制应用到异步 RL 框架中，能够通过重新使用旧数据来显著提高数据效率，这会在一些问题学习中实现更快的训练速度，比如 TORCS，该问题中和环境交互的费用要比 A3C 结构模型更新的费用更昂贵，因此虽然采用经验回放机制会提高后者的费用，却能减少前者的费用，是可以考虑的方法.此外，通过改进优势函数的估计方式以及一些对神经网络结构的补充改进如竞争网络结构等，可能会进一步提高 A3C 的学习效果.

3.8 A3C 的改进方法

3.8.1 GPU 版 A3C

Babaeizadeh 等人[101]提出 A3C 的替代结构：GPU 版 A3C(GA3C).此方法强调对 GPU 的利用来增加每秒生成和处理的训练数据数量.

GA3C 的主要组成部分包括在 GPU 上训练和预测的 DNN，以及拥有以下 3 个组成部分的多线程 CPU 结构：

(1)智能体(Agent).和 A3C 中的智能体功能相同：根据所学策略来选择动作，并收集样本以进行进一步的训练.不同的是，每个智能体不再需要复制一份模型，只需要在每次选择动作之前，将当前的状态作为请求加入预测队列(Prediction Queue)中.动作执行一定步数后，将倒退算出的总回报以及这些步数的经验数据加入到训练序列(Training Queue)中；

(2)预测器(Predictor).将预测队列中排队的预测请求样本取出队列，并把它们加入 GPU 中的 DNN 模型中.当预测完成时，预测器将预测的策略结果返回给相应的智能体.为了减少延迟，一个或多个预测器可以同时运行；

(3)训练器(Trainer).将训练序列中请求样本取出队列，并把它们加入到 GPU 中用于模型更新.同样地，为了减少延迟，可以同时运行多个预测器.

研究结果表明 GA3C 比用 CPU 实现 A3C 收敛速度更快，以更短的时间实现当今最高的学习水平，也是当前 RL 领域针对各种游戏任务最好的方法.不过 GA3C 仍然存在以下两个问题：

(1)CPU 中的智能体、预测器和训练器三个部分之间需要协调各自的数量；

(2)可能存在策略延迟，即用于生成当前样本的策略并不是当前要更新的策略，这样的后果就是算法可能不稳定.

3.8.2 并行优势行动者-评论家

Clemente 等人[102]提出一种灵活的并行框架，可用于在线策略、离策略、基于值和基于策略的算法.并基于此框架在 GPU 上实现了并行优势行动者-评论

① Divide the gradient by a running average of its recent magnitude. https://zh. coursera. org/learn/neuralnetworks/lecture/YQHki/rmsprop-divide-the-gradient-by-a-running-average-of-its-recent-magnitude 2017,4,21

家(Parallel Advantage Actor-Critic，PAAC)算法，该算法在 Atari 2600 中比 A3C 和 GA3C 需要更少的训练时间就能达到最先进的表现.

3.8.3　无监督强化和辅助学习

DRL 智能体通过直接最大化累积奖励达到了最先进的学习效果.然而学习环境中可能还包含更广泛的各种可用的训练信息.Jaderberg 等人[103]基于 A3C 框架，引入两种辅助任务提出无监督强化和辅助学习(UNsupervised REinforcement and Auxiliary Learning，UNREAL)算法.

为了促进更快的训练、更鲁棒的学习并最终取得更好的效果，作者在训练 A3C 的同时引入两种辅助任务，分别是控制任务和奖励预测任务.其中控制任务包括以下两类：

(1)像素控制.像素控制是指训练智能体使其学到让输入图像像素变化最大的策略，因为这种变化往往对应环境中发生的重要事件；

(2)隐藏层单元激活控制.因为策略网络或值网络用来学习提取和任务相关的高级环境特征[16,104-105]，而这对于智能体的学习十分有用，因此任何智能体神经网络的隐藏单元被激活本身就是一个辅助的奖励.此类控制任务目的就是最大程度激活隐藏层的每一个单元.

除了学习环境的动态过程之外，智能体必须学会如何让整体奖励最大.为了学习使得奖励最大化的策略，智能体需要能够识别获得高奖励的状态的特征，这将有利于学习到好的值函数，反过来也会有利于策略的学习.

然而在很多环境中奖励是稀疏的，这意味着需要很长时间来实现特征提取的过程，以便识别出奖励开始所对应的那个状态.为此我们引入另一种辅助任务：奖励预测.它以使用历史连续多帧的图像输入来预测下一步的回馈值作为训练目标，让神经网络具有更好的表达能力.

除了以上两种辅助任务之外，还利用经验回放机制来进行值函数回放的任务，即从行为策略分布中重新采样最近的历史序列，除了 A3C 中在线策略值函数回归之外又执行额外的值函数回归，进一步提高了训练速度.实验表明，UNREAL 算法在 Atari 游戏上取得人类水平 8.8 倍的表现.同时在第一视角的 3D 迷宫任务中达到人类专家 87%的水平，在学习速度、参数设置的鲁棒性和最终的表现上也都超过了 A3C.

除了上述工作之外，Li 等人[106]以及 Mirowski 等人[107]也都引入辅助任务来改进 DRL 智能体所学习到的特征表示，并因此提高了这些智能体的学习速度和最终性能.

3.8.4　A3C 其它改进方法

Sharma 等人[108]提出一种新的框架：因式分解动作空间表示(Factored Action space Representation，FAR)，用于在离散的动作空间中分解策略和动作值函数.FAR 将策略和值函数分解为独立的分量，并使用神经网络的独立输出层对其进行建模.FAR 和 A3C 结合就形成 FARA3C 算法，实验结果表明在 Atari 游戏 14 个任务中的 9 个表现都超过 A3C.Sharma 等人[109]提出 λ 步回报的推广：基于置信度的自主学习回报(Confidence-based Autodidactic Returns，CAR)方法，该方法用之前的 n 步奖励值的预测回报值的置信度作为权值，对 n 步奖励值的预测回报值加权，将这些回报加权平均求得最终回报值.将该方法和 A3C 结合形成 CARA3C 算法.实验结果表明，该算法在 Atari 游戏 22 个任务中的 18 个表现都超过 A3C.

Wang 等人[110]在 A3C 的基础上，利用 Retrace 算法[111]并结合经验回放机制、截断重要性采样方法[112](truncated importance sampling)和竞争网络，提出一个稳定的 DRL 模型.此外其他一些学者将基于值函数和基于策略的 RL 相结合，对 A3C 算法进行了改进[113-115].

除了自主学习之外，通过对所学知识的总结、与同伴交流以及随后利用不同来源的知识来辅助当前的学习目标，人类的学习过程得到极大的提高.这种协作学习的过程确保了知识的共享，不断的改良，从不同的角度总结出对问题的更深刻的理解.在人类协作学习过程的启发下，Lin 等人[116]提出一种协作 DRL(Collaborative Deep Reinforcement Learning，CDRL)框架，在异构学习智能体之间进行自适应的知识转移.提出的 CDRL 利用一个新的深度知识提取[117](deep knowledge distillation)方法来学习所有任务的共同表示，并将深度知识提取方法引入智能体的在线训练中，提出协作 A3C(collaborative Asynchronous Advantage Actor-Critic，cA3C)算法，在 OpenAI gym 环境中取得很好的效果.

3.9　基于策略梯度的其它 DRL 方法

许多针对连续控制任务的 RL 方法都是基于通过最大化逼近后的动作值函数或 Q 函数来更新策略函数.然而 Q 函数也依赖于策略，而这种依赖常常导致不稳定的策略学习.为了克服这个问题，

Tangkaratt 等人[118]提出一种专注于稳定策略学习的方法,但不像现有的方法以贪婪的方式来最大化 Q 函数.为此在最大化 Q 函数的时候给新策略的 KL 散度添加一个上界限制.此外还给新策略的熵添加一个下界,以保持其探索行为.这些约束所带来的稳定性能够使此方法性能更好并避免了局部最优.此方法可以看作是 DPG 的一种变体,实验表明,它比 DDPG 和基于 NAF 的 DQN 有着更稳定的性能.

