文章编号：1671-7848(2021)10-1998-07 **DOI:** 10.14107/j.cnki.kzgc.20200033

强化学习在机器博弈上的应用综述

杜康豪[1]，宋睿卓[1]，魏庆来[2]

（1. 北京科技大学 自动化学院，北京 100083；2. 中国科学院 自动化研究所，北京 100190）

摘 要：人工智能是未来科技发展的必然趋势，将会对世界产生巨大的影响，而机器博弈更是人工智能研究的热点内容。目前，解决机器博弈问题最先进的算法都来源于强化学习。强化学习是机器学习最重要的方法之一，主要用来解决决策问题。它具有接近人类思维的学习机制，通过试错的方式同环境发生交互，累积最大奖赏并得到最优策略。博弈具有多种多样的形式，内容也十分广泛，根据不同的标准会产生不同的分类，可以将其分为完全信息博弈和非完全信息博弈，但它们都可以通过强化学习进行解决。

关键词：强化学习；机器博弈；非完全信息博弈；围棋；德州扑克；DOTA2

中图分类号：TP181 **文献标识码**：A

Review of Reinforcement Learning Applications in Machine Games

DU Kang-hao[1], *SONG Rui-zhuo*[1], *WEI Qing-lai*[2]

(1. School of Automation and Electrical Engineering, University of Science and Technology Beijing, Bejing 100083, China;
2. Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China)

Abstract: Artificial intelligence (AI) is the inevitable trend of future scientific and technological development, which will have a tremendous impact on the world. At the same time, machine game is a hot topic in artificial intelligence research. At present, the most advanced algorithms for solving machine game problems are derived from reinforcement learning. Reinforcement learning, with a learning mechanism close to human thinking, is one of the most important methods of machine learning, which is mainly used to solve decision-making problems. It interacts with the environment through trial and error, accumulates the greatest reward and gets the best strategy. Game has a variety of forms and a wide range of contents. According to different standards, different classifications will be produced. Game can be divided into complete information game and incomplete information game, and both of them can be solved by reinforcement learning.

Key words: Reinforcement learning; machine game; incomplete information game; go; Texas Hold'em poker; DOTA2

1 引 言

人工智能(artificial intelligence, AI)的兴起让机器博弈越来越受到学术界的关注，同时伴随着计算机围棋AlphaGo的巨大成功，也在民间掀起了热潮，其中使用的强化学习算法更是广为人知。强化学习(reinforcement learning, RL)[1]是机器学习重要的一种方法，也叫做再励学习或者增强学习，它受生物可以自适应环境变化的启发发展而来。目前解决机器博弈问题最先进的算法都是以强化学习为框架，它的核心机制是以试错的方式同环境发生交互，通过最大化累积奖赏来学习最优策略。强化学习与其他机器学习方法最大的不同在于，强化学习不去告诉Agent如何产生一个最优的动作，而是通过环境交互、累积奖赏信号再去评价之前动作的优劣[2]。也就是说，它是在试探环境、评价动作的过程中学习，这就使强化学习成为了研究自主学习问题非常重要的一种方法，被广泛应用于机器博弈中。本文将综

收稿日期：2020-01-15；**修回日期**：2020-04-30

基金项目：国家自然科学基金资助项目(61304079, 61673054, 61722312, 61873300)

作者简介：杜康豪(1995−)，男，河南许昌人，研究生，主要研究方向为强化学习、最优控制等；宋睿卓(1982−)，女，吉林公主岭人，博士，教授，主要从事智能控制、增强学习等方面的教学与科研工作（本文通信作者，Email: ruizhuosong@foxmail.com）；魏庆来(1979−)，男，辽宁沈阳人，博士，研究员，主要从事智能控制、增强学习等方面的科研工作。

述强化学习的基本框架和技术发展，并深入剖析强化学习在博弈中取得的成就。

2 强化学习

强化学习模型由 4 个部分组成，分别是环境(environment)、状态(state)、奖赏(reward)和动作(action)。在学习过程中，Agent 在当前状态 s 下，根据策略 π 选择并执行一个动作 a，然后状态 s 以概率 P 转移到下一个状态 s'，同时环境把一个奖赏信号 r 反馈给 Agent。在与环境的不断交互中，Agent 通过调整策略 π 来累积奖赏使其到达最大值[3]，并通过值函数来评估策略 π 的好坏。