针对状态部分可观测的连续控制问题,Heess 等人[119]通过使用基于时间的反向传播方法训练 RNN,将 DPG 和随机值梯度[120](Stochastic Value Gradient,SVG)这两种算法分别扩展为循环 DPG(Recurrent DPG,RDPG)算法和循环 SVG(Recurrent SVG,RSVG)算法.它们能够解决各种需要考虑时间记忆特性的物理控制问题,包括来自于噪声传感器的信息的短期集成、系统参数的识别、需要在很长时间保存信息的长期记忆问题以及 Morris 水迷宫任务这样的具有挑战性的探索和记忆相结合的问题.

Balduzzi 等人[121]基于兼容离策略确定性 AC[83](Compatible Off-Policy Deterministic Actor Critic,COPDAC)算法提出值梯度反向传播(Value-Gradient Backpropagation,GProp)算法.该算法相对于 COPDAC 有两个创新点,一个是把值函数梯度的显式估计引入到算法中,二是提出 DAC(Deviator-Actor-Critic)模型,它包括三个神经网络,分别用来估计值函数、值函数的梯度以及决定 Actor 网络的策略.实验表明,GProp 在赌博机问题上和全监督方法旗鼓相当,而在章鱼臂任务[122]上实现了最好的效果.

3.10 阿尔法狗和阿尔法元

3.10.1 阿尔法狗

围棋因其庞大的搜索空间和评估棋盘位置、棋子移动的难度,一直被认为是最具挑战性的 AI 经典难题.但 Silver 等人[105]使围棋算法产生质的飞跃,提出阿尔法狗(AlphaGo)程序.

AlphaGo 将神经网络评估和蒙特卡洛树搜索[123](Monte Carlo Tree Search,MCTS)快速走子[124](Rollout)相结合,用值网络来评估棋盘的位置,用策略网络来选择棋子的移动,这些神经网络由人类专家的监督学习和自我对局的强化学习组合训练.AlphaGo 达到对其它围棋程序 99.8% 的获胜率,并且以 5 比 0 击败人类欧洲围棋冠军,达到最强人类棋手的水平,这一伟大成就也为其它难以实现人类水平的 AI 领域带来了希望.

3.10.2 阿尔法元

继 AlphaGo 之后,Silver 等人[125]又提出新一代围棋程序阿尔法元(AlphaGo Zero).AlphaGo Zero 和之前的 AlphaGo 版本相比主要有以下 4 个方面不同:

(1) AlphaGo Zero 通过自我对弈 RL 进行训练,没有使用任何的监督或人类数据.

(2) AlphaGo Zero 只使用棋盘上的黑子和白子作为输入特征,没有人工构建的特征.

(3) AlphaGo Zero 只使用一个神经网络来同时进行落子和评估,而不再是分开的值网络和策略网络.同时 AlphaGo Zero 使用的神经网络为残差网络,而 AlphaGo 使用的神经网络为卷积网络.

(4) AlphaGo Zero 使用依赖于单一神经网络的简化版树搜索进行位置的评估和落子的采样,而不再进行 Rollout 过程.

AlphaGo Zero 使用参数为 θ 的 DNN f_θ.该神经网络将原始棋盘的位置及其历史的表示 s 作为输入,输出落子位置的概率和一个值 $(p,v)=f_\theta(s)$.落子位置概率向量 $\boldsymbol{p}$ 表示选择每个落子位置的概率 $p_a=Pr(a|s)$,值 v 是个标量估值,用来估计当前棋手处在当前位置 s 时获胜的概率.这个神经网络将 AlphaGo 中策略网络和值网络扮演的角色结合到单一架构中.

AlphaGo Zero 中的神经网络使用 MCTS 选择每一步落子的自我对弈算法进行训练,在每个位置 s,MCTS α_θ 由神经网络 f_θ 指导执行搜索过程,α_θ 输出每一步落子的概率 $\pi=\alpha_\theta(s)$,这些概率通常要优于 f_θ 输出的原始落子概率 p,MCTS 因此可以被视为一个强力的策略优化算子.结合搜索的自我对弈就是用基于 MCTS 的改进策略选择每一次落子,然后将比赛获胜者 z 作为值的样本,这个过程可以被视为一个强力的策略估计算子.AlphaGo Zero 算法的主要思想就是在策略迭代过程中重复地使用这些搜索算子:神经网络的参数不断更新以使落子概率和值 $(p,v)=f_\theta(s)$ 逐渐匹配改进的搜索策略和自我对弈获胜者 (π,z).这些新的参数用于自我对弈的下一次迭代中以使搜索过程不断优化,上述过程具体实现如下:

首先,神经网络权重初始化为随机值 θ_0,在接下来的每一次迭代中 $(i\geq 1)$,自我对弈棋局不断生成.在每个时间点 t,使用上一次迭代更新的神经网络

$f_{\theta_{i-1}}$ 执行一次 MCTS 过程，同时对搜索概率进行采样选择一次落子 $a_t \sim \pi_t$，按照规则当比赛在时刻 T 结束时，将会计算出一个最终奖励 $r_T \in \{-1,+1\}$，每个时间点 t 的数据存为 (s_t,π_t,z_t)，其中 $z_t=\pm r_T$ 是时间点 t 的比赛获胜者（从当前选手角度来看）. 与此同时，对自我对弈上一次迭代过程中的所有时刻进行均匀采样得出数据 (s,π,z)，并根据此数据训练得出新的神经网络参数 θ_i. 神经网络 $(p,v)=f_\theta(s)$ 通过不断迭代来最小化预测值 v 和自我对弈获胜者 z 之间的误差，最大化落子概率 p 和搜索概率 π 之间的相似性. 具体地，对以下损失函数采用梯度下降算法来更新参数 θ：

$$L=(z-v)^2-\pi^T\log p+c\|\theta\|^2 \tag{63}$$

其中，c 是控制 L2 范数权正则化水平的参数，防止过拟合.

3.10.3　AlphaGo Zero 与基于值函数、策略梯度 DRL 方法的联系

基于值函数和策略梯度的 DRL 方法实际上都可看作是通过迭代不断优化策略的过程，这其中又涉及到策略估计和策略改善这两个过程. 策略估计就是估计当前策略的值函数，策略改善就是用值函数来得到更好的策略.

AlphaGo Zero 可以看作是将 MCTS 同时用于策略估计和策略改善的策略迭代过程. 策略改善始于神经网络策略，执行基于该策略的 MCTS 过程之后，将得到的改善后的搜索策略 π 反馈给神经网络. 策略估计被应用到搜索策略：自我对弈棋局的输出 z 也被反馈给神经网络，这些反馈过程通过上面提到的神经网络训练过程来实现.

3.11　小结与分析

本节首先介绍了策略梯度的概念，策略梯度包括 SPG 和 DPG 两种，常见的策略梯度算法包括 REINFORCE 和 AC 算法等. 然后阐述了随机性 AC 算法、离策略随机性 AC 算法和离策略确定性 AC 算法这 3 种 AC 算法的学习过程. 然后依次介绍了 3 种最主要的基于策略梯度的 DRL 方法：DDPG、TRPO 和 A3C 及基于这 3 种方法的一些改进方法. 接着介绍了两代围棋程序 AlphaGo 和 AlphaGo Zero，并分析了二者和本文两类 DRL 方法之间的联系. 最后简要介绍了基于策略梯度的一些其它方法. 表 2 总结了 DDPG、TRPO、A3C 及其各自改进方法解决的问题及意义和实验结果.

表 2　DDPG、TRPO、A3C 及其各自改进方法所解决的问题和优缺点及意义

方法名称	解决问题及意义	实验结果
深度确定性策略梯度(DDPG)	将 DQN 的思想应用到连续动作域中	能够鲁棒地解决超过 20 个物理仿真任务，性能能够与传统的规划算法相媲美
更新次数可配置的异步确定性策略梯度	解决机器人的灵活操纵任务	学到控制策略来鲁棒地抓取和堆叠物体
概率代理动作确定性策略梯度	连续控制离散的动作	离散控制任务 Acrobot 中媲美了 DQN 的表现
示范 DDPG	根据问题给出具有很好轨迹的奖励值函数，为智能体的每个状态提供一个平滑可变的奖励信号	在 4 个模拟机器人按指定位置插入物体的任务中，DDPGfD 比 DDPG 效果更好
信赖域策略优化(TRPO)	保证策略优化过程单调改进	对于优化像神经网络这样的大型非线性策略十分有效，在机器人的很多任务中都有着鲁棒的性能
使用克罗内克分解的行动者-评论家	从原始的像素输入中直接学习连续控制任务和离散控制策略	比优势行动者-评论家和 TRPO 获得更好的奖励和更高的抽样效率
使用广义优势估计的 TRPO	减少样本数量，在输入数据不平稳的情况下获得稳定的策略改进	在一些极具挑战性的 3D 运动任务上取得很好的效果
异步优势行动者-评论家(A3C)	创建多个智能体，在多个环境中异步、并行地执行和学习，去除更新序列的相关性	Atari 游戏中比以前基于 GPU 的算法用了更少的时间，比大规模分布式方法用了更少的存储和计算资源，取得更好的效果，同时也在一些连续运动问题上取得成功
GPU 版 A3C	A3C 的替代结构，强调对 GPU 的利用来增加每秒生成和处理的训练数据数量	比用 CPU 实现 A3C 收敛速度更快，以更短的时间实现当今最高的学习水平，也是当前 RL 领域针对各种游戏任务最好的方法.
并行优势行动者-评论家	提出一种灵活的并行框架，可用于在线策略、离策略、基于值和基于策略的算法	比 A3C 和 GA3C 需要更少的训练时间就能达到 Atari 游戏中最好的表现
无监督强化和辅助学习	充分利用学习环境中可用的训练信息以实现更快的训练和更鲁棒的学习	在 Atari 游戏上取得人类水平 8.8 倍的表现. 同时在 3D 迷宫任务中达到人类专家 87%的水平，在学习速度、参数设置的鲁棒性和最终的表现上也都超过 A3C
结合因式分解动作空间表示的 A3C	在离散的动作空间中分解策略和动作值函数	在 Atari 游戏 14 个任务中的 9 个表现都超过 A3C
结合基于置信度的自主学习回报的 A3C	动态地将权重分配给 n 步回报	在 Atari 游戏 22 个任务中的 18 个表现都超过 A3C
协作 A3C	在异构学习智能体之间进行自适应的知识转移	在 OpenAI gym 环境中取得很好的效果

4 实验对比与分析

4.1 常用实验平台介绍

4.1.1 ALE 学习环境

DRL 算法最常用的基准实验环境就是 ALE 学习环境[126]. ALE 建立在一个开源的 Atari 2600 模拟器 Stella 的基础上. ALE 通过接收操纵杆的动作、发送屏幕和/或 RAM 信息来让用户与 Atari 2600 交互. ALE 还提供一个游戏处理层,通过识别累积游戏得分和游戏是否结束,将每个游戏转换成一个标准的 RL 问题. 动作空间由操纵杆控制器定义的 18 个离散动作组成. 在实时运行时,模拟器每秒生成 60 帧,全速模拟时则可以达到每秒 6000 帧. 每个时间步骤的奖励是在游戏中按游戏规则来定义的,通常是通过在帧之间的得分差异来确定的. 在发出复位命令后,在第一个帧上开始一个时间段,并在游戏结束时或在预定义的帧数之后终止. 因此,用户可以通过一个公共接口访问几十个游戏.

ALE 还提供保存和恢复仿真器状态的功能. 当发出"保存"命令时,ALE 保存当前游戏的所有相关数据,包括 RAM、寄存器和地址计数器的内容. "恢复"命令同样将游戏重置为先前保存的状态. 这让 ALE 能够作为一个生成式模型来研究诸如像基于模型的 RL 这样的问题.

4.1.2 OpenAI Gym

OpenAI Gym[127] 是一个用于开发和比较 RL 算法的工具箱. OpenAI Gym 致力于将之前基准集(如 ALE 等)的优点以软件包的形式结合起来.

OpenAI Gym 包含一系列不同任务的集合(称为环境),并且这些任务共享一个公共接口,这意味着可以写一个通用的算法去完成不同的任务. 环境通过版本化的形式以确保软件更新之后,结果仍然是有意义且可复制的. OpenAI Gym 的环境包括 Atari 游戏、棋盘游戏、经典控制任务以及一些 2D 和 3D 机器人控制任务等.

除了软件库,OpenAI Gym 还有一个网站:gym.openai.com,在该网站可以找到所有环境的记分牌,展示用户提交的结果. 平台鼓励用户提供指向源代码的链接,以及如何复制其结果的详细说明.

4.1.3 RLLab

RLLab[128] 是针对连续控制问题,用于开发和评估 RL 算法的框架. RLLab 支持在 EC2 集群上运行实验,并提供用于可视化的工具,同时和 OpenAI Gym 完全兼容. RLLab 包括广泛的连续控制任务和一些算法的实现.

RLLab 上的任务可以分为 4 类:

(1) 基本任务. 包括经典的Cart-Pole和Acrobot等.

(2) 运动任务. 所有任务的目标是尽可能快地前进. 由于高度的自由,这些任务比基本任务更具挑战性.

(3) 部分可观测任务. 在被提供非完整状态信息(比如传感器的噪音和传感器视角的遮挡等)的环境下完成任务.

(4) 分层任务. 包括一些需要用到低级别电机控制和高级别决策的任务,比如"运动+食物搜集"和"运动+迷宫"任务等.

RLLab 实现的算法主要包括批处理算法和在线算法. 其中批处理算法包括 REINFORCE 和 TRPO 等,在线算法为 DDPG.

4.1.4 MuJoCo

MuJoCo[129] 是为基于模型的控制专门设计的物理 3D 模拟器,旨在促进机器人、生物力学、图形、动画以及其它需要快速精确模拟领域的研究和开发. MuJoCo 提供速度、精度和建模能力的独特组合,它是第一个以基于模型的优化为目的而设计的全功能模拟器,特别是通过接触进行优化. MuJoCo 可以扩展计算密集型技术,例如最优控制、物理一致性状态估计、系统识别和自动化机制设计,并将其应用到具有丰富交互行为的复杂动态系统中. MuJoCo 还拥有很多传统应用,例如在部署物理机器人之前控制方案的测试和验证、交互式科学可视化、虚拟环境、动画和游戏.

4.1.5 TORCS

TORCS 是一种现代化的、模块化的、高度便携的多玩家、多智能体赛车模拟器. TORCS 具有高度的模块化和可移植性,使其成为 AI 研究的理想对象.

TORCS 可用于开发各种问题的 AI 智能体. 在汽车层面,可以开发新的模拟模块,其中包括各种汽车零部件的智能控制系统. 在驾驶员层面,低级别 API 提供对模拟状态的详细(但只是部分)访问. 这可以用来开发任何东西,从中级控制系统到复杂的驾驶智能体. 该智能体可以找到最优的赛车路线,对意外做出反应,并可规划比赛策略. TORCS 也为那些喜欢挑战并且对视觉处理感兴趣的研究者提供 3D 投影界面.

4.2 两类 DRL 方法对比

在 ALE 环境下,通过实验评估基于值函数和

基于策略梯度这两类 DRL 方法在 57 个 Atari 2600 游戏中的性能. 其中基于值函数的 DRL 方法包括 DQN、DRQN、DDQN 和竞争 DDQN，基于策略梯度的 DRL 方法包括 TRPO、采用 DDQN 前馈网络结构的 A3C(表 3 中表示为 A3C，FF)、采用 LSTM 循环网络结构的 A3C(表 3 中表示为 A3C，LSTM)和 UNREAL，总计 8 种 DRL 方法. 选取这 8 种方法的原因是：这 8 种方法是非常常见和典型的 DRL 方法，许多 DRL 方法都是由这 8 种方法衍生而来，因此具有代表性.