根据累积奖励方式的不同，基于状态的值函数可以定义为 3 种形式：

$$V^{\pi}\left(S_t\right)=\sum_{k=0}^{\infty}\gamma r_{k+1},0<\gamma\le 1 \tag{1}$$

$$V^{\pi}\left(S_t\right)=\sum_{k=0}^{h} r_t \tag{2}$$

$$V^{\pi}\left(S_t\right)=\lim_{h\to\infty}\left(\frac{1}{h}\sum_{t\to 0}^{h} q_t\right) \tag{3}$$

式中，r 为 t 时刻的即时奖赏；q 为 t 时刻后的累积奖赏。奖赏函数 r 的设计具有技巧性，根据不同情况具有不同的设计思路，而优秀的奖赏函数设计方案可以有效地加速强化学习的训练速度。例如 Dong 等[4]通过李雅普诺夫稳定性理论进行了奖赏设计(reward shaping)，加快强化学习训练速度； Goyal 等[5]结合自然语言进行奖赏设计。

另外，式中的 γ 为折扣因子，它代表未来奖赏对于当前奖赏的重要程度。最优策略可以通过下式得到：

$$\pi^{*}=\arg\max_{\pi} V^{\pi}\left(s\right),\forall s\in S \tag{4}$$

另一种状态-动作值函数可以定义为

$$Q^{\pi}\left(s,a\right)=\sum_{s'}P_{ss'}^{a}\left[R_{ss'}^{a}+\gamma\sum_{a'}Q^{\pi}\left(s',a'\right)\right] \tag{5}$$

P 为转移概率。根据动态规划理论 Bellman 最优方程确定最优策略，公式如下：

$$\pi^{*}=\arg\max_{a\in A} Q^{\pi}\left(s,a\right) \tag{6}$$

强化学习的研究起源很早，1956 年，Bellman 提出了动态规划(dynamic programming, DP)方法，在求解马尔可夫决策过程时使用了类似强化学习迭代求解的方式。因此，许多研究者也将动态规划方法视为强化学习的起源，并且往往用马尔可夫决策过程来定义强化学习。1965 年，M. D. Waltz 和 K. S. Fu 在控制理论中独立提出强化学习的概念，并且明确了强化学习的核心机制。强化学习方法的发展历史见表 1。

表 1 强化学习的发展历史
Tab. 1 History of reinforcement learning

年度	事件
1954 年	Minsky[6]首次提出强化学习等术语。
1955 年	Bush 等[7]提出奖励、惩罚等术语。
1956 年	Bellman[8]提出动态规划方法。
1965 年	Waltz 等[9]在控制理论中独立提出强化学习的概念。
1977 年	Werbos[10]提出自适应动态规划方法。
1988 年	Sutton 等[11]提出时间差分(temporal difference, TD)算法。
1992 年	Watkins[12]提出 Q 学习算法。
1994 年	Rummery[13]提出 SARSA 学习算法。
1995 年	Bersekas[14]提出解决随机过程中优化控制的神经动态规划方法。
2006 年	Kocsis[15]提出置信上限树算法。
2007 年	Zinkevich 等[16]提出虚拟遗憾最小化(counterfactual regret minimization, CFR)算法。
2014 年	Silver 等[17]提出确定性策略梯度算法。
2015 年	DeepMind 公司提出深度 Q 网络(deep Q-network, DQN)算法[18]。
2016 年	Heinrich 等[19]提出神经虚拟自我对弈(neural fictitious self play, NFSP)算法。
2017 年	OpenAI 公司提出近端策略优化(proximal policy optimization, PPO)算法[20]。

3 强化学习在完全信息博弈中的应用

博弈(game)是真实世界普遍存在的现象，可以被定义为一定条件和规则下，多人互动决策获取收益的过程，也常常将其称为游戏。人工智能兴起以来，人们就不断尝试用计算机模拟博弈环境，并推动现实问题的解决，因此机器博弈成为了一个重要的研究方向。根据游戏状态是否完全可见，博弈可以分为完全信息博弈(complete information games)，如象棋、围棋，和非完全信息博弈（如扑克）。根据游戏状态是否完全由玩家决定，博弈可以分为确定性博弈和非确定性博弈（如西洋双陆棋）。机器博弈的核心是搜索算法，它与一定规则和局面评估机制结合构成一个完整的博弈程序。随着博弈难度的提升，需要不断增加搜索的深度和广度，人们越来越重视利用解决连续决策问题的强化学习方法去提高机器博弈水平。