表 3　8 种 DRL 方法在 57 个 Atari 游戏中的训练时间、得分均值(以人类表现的百分比衡量)和得分中值(以人类表现的百分比衡量)

方法	训练时间	得分均值/%	得分中值/%
DQN	GPU 上训练 8 天	118.6	49.5
DRQN	GPU 上训练 8 天	180.7	65.8
DDQN	GPU 上训练 8 天	324.9	109.3
竞争 DDQN	GPU 上训练 8 天	345.4	120.3
TRPO	GPU 上训练 8 天	150.2	54.4
A3C，FF	CPU 上训练 1 天	347.0	72.2
A3C，LSTM	CPU 上训练 4 天	619.8	108.6
UNREAL	CPU 上训练 4 天	862.5	233.2

实验过程中，先对 Beamrider、Breakout、Pong、Q*bert、Seaquest 和 Space Invaders 这 6 个游戏进行参数调试，然后将调试好的参数应用到所有 57 个游戏中并固定不变，其中所有游戏所用的优化算法均为 RMSProp. 对于 2 种 A3C 方法和 UNREAL，使用 16 核 CPU 进行训练，其它智能体则使用 Nvidia K40 GPU 进行训练. 表 3 给出这 8 种 DRL 算法在所有 57 个游戏中的训练时间、得分均值(以人类表现的百分比来衡量)和得分中值(以人类表现的百分比来衡量). 其中得分均值和得分中值可以大体上反映某种方法在 57 个游戏中的整体性能和表现，是许多针对 Atari 2600 游戏的 DRL 实验研究采用的做法. 57 个游戏中统一所用参数及其对应的值如表 4 所示.

表 4　57 个 Atari 游戏中统一所用参数和值

参数名称	参数值
Minibatch 大小	32
回放记忆序列大小	1 000 000
折扣率	0.99
RMSProp 学习率	0.0002
RMSProp 梯度动量	0.95
RMSProp 平方梯度动量	0.95
RMSProp 最小平方梯度	0.01
ε-贪婪算法 ε 初值	1
ε-贪婪算法 ε 终值	0.1

实验结果分析：从上面的实验结果可以看出，实验条件相同(都在 GPU 上训练 8 天)的情况下，4 种基于值函数的 DRL 方法中，DQN、DRQN、DDQN 和竞争 DDQN 在 57 个游戏中的表现依次变好，得分均值和得分中值都依次变高. 其中 DRQN 相比 DQN 效果提升并不明显，这是因为前者针对的是状态部分可观测的情况，而此实验中游戏画面是完整的. DDQN 通过添加目标网络解决 Q 学习中过高估计值函数的问题，因此效果较 DQN 有大幅度提高，而竞争 DDQN 则在 DDQN 的基础上使用竞争网络结构以及优先经验回放技术，进一步提升效果，取得 4 种方法中的最佳表现.

4 种基于策略梯度的 DRL 方法中，TRPO 相比 DQN 效果并没有明显提高，实验效果也低于其它基于策略梯度的 DRL 方法，这是因为 TRPO 并不是为 Atari 游戏特别设计的，它是能应用到其它任务的泛化能力很强的算法. 同时，采用 LSTM 循环网络结构的 A3C 在得分均值和得分中值上都远高于 DDQN 前馈网络结构的 A3C，可见 LSTM 网络整合游戏画面帧间信息的能力更高. UNREAL 在 4 种基于策略梯度的 DRL 算法中，得分均值和得分中值都明显高于其它 3 种方法，足见在 A3C 中引入辅助任务十分有效. 除此之外可以注意到，采用 DDQN 前馈网络结构的 A3C 出现得分均值相对较高但得分中值相对较低的情况，这可能是因为它在游戏本身设定分数较高的那些游戏中效果较好，但在分数较低的那些游戏中效果较差.

A3C 和 UNREAL 在只使用 16 核 CPU 而没有使用 GPU 的情况下，相比其它 DRL 方法仍然极大地提高 57 个 Atari 2600 游戏的得分均值，并大幅度缩短所需要的训练时间. 经过 1 天的训练后，采用 DDQN 前馈网络结构的 A3C 就媲美竞争 DDQN 训练 8 天的得分均值. 经过 4 天的训练后，采用 LSTM 循环网络结构的 A3C 和 UNREAL 就远高于基于值函数的 4 种 DRL 方法训练 8 天的得分均值. 可见采用“异步并行”思想的 A3C 和 UNREAL 这 2 种基于策略梯度的 DRL 方法，无论在速度上还是在性能上都媲美或者超过 4 种基于值函数的 DRL 方法.

4.3　Q 值逼近

Gridworld[130] 是常见的 RL 问题. 和 ALE 相反，Gridworld 有更小的状态空间，可以让回放记忆序列包含所有的状态-动作对. 除此之外，Gridworld 环境下可以精确计算出最优值函数 Q^*. 因此对于 DRL 方法可以在 Gridworld 环境下研究 DNN 逼近 Q 值的问题.

Anschel 等人[49] 在 Gridworld 环境下通过实验

研究平均 DQN 中 K 值大小对学习到的策略好坏的影响，以及 DQN 和平均 DQN 分别逼近的预测 Q 值收敛到真值的情况，其中在每次目标网络参数的更新中对 32 个样本的 300 个 minibatch 使用 ADAM 优化器[131]进行训练.

如图 15 所示，加粗虚线表示真值，4 条变化曲线分别表示 DQN(此时 $K=1$)、$K=5$ 的平均 DQN、$K=10$ 的平均 DQN 和 $K=20$ 的平均 DQN 这 4 种方法的平均预测 Q 值随训练迭代次数的变化情况，4 条曲线的阴影部分表示各自的标准差，A、B、C、D 分别表示上面 4 种方法最终收敛时的平均预测 Q 值和真值之差.

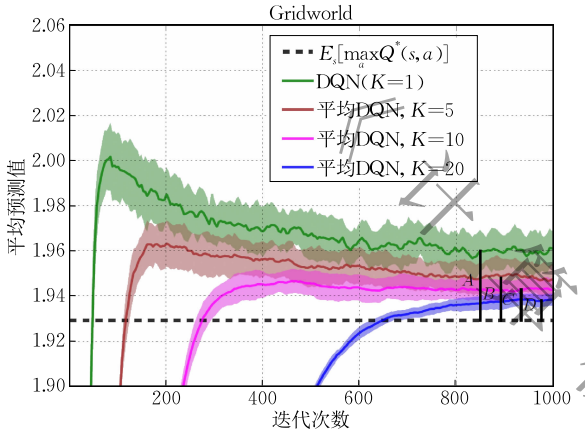

图 15 Gridworld 环境下 DQN 和平均 DQN 的平均预测值

实验结果分析：从上面的实验结果可以看出，虽然最终 4 种方法都能较好地收敛到真值，但随着 K 值的逐渐增大，最终收敛的平均预测 Q 值和真值之差逐渐减小，即增大 K 值可以减少过高估计. 增大 K 值也可以减少 Q 值的超调量，并减小标准差，有着更平滑的收敛. 同时 K 取值不同的 3 种平均 DQN 方法都比 DQN 方法在效果上有明显的提升：逼近误差更小，训练过程更稳定，足见平均 DQN 将之前的 Q 值估计取平均的做法的有效性.

5 深度强化学习的未来研究方向

虽然 DRL 发展迅速，但是要实际应用 DRL，仍旧面临着很多的挑战. 基于这些挑战和待解决的问题，我们尝试提出 DRL 的未来研究方向如下：

(1) DRL 需要大量的样本而不是使用相对更加简单的模型来调整 DNN 包含的大量参数，包括 DNN 连接边权重矩阵和偏置. 因此目前 DRL 领域中大部分的研究集中于无模型(model-free)方法. 而样本获取的过程则是通过和环境的大量交互实现的，但交互次数在实际中可能是有限的. 比如在合理的时间内，在没有重大硬件损耗的情况下，用机器人进行数百万次的实验显然是不现实的. 因此在不直接与环境交互的情况下对环境建模，将会有效减少和环境交互的次数，这就要使用基于模型(model-based)的方法. 基于模型的学习可以提高采样效率，原则上，一个好的模型可以解决一系列的问题. 就像我们在 AlphaGo 中看到的一样，拥有一个比较全面的模型使得学习解决方案变得更加容易. 在提高 DNN 的数据效率方面，基于模型的 DRL 研究未来将会受到广泛的关注.

(2) DRL 最著名的基准测试环境就是 Atari 2600 游戏，它以每秒 60 帧的速度运行. 但即便是一些性能很好的 DRL 方法也需要几千万帧数据训练之后才能达到人类水平，这显然比实践水平的采样效率低几个数量级. 解决这个问题的方法之一就是开发出更好的硬件，运行速度越快，就可以越有效率地解决问题. 因此，扩展硬件成为 DRL 发展的关键方向.

(3) DRL 的学习过程中需要存在一个奖励函数，创建一个奖励函数并不困难，但困难在于设计可学习的奖励函数去激励智能体得到期望的行为. 为了达到目的，必须精准地设计奖励函数，否则奖励函数会过拟合地学习，从而导致不希望出现的结果. Atari 游戏之所以是出色基准的原因，是因为在 Atari 游戏中，不仅能够轻易地得到大量的样本，而且每款游戏都有着同样一个目标，那就是将得分最大化，所以每款游戏都有一样的奖励函数. 但在很多环境中，设计出一个适合的奖励函数并不容易. 不过利用模仿学习[132](Imitation Learning)和逆强化学习[133](Inverse Reinforcement Learning)已经可以学习到更好的奖励函数，其中奖励函数可以通过人类的演绎或者评估来隐式地定义. 因此模仿学习和逆强化学习与 DRL 的结合将会是该领域在未来的重点研究方向.

(4) DRL 应用到实际问题的难点之一就是在很多情况下环境中的奖励是稀疏的，此时获得的帮助信息很少，需要增加更多的学习信息，目前处理该问题的做法有：引入示范数据，如 Večerík 等人[88]将示范转移数据和智能体数据之间的采样比率通过优先重放机制自动调整；引入辅助任务，如 Jaderberg 等人[103]引入辅助任务：奖励预测，以使用历史连续多帧的图像输入来预测下一步的回馈值作为训练目标，让神经网络具有更好的表达能力；采用分层 DRL 的结构，如 Chen 等人[134]使用顶层策略来选择

子策略，状态空间和子目标的划分通过手动方式实现. 但稀疏奖励的问题仍将是 DRL 未来的重点研究方向之一.

(5) 尽管 DRL 可以处理许多高维输入问题，但这需要大量的样本. 为了加快 DRL 的学习速度，我们需要利用之前从相关任务中获得的知识，此时就需要将迁移学习(transfer learning)和 DRL 相结合[135]. 迁移学习是将不同域中的知识进行迁移的过程，这些域可能有着不同的特征空间或不同的数据概率分布. 迁移学习可以减少训练样本和环境交互次数，帮助 DRL 进行多任务训练和构建更有效的策略. 借助迁移学习让 DRL 来鲁棒地解决多个不同的任务，将是未来的重点研究方向.

(6) 随着近几年 DRL 的不断发展，DNN 的一个缺点也日益明显：训练过程需要消耗大量人类标注样本，而这对于应用到如医疗图像处理这样的小样本领域是很难办到的. 人工智能的重要目标之一就是在没有任何人类先验知识的前提下，通过完全的自学在极具挑战性的应用领域达到非凡的学习效果. 2017 年最新的围棋程序 AlphaGo Zero 实现对战 AlphaGo 的 100∶0 完胜，它摆脱对人类先验知识的依赖，只靠人类给定的围棋规则，就实现“无师自通”的当今最先进的围棋竞技水平. 如何将类似的 DRL 算法更广泛地应用到其它人类缺乏了解或是缺乏大量标注数据的领域，比如解决诸如蛋白质折叠和新材料开发这样的重要问题，将是 DRL 未来重点研究的方向之一. 同时我们在未来将要面对的一个挑战可能是在一些与日常生活有关的决策问题上，人类经验和机器经验同时存在，而两者之间又存在着很大的差别时，我们又该如何去选择和利用?

(7) 研究离散动态模型和连续动态模型混合的多智能体 DRL 混杂模型. 多智能体混杂模型的混合过程由事件驱动，在不同的事件场景中，切换到适当的单个或多个模型中去.

6　结论与展望

鉴于 DRL 的理论意义和实际应用价值，本文对基于值函数和策略梯度的 DRL 进行了系统的综述. 本文首先介绍了 DQN 和基于 DQN 的各种改进方法，它们是基于值函数的 DRL 的最主要方法. 其中基于 DQN 的各种改进方法可主要分为训练算法的改进、神经网络结构的改进、引入新的学习机制的改进和基于新提出的 RL 算法的改进这 4 大类. 训练算法的改进方法包括双 DQN、示范深度 Q 学习、对称 DQN、指导 DQN、最小二乘 DQN、反向传播贝叶斯 Q 网络、引入内部惩罚信号的 DQN、深度乐观线性支持学习和平均 DQN；神经网络结构的改进方法包括竞争网络结构、深度循环 Q 网络、基于文本的 LSTM-DQN、特定动作深度循环 Q 网络、自举 DQN、增广对数先验深度 Q 学习和 Gorila DQN；引入新的学习机制的改进方法包括优先经验回放、注意力机制深度循环 Q 网络和引入内部恐惧机制的 DQN；基于新提出的 RL 算法的改进方法包括基于归一化优势函数的 DQN 和深度多 Q 学习.

然后详细阐述了 DDPG、TRPO 和 A3C 这三种最重要的基于策略梯度的 DRL 方法，并概述了基于这三种方法的一些改进版本. 其中 DDPG 的改进方法包括更新次数可配置的异步 DPG、概率代理动作 DPG 和示范 DDPG；TRPO 的改进方法包括使用克罗内克分解的行动者-评论家和广义优势估计的 TRPO；A3C 的改进方法包括 GPU 版 A3C、并行优势行动者-评论家、无监督强化和辅助学习、结合因式分解动作空间表示的 A3C、结合基于置信度的自主学习回报的 A3C 和协作 A3C.

然后对两代围棋程序阿尔法狗、阿尔法元进行介绍，并对二者与本文阐述的两类 DRL 方法之间的联系进行分析. 最后提出了 DRL 当前存在的问题和挑战，并尝试对 DRL 未来的研究方向进行展望. 虽然 DRL 领域仍然存在很多挑战，但随着研究不断深入和理论不断发展，DRL 将会成为构建通用人工智能系统的重要组成部分.

参　考　文　献

[1] Munos R. From bandits to Monte-Carlo tree search: The optimistic principle applied to optimization and planning. Foundations and Trends in Machine Learning, 2014, 7(1): 1-129

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

[3] Bertsekas D P, Bertsekas D P, Bertsekas D P, et al. Dynamic Programming and Optimal Control. Belmont, USA: Athena Scientific, 1995

[4] Szepesvari C. Algorithms for reinforcement learning. Synthesis Lectures on Artificial Intelligence and Machine Learning, 2010, 4(1): 1-103

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

[6] Sermanet P, Kavukcuoglu K, Chintala S, et al. Pedestrian detection with unsupervised multi-stage feature learning//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Portland, USA, 2013: 3626-3633

[7] Dahl G E, Acero A. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. IEEE Transactions on Audio Speech & Language Processing, 2011, 20(1): 30-42

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

[9] Huval B, Coates A, Ng A. Deep learning for class-generic object detection. arXiv preprint arXiv:1312.6885, 2013

[10] Makantasis K, Karantzalos K, Doulamis A, et al. Deep learning-based man-made object detection from hyperspectral data//Proceedings of the Advances in Visual Computing-11th International Symposium. Las Vegas, USA, 2015: 717-727