强化学习在机器博弈中的成功应用必须要解决两个问题。一是如何描述问题、表达游戏状态；二是如何寻找策略(policy)。下文我们将介绍强化学习在解决机器博弈问题时的巧妙应用。

3.1 西洋双陆棋

西洋双陆棋是靠掷骰子决定走棋步数的棋盘

类游戏，这是一种存在非确定性的完全信息博弈，同时也是强化学习最早尝试解决的一个复杂博弈系统。虽然强化学习的起源很早，但是在解决博弈问题时却面临相当大的困境。一个障碍是传统的强化学习都是基于有限离散状态的，使用线性函数来描述问题。而机器博弈往往起源于复杂的现实问题，其状态、动作空间常常是连续的，线性函数逼近已经不足以解决问题。另一个障碍则是强化学习算法自身所固有的。在强化学习中，Agent 接受环境反馈的奖赏信号再进行决策，这种方式在训练中会形成奖赏延迟，经常进行多步的动作后才可获得奖赏信号。这就是“时间信度分配”问题，即不同时刻的行动在更新值函数时需要给其分配多少误差信号。1992 年，IBM Thomas J. Watson 研究中心首次使用名为 TD(λ)的强化学习方法训练得到 TD-Gammon，一种能够在西洋双陆棋中达到专家水平的算法[21]。这是强化学习第一次突破这两个问题的限制，并成功应用于机器博弈当中。

① 非线性函数逼近。传统强化学习中利用一张表记录状态动作对所对应的值，通过迭代公式进行更新。从表格值中寻找输入与输出的线性关系。这种方法无法描述更复杂的问题。而 TD-Gammon 算法利用多层感知机(multi-layer perceptron, MLP)即一般的神经网络组成非线性函数逼近器，它是含有一个隐藏层的三层神经网络，棋盘状态最终经由这个网络实现从原始数据到价值函数估计的映射。西洋双陆棋的棋盘状态将由 198 个神经元代表为输入端，中间的隐层有 40~80 个神经元，并辅以人工设计的规则表达出值函数。从理论上讲，这种方法将具有良好的泛化性，其映射关系可以推广至任何函数定义域，避免了传统强化学习中利用表格值来记录状态动作对的巨大工作量以及映射（值函数）设计困难的问题。

② 时间信度分配。TD(λ)是 Richard S. Sutton 发明的一种时间差分学习算法[11]，用来解决时间信度分配问题。在 TD-Gammon 的训练过程中，每一个 time step 对应一次半移(ply 或 half-move)，使用 TD(λ)算法来改变网络的权重。权重迭代公式如下：

$$\boldsymbol{w}_{t+1}-\boldsymbol{w}_t=\partial\left(Y_{t+1}-Y_t\right)\sum_{k=1}^{t}\lambda^{t-k}\nabla_{\boldsymbol{w}}Y_k \tag{7}$$

式中，∂ 为学习速率；$\boldsymbol{w}$ 为网络的权重向量；$\nabla_{\boldsymbol{w}}Y_k$ 为神经网络输出对于权重的梯度，即改变权重对输出的影响程度；λ 为时间信度分配的启发式参数，它决定了每一个 time step 中检测到的误差如何反馈并修正先前的估计值。当 $\lambda=0$ 时，在当前 time step 之外没有反馈发生；而当 $\lambda=1$ 时，误差在没有任何衰减的情况下反馈。

TD-Gammon 不采用任何领域知识，完全使用强化学习算法进行自我训练，在双陆棋上取得了出人意料的成果。其中，初版 TD-Gammon 1.0，经过 30 万次自我训练，使用 80 个隐层神经元，达到了当时计算机程序的最佳水平。TD-Gammon 2.0 使用两步搜索，40 个隐层神经元，并在 1992 年世界双陆棋锦标赛中亮相，与人类顶级玩家 Kent Goulding、Kit Woolsey、Wilcox Snellings，以及前世界冠军 Joe Sylvester、Joe Russell 进行了 38 场比赛，TD-Gammon 2.0 仅净损失 7 分。在随后的升级版 2.1 中，结合两步搜索，使用 80 个隐层神经元，经过 150 万次游戏的训练，TD-Gammon 2.1 达到了顶级玩家的水平，40 局比赛中几乎已全胜的结果战胜了 Bill Robertie。