[11] Jain V, Seung H S. Natural image denoising with convolutional networks//Proceedings of the 22nd Annual Conference on Neural Information Processing Systems. Vancouver, Canada, 2008: 769-776

[12] Mikolov T, Karafiát M, Burget L, et al. Recurrent neural network based language model//Proceedings of the Conference of International Speech Communication Association. Chiba, Japan, 2010: 1045-1048

[13] He K M, Zhang X Y, Ren S. Deep residual learning for image recognition//Proceedings of the IEEE Conference on Computer and Pattern Recognition. Las Vegas, USA, 2016: 770-778

[14] Goodfellow I J, Pouget A, Mirza M, et al. Generative adversarial nets//Proceedings of the Neural Information and Processing System. Montreal, Canada, 2014: 2672-2680

[15] Mnih V, Kavukcuoglu K, Silver D, et al. Playing Atari with deep reinforcement learning. //Proceedings of the Workshops at the 26th Neural Information Processing Systems 2013. Lake Tahoe, USA, 2013: 201-220

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

[17] Zhang M, Geng X, Bruce J, et al. Deep reinforcement learning for tensegrity robot locomotion//Proceedings of the International Conference on Robotics and Automation. Singapore, 2017: 634-641

[18] Gu S, Holly E, Lillicrap T, et al. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates //Proceedings of the IEEE International Conference on Robotics and Automation. Singapore, 2017: 3389-3396

[19] Cuayáhuitl H, Yu S, Williamson A, et al. Scaling up deep reinforcement learning for multi-domain dialogue systems//Proceedings of the International Joint Conference on Neural Networks. Anchorage, USA, 2017: 3339-3346

[20] Zhao T, Eskenazi M. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning//Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue. Los Angeles, USA, 2016: 1-10

[21] Gao J, Shen Y, Liu J, et al. Adaptive traffic signal control: Deep reinforcement learning algorithm with experience replay and target network. arXiv preprint arXiv:1705.02755, 2017

[22] Genders W, Razavi S. Using a deep reinforcement learning agent for traffic signal control. arXiv preprint arXiv:1611.01142, 2016

[23] Xiong X, Wang J, Zhang F, et al. Combining deep reinforcement learning and safety based control for autonomous driving. arXiv preprint arXiv:1612.00147, 2016

[24] Sallab A E L, Abdou M, Perot E, et al. Deep reinforcement learning framework for autonomous driving. Electronic Imaging, 2017, 2017(19): 70-76

[25] Thananjeyan B, Garg A, Krishnan S, et al. Multilateral surgical pattern cutting in 2D orthotropic gauze with deep reinforcement learning policies for tensioning//Proceedings of the IEEE International Conference on Robotics and Automation. Singapore, 2017: 2371-2378

[26] O'Shea T J, Clancy T C. Deep reinforcement learning radio control and signal detection with KeRLym, a gym RL agent. arXiv preprint arXiv:1605.09221, 2016

[27] Liu Quan, Zhai Jian-Wei, Zhang Zong-Zhang, et al. A survey on deep reinforcement learning. Chinese Journal of Computers, 2018, 40(1): 1-27(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, 40(1): 1-27)

[28] Zhao Dong-Bin, Shao Kun, Zhu Yuan-Heng, et al. Review of deep reinforcement learning and discussions on the development of computer Go. Control Theory & Applications, 2016, 33(6): 701-717(in Chinese)
(赵冬斌, 邵坤, 朱圆恒等. 深度强化学习综述: 兼论计算机围棋的发展. 控制理论与应用, 2016, 33(6): 701-717)

[29] Sutton R S. Learning to predict by the methods of temporal differences. Machine Learning, 1988, 3: 9-44

[30] Cjch W, Dayan P. Q-learning. Machine Learning, 1992, 8(3-4): 279-292

[31] Lin L J. Self-improving reactive agents based on reinforcement learning, planning and teaching. Machine Learning, 1992, 8(3-4): 293-321

[32] Hasselt H V, Guez A, Silver D. Deep reinforcement learning with double Q-learning//Proceedings of the 13th AAAI Conference on Artificial Intelligence. Phoenix, USA, 2016: 2094-2100

[33] Hasselt H V. Double Q-learning//Proceedings of the 23: 24th Annual Conference on Neural Information Processing Systems. Vancouver, Canada, 2010: 2613-2621

[34] Hester T, Vecerik M, Pietquin O, et al. Learning from demonstrations for real world reinforcement learning. arXiv preprint arXiv:1704.03732, 2017

[35] Schaul T, Quan J, Antonoglou I, Silver D, et al. Prioritized experience replay//Proceedings of the 4th International Conference on Learning Representations. San Juan, Puerto Rico, 2016: 322-355

[36] Wang Z, Schaul T, Hessel M, et al. Dueling network architectures for deep reinforcement learning//Proceedings of the 33rd International Conference on Machine Learning. New York, USA, 2016: 1995-2003

[37] Mahajan A, Tulabandhula T. Symmetry learning for function approximation in reinforcement learning. arXiv preprint arXiv: 1706.02999, 2017

[38] Narayanamurthy S M, Ravindran B. Efficiently exploiting symmetries in real time dynamic programming//Proceedings of the 20th International Joint Conference on Artificial Intelligence. Hyderabad, India, 2007: 2556-2561

[39] Taitler A, Shimkin N. Learning control for air hockey striking using deep reinforcement learning. arXiv preprint arXiv: 1702.08074, 2017

[40] Levine N, Zahavy T, Mankowitz D J, et al. Shallow updates for deep reinforcement learning//Proceedings of the Annual Conference on Neural Information Processing Systems. Long Beach, USA, 2017: 3138-3148

[41] Hill B. Bayesian inference in statistical analysis. Technometrics, 1973, 16(3): 478-479

[42] Lipton Z C, Li X, Gao J, et al. Efficient dialogue policy learning with BBQ-networks. arXiv preprint arXiv: 1608.05081, 2016

[43] Blundell C, Cornebise J, Kavukcuoglu K, et al. Weight uncertainty in neural networks. arXiv preprint arXiv: 1505.05424, 2015

[44] Leibfried F, Graumoya J, Bouammar H. An information-theoretic optimality principle for deep reinforcement learning. arXiv preprint arXiv: 1708.01867, 2017

[45] Ortega P A, Braun D A. Thermodynamics as a theory of decision-making with information processing costs. Royal Society A Mathematical Physical & Engineering Sciences, 2013, 469(2153): 926-930

[46] Mossalam H, Assael Y M, Roijers D M, et al. Multi-objective deep reinforcement learning. arXiv preprint arXiv: 1610.02707, 2016

[47] Roijers D M, Whiteson S, Oliehoek F A. Algorithmic Decision Theory: Computing Convex Coverage Sets for Faster Multi-Objective Coordination. Berlin and Heidelberg: Springer-Verlag, 2015

[48] Roijers D M, Vamplew P, Whiteson S, et al. A survey of multi-objective sequential decision-making. Journal of Artificial Intelligence Research, 2013, 48: 67-113

[49] Anschel O, Baram N, Shimkin N. Averaged-DQN: Variance reduction and stabilization for deep reinforcement learning//Proceedings of the 34th International Conference on Machine Learning. Sydney, Australia, 2017: 176-185

[50] Raghu A, Komorowski M, Celi L A, et al. Continuous state-space models for optimal sepsis treatment—A deep reinforcement learning approach//Proceedings of the Machine Learning for Health Care. Boston, USA, 2017: 147-163

[51] Hausknecht M, Stone P. Deep recurrent Q-learning for partially observable MDPs. arXiv preprint arXiv: 1507.06527, 2015

[52] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation, 1997, 9(8): 1735-1780

[53] Narasimhan K, Kulkarni T, Barzilay R. Language understanding for text-based games using deep reinforcement learning //Proceedings of the Conference on Empirical Methods in Natural Language Processing. Lisbon, Portugal, 2015: 1-11

[54] Zhu P, Li X, Poupart P. On improving deep reinforcement learning for POMDPs. arXiv preprint arXiv: 1704.07978, 2017

[55] Osband I, Roy B V, Wen Z. Generalization and exploration via randomized value functions//Proceedings of the 33rd International Conference on Machine Learning. New York, USA, 2016: 2377-2386

[56] Osband I, Blundell C, Pritzel A, et al. Deep exploration via bootstrapped DQN//Proceedings of the Annual Conference on Neural Information Processing Systems. Barcelona, Spain, 2016: 4026-4034

[57] Bickel P J, Freedman D A. Some asymptotic theory for the bootstrap//Proceedings of the 30th AAAI Conference on Artificial Intelligence. Phoenix, USA, 2016: 2094-2100

[58] Jaques N, Gu S, Bahdanau D, et al. Sequence tutor: Conservative fine-tuning of sequence generation models with KL-control//Proceedings of the 34th International Conference on Machine Learning. Sydney, Australia, 2017: 1645-1654

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

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

[61] Lipton Z C, Kumar A, Gao J, et al. Combating deep reinforcement learning's sisyphean curse with reinforcement learning. arXiv preprint arXiv: 1611.01211, 2017

[62] Gu Shixiang, Lillicrap T, Sutskever I, et al. Continuous deep Q-learning with model-based acceleration//Proceedings of the 33rd International Conference on Machine Learning. New York, USA, 2016: 2829-2838

[63] Duryea E, Ganger M, Hu W. Exploring deep reinforcement learning with multi Q-Learning. Intelligent Control & Automation, 2016, 07(4): 129-144

[64] 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

[65] 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

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

[67] Hessel M, Modayil J, Van Hasselt H, et al. Rainbow: Combining improvements in deep reinforcement learning. arXiv preprint arXiv:1710.02298, 2017

[68] Lee K, Choi S, Oh S. Sparse Markov decision processes with causal sparse Tsallis entropy regularization for reinforcement learning. arXiv preprint arXiv:1709.06293, 2017

[69] He H, Boydgraber J, Kwok K, et al. Opponent modeling in deep reinforcement learning//Proceedings of the 33rd International Conference on Machine Learning. New York, USA, 2016: 1804-1813

[70] Palmer G, Tuyls K, Bloembergen D, et al. Lenient multi-agent deep reinforcement learning. arXiv preprint arXiv: 1707.04402, 2017

[71] Potter M A, De Jong K A. A cooperative coevolutionary approach to function optimization. Computer Science, 1994, 866: 249-257

[72] Omidshafiei S, Pazis J, Amato C, et al. Deep decentralized multi-task multi-agent reinforcement learning under partial observability//Proceedings of the 34th International Conference on Machine Learning. Sydney, Australia, 2017: 2681-2690

[73] Qureshi A H, Nakamura Y, Yoshikawa Y, et al. Robot gains social intelligence through multimodal deep reinforcement learning//Proceedings of the 16th IEEE International Conference on Humanoid Robots. Cancun, Mexico, 2016: 745-751

[74] Bellemare M G, Ostrovski G, Guez A, et al. Increasing the action gap: New operators for reinforcement learning// Proceedings of the 13rd AAAI Conference on Artificial Intelligence. Phoenix, USA, 2016: 1476-1483

[75] He F S, Liu Y, Schwing A G, et al. Learning to play in a day: Faster deep reinforcement learning by optimality tightening. arXiv preprint arXiv:1611.01606, 2016

[76] Lillicrap T P, Hunt J J, Pritzel A, et al. Continuous control with deep reinforcement learning. Computer Science, 2015, 8(6): A187

[77] Schulman J, Levine S, Moritz P, et al. Trust region policy optimization//Proceedings of the International Conference on Machine Learning. Lugano, Switzerland, 2015: 1889-1897

[78] Williams R J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning, 1992, 8(3-4): 229-256

[79] Peters J, Vijayakumar S, Schaal S. Natural actor-critic// Proceedings of the 16th European Conference on Machine Learning. Porto, Portugal, 2005: 280-291

[80] Bhatnagar S, Sutton R S, Ghavamzadeh M, et al. Incremental natural actor-critic algorithms//Proceedings of the 21st Annual Conference on Neural Information Processing Systems. Vancouver, Canada, 2007: 105-112

[81] Degris T, Pilarski P M, Sutton R S. Model-free reinforcement learning with continuous action in practice//Proceedings of the American Control Conference. Montreal, Canada, 2012: 2177-2182

[82] Sutton R S. Policy gradient methods for reinforcement learning with function approximation//Proceedings of the Advances in Neural Information Processing Systems. Denver, USA, 1999: 1057-1063

[83] Silver D, Lever G, Heess N, et al. Deterministic policy gradient algorithms//Proceedings of the 31st International Conference on Machine Learning. Beijing, China, 2014: 387-395

[84] Degris T, White M, Sutton R S. Linear off-policy actor-critic //Proceedings of the 29th International Conference on Machine Learning. Edinburgh, UK, 2012

[85] Sutton R S, Maei H R, Precup D, et al. Fast gradient-descent methods for temporal-difference learning with linear function approximation//Proceedings of the 26th Annual International Conference on Machine Learning. Montreal, Canada, 2009: 993-1000

[86] Popov I, Heess N, Lillicrap T, et al. Data-efficient deep reinforcement learning for dexterous manipulation. arXiv preprint arXiv:1704.03073, 2017

[87] Wang S, Jing Y. Deep reinforcement learning with surrogate agent-environment interface. arXiv preprint arXiv: 1709.03942, 2017

[88] Večerík M, Hester T, Scholz J, et al. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. arXiv preprint arXiv:1707.08817, 2017

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

[90] Bruin T D, Kober J, Tuyls K, et al. Improved deep reinforcement learning for robotics through distribution-based experience retention//Proceedings of the International Conference on Intelligent Robots and Systems. Daejeon, South Korea, 2016: 3947-3952

[91] Kakade S, Langford J. Approximately optimal approximate reinforcement learning//Proceedings of the 19th International Conference on Machine Learning. Sydney, Australia, 2002: 267-274

[92] Baxter J, Bartlett P L. Infinite-horizon policy-gradient estimation. Journal of Artificial Intelligence Research, 2011, 15(1): 319-350

[93] Lagoudakis M G, Parr R. Reinforcement learning as classification: Leveraging modern classifiers//Proceedings of the 20th International Conference on Machine Learning. Washington, USA, 2003: 424-431

[94] Gabillon V, Ghavamzadeh M, Scherrer B. Approximate dynamic programming finally performs well in the game of Tetris//Proceedings of the Advances in Neural Information Processing Systems. Lake Tahoe, USA, 2013: 1754-1762

[95] Kakade S. A natural policy gradient//Proceedings of the Advances in Neural Information Processing Systems. Vancouver, Canada, 2001: 1531-1538

[96] Wu Y, Mansimov E, Liao S, et al. Scalable trust-region method for deep reinforcement learning using Kronecker-factored approximation. arXiv preprint arXiv:1708.05144, 2017

[97] Grosse R, Martens J. A Kronecker-factored approximate fisher matrix for convolution layers//Proceedings of the International Conference on Machine Learning. New York, USA, 2016: 573-582

[98] Martens J, Grosse R. Optimizing neural networks with Kronecker-factored approximate curvature//Proceedings of the 32nd International Conference on Machine Learning. Lille, France, 2015: 2408-2417

[99] Schulman J, Moritz P, Levine S, et al. High-dimensional continuous control using generalized advantage estimation. arXiv preprint arXiv:1506.02438, 2015

[100] Feng N, Recht B, Re C, et al. HOGWILD!: A lock-free approach to parallelizing stochastic gradient descent// Proceedings of the Advances in Neural Information Processing Systems. Granada, Spain, 2011: 693-701

[101] Babaeizadeh M, Frosio I, Tyree S, et al. Reinforcement learning through asynchronous advantage actor-critic on a GPU. arXiv preprint arXiv:1611.06256, 2017

[102] Clemente A V, Castejón H N, Chandra A. Efficient parallel methods for deep reinforcement learning. arXiv preprint arXiv: 1705.04862, 2017

[103] Jaderberg M, Mnih V, Czarnecki W M, et al. Reinforcement learning with unsupervised auxiliary tasks. arXiv preprint arXiv:1611.05397, 2016

[104] Zahavy T, Zrihem N B, Mannor S. Graying the black box: Understanding DQNs//Proceedings of the 33rd International Conference on Machine Learning. New York, USA, 2016: 1899-1908

[105] Silver D, Huang A, Maddison C J, et al. Mastering the game of Go with deep neural networks and tree search. Nature, 2016, 529(7587): 484-489

[106] Li X, Li L, Gao J, et al. Recurrent reinforcement learning: A hybrid approach. arXiv preprint arXiv:1509.03044, 2015

[107] Mirowski P, Pascanu R, Viola F, et al. Learning to navigate in complex environments. arXiv preprint arXiv:1611.03673, 2016

[108] Sharma S, Suresh A, Ramesh R, et al. Learning to factor policies and action-value functions: Factored action space representations for deep reinforcement learning. arXiv preprint arXiv:1705.07269, 2017

[109] Sharma S, Raguvir J G, Ramesh S, et al. Learning to mix n-Step returns: Generalizing lambda-returns for deep reinforcement learning. arXiv preprint arXiv:1705.07445, 2017

[110] Wang Z, Bapst V, Heess N, et al. Sample efficient actor-critic with experience replay. arXiv preprint arXiv:1611.01224, 2016

[111] Munos R, Stepleton T, Harutyunyan A, et al. Safe and efficient off-policy reinforcement learning//Proceedings of the Advances in Neural Information Processing Systems. Barcelona, Spain, 2016: 1046-1054

[112] Wawrzyński P. Real-time reinforcement learning by sequential Actor-Critics and experience replay. Neural Networks, 2009, 22(10): 1484-1497

[113] Nachum O, Norouzi M, Xu K, et al. Bridging the gap between value and policy based reinforcement learning// Proceedings of the Advances in Neural Information Processing Systems. Long Beach, USA, 2017: 2772-2782

[114] O'Donoghue B, Munos R, Kavukcuoglu K, et al. Combining policy gradient and Q-learning. arXiv preprint arXiv:1611.01626, 2016

[115] Schulman J, Chen X, Abbeel P. Equivalence between policy gradients and soft Q-Learning. arXiv preprint arXiv:1704.06440, 2017

[116] Lin K, Wang S, Zhou J. Collaborative deep reinforcement learning. arXiv preprint arXiv:1702.05796, 2017

[117] Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. Computer Science, 2015, 14(7): 38-39

[118] Tangkaratt V, Abdolmaleki A, Sugiyama M. Deep reinforcement learning with relative entropy stochastic search. arXiv preprint arXiv:1705.07606, 2017

[119] Heess N, Hunt J J, Lillicrap T P, et al. Memory-based control with recurrent neural networks. arXiv preprint arXiv: 1512.04455, 2015

[120] Heess N, Wayne G, Silver D, et al. Learning continuous control policies by stochastic value gradients//Proceedings of the Advances in Neural Information Processing Systems. Montreal, Canada, 2015: 2944-2952

[121] Balduzzi D, Ghifary M. Compatible value gradients for reinforcement learning of continuous deep policies. Computer Science, 2015, 8(6): A187

[122] Engel Y, Szabo P, Volkinshtein D. Learning to control an octopus arm with Gaussian process temporal difference methods//Proceedings of the Advances in Neural Information Processing Systems. Vancouver, Canada, 2005: 347-354

[123] Coulom R. Efficient selectivity and backup operators in Monte-Carlo tree search. Lecture Notes in Computer Science, 2006, 4630: 72-83

[124] Tesauro G, Galperin G R. On-line policy improvement using Monte-Carlo search//Proceedings of the Advances in Neural Information Processing Systems. Denver, USA, 1996: 1068-1074

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

[126] Bellemare M G, Naddaf Y, Veness J, et al. The arcade learning environment: An evaluation platform for general agents (extended abstract)//Proceedings of the 24th International Joint Conference on Artificial Intelligence. Buenos Aires, Argentina, 2015: 4148-4152

[127] Brockman G, Cheung V, Pettersson L, et al. Openai gym. arXiv preprint arXiv:1606.01540, 2016

[128] Duan Y, Chen X, Houthooft R, et al. Benchmarking deep reinforcement learning for continuous control//Proceedings of the 33rd International Conference on Machine Learning. New York, USA, 2016: 1329-1338

[129] Todorov E, Erez T, Tassa Y. MuJoCo: A physics engine for model-based control//Proceedings of the International Conference on Intelligent Robots and Systems. Vilamoura, Portugal, 2012: 5026-5033

[130] Boyan J. Generalization in reinforcement learning: Safely approximating the value function//Proceedings of the Advances in Neural Information Processing Systems. Denver, USA, 1994: 369-376

[131] Kingma D P, Ba J. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014

[132] Christiano P, Leike J, Brown T B, et al. Deep reinforcement learning from human preferences//Proceedings of the Advances in Neural Information Processing Systems. Long Beach, USA, 2017: 4302-4310

[133] Finn C, Levine S, Abbeel P. Guided cost learning: Deep inverse optimal control via policy optimization//Proceedings of the 33rd International Conference on Machine Learning. New York, USA, 2016: 49-58

[134] Chen T, Givony S, Zahavy T, et al. A deep hierarchical approach to lifelong learning in Minecraft//Proceedings of the 31st AAAI Conference on Artificial Intelligence. San Francisco, USA, 2017: 1553-1561

[135] Teh Y W, Bapst V, Czarnecki W M, et al. Distral: Robust multitask reinforcement learning//Proceedings of the Advances in Neural Information Processing Systems. Long Beach, USA, 2017: 4499-4509

LIU Jian-Wei, Ph. D., associate professor. His main research interests include machine learning, intelligent information processing, analysis, prediction, controlling of complicated nonlinear system, and analysis of the algorithm and the designing.

GAO Feng, M. S. candidate. His main research interest is machine learning.

LUO Xiong-Lin, Ph. D., professor. His main research interests include intelligent control, and analysis, prediction, controlling of complicated system.

Background

Reinforcement learning is a framework in which the agent (or controller) typically optimizes its behavior by interacting with its environment and learn to maximize its expected future rewards from interaction with an environment. The environment is usually modelled as a Markov decision process (MDP). After taking an action in some states according to a policy which is derived from the agent's previous experience, the agent receives a scalar reward from the environment, which gives the agent an indication of the quality of that selected action. An agent behaves according to a policy that specifies a distribution over available actions at each state of the MDP. The agent's principal goal is to find a policy that maximizes the total accumulated rewards, also called the return.

Deep neural networks have achieved unprecedented performance in a wide variety of different application areas; for example image classification, face recognition, human-level concept learning, playing Atari games and AlphaGo.

Deep reinforcement learning combines the benefits of reinforcement learning and deep learning. On one hand, reinforcement learning has a wider scope of applicability since it can handle difficult continuous and discrete optimation problems and find the globally opimal solution. On the other hand, deep learning can extract a good representation at different levels of abstraction, which disentangles better the factors of variations underlying the data.

Although general surveys on reinforcement learning techniques already exist, no survey is specifically dedicated to value function and policy gradient iteration methods in particular.

In this paper we aim to survey and place in broad context a number of issues related to value function and policy gradient iteration methods for finite-state, and infinite-state reinforcement learning problems.

DQN, improved methods based on DQN, DDPG, TRPO, A3C and some corresponding improved methods, which have made their way into many deep reinforcement learning algorithms in the past few years, are amongst the most popular algorithms and constitute the largest part of the deep reinforcement learning algorithms.

In this paper, we give a systematical survey of the value function and policy gradient iteration methods. We point out different value function and policy gradient iteration methods, advantages and disadvantages and their specific application scenes. Finally, we give a discussion on open issues related to value function and policy gradient iteration methods.

This work is supported by the National Natural Science Foundation of China (No. 21676295) and the Science Foundation of China University of Petroleum Beijing (No. 2462018QZDX02).