与这些成果相比，学术界更加热衷于提取 TD-Gammon 在机器博弈中成功的基本原理，并尝试将其扩展到其他领域。但是当 TD 算法迁移到其他象棋、围棋游戏时，却并不成功，相关研究表明 TD-Gammon 算法在双陆棋上的成功源于双陆棋本身所存在的较大随机性与算法本身的契合。同时 Sutton 所进行的分析仅仅针对于马尔可夫决策过程，虽然他指出非马尔可夫过程也可以适用于此框架，但实际上，解决复杂问题时使用非线性函数逼近与无模型算法结合，经常导致不能收敛的情况。并且在泛化过程中，任务的复杂性使得输入空间和搜索空间巨大，也导致算法不能得到良好的效果。因此，随后的研究中，人们又把更多的注意力放在能够更好保证收敛性的线性函数逼近器上。

3.2 Atari 游戏

Atari 游戏是一些视频小游戏的合集，它们都基于高 210 像素、长 160 像素的屏幕，这些游戏共定义了 18 个离散的动作，大多属于单人博弈，玩家通过手柄（即执行动作）来累积获得收益[22]。这使它成为了试验强化学习、探索机器博弈的理想场地。在双陆棋中获得成功的 TD 算法却并不适用于 Atari 游戏。首先是训练稳定性问题。TD 算法是一种在线策略(on-policy)方法，它基于值函数迭代求解最优策略，在训练过程中，每一次的迭代都会导致值函数的微小变化，往往改变了以前迭代的状态动作对的表格值，容易造成结果的波动甚至发散，或者陷入局部收敛。同时这种值函数法只能得到确定性策略，但有时候最优策略却是随机的。此外，这种方法也不适合求解连续动作问题。对于 Atari 游戏这样需要一系列连续动作的场景，值函数迭代的方法将会需要巨大的状态动作空间，很难寻找到奖赏值

最大的策略，并且极大拖缓训练速度。2013 年，Google DeepMind 提出了 DQN 算法[18]，采用了经验回放机制和策略梯度法解决这两个问题。

① 稳定性。与 TD 算法这样的在线策略(on-policy)算法不同，DQN 属于离线策略(off-policy)算法，使用均匀分布的经验回放(experience replay)[23]机制。经验回放相当于构建了一个经验池(pool of stored samples)，将每一个 time-step 的状态与动作，记为 $e_t=(s_t,a_t,r_t,s_{t+1})$，存储到一个回放记忆(replay memory)单元 $U=\{e_1,e_2,\cdots,e_t\}$ 中。训练时，定期从经验池 U 中随机抽取样本，使用 Q-learning 更新或者小批量更新，并且采用一种贪婪策略(ε-greedy policy)选择并执行动作。为减少计算难度，DeepMind 采用了固定长度的经验数据。当其采用 Q-learning 更新时，损失函数[24]如下：

$$\nabla_{\theta_i}L_i(\theta_i)=E_{(s,a,r,s')\sim\varepsilon(U)}\times[(r+\gamma\max_{a'}Q(s',a';\theta_{i-1})-Q(s,a;\theta_i))\nabla_{\theta_i}Q(s,a;\theta_i)] \tag{8}$$

式中，(s,a,r,s') 为存储样本的随机抽样。

这种方法使得训练时可以从过往数据中提取样本，有效地降低由于新数据加入造成的波动性，并且在自我博弈训练获得最优策略之外，还可以通过观测他人的博弈数据进行学习。后来的研究人员又提出带有优先级的经验重播(experience replay)[28]技术，增大那些具有高优先级经验数据的选择概率，并且在实验中取得了更好的结果。

② 连续动作决策。DQN 采用策略梯度法来代替值函数迭代，不仅有效解决值函数迭代过程中造成的动作选择的不稳定性，还有效解决连续动作决策问题。策略梯度法利用梯度更新策略参数，通过调整动作概率调整策略，而不去计算奖励并更新表格值，避免因连续状态动作空间庞大而无法计算的情况，在一定条件下收敛性必然可证。DQN 使用随机梯度下降法(stochastic gradient descent, SGD)来进行参数更新，公式如下：

$$\theta'=\theta_j-\frac{\sigma J(\theta)}{\sigma\theta_j}=\theta_j-\alpha\frac{1}{m}\sum_{i=1}^{m}\left(y^i-h_\theta\left(x^i\right)\right)x_j^i \tag{9}$$

同时为避免 SGD 中由链式法则造成的梯度消失问题，Hochreiter 等[25]在 DQN 中加入了长短期记忆(long short term memory, LSTM)模块，这种结合不仅解决了梯度消失的问题，还使算法能够综合历史局面给出更优的动作决策。

2015 年，DeepMind 完善了 DQN，让该算法在雅达利游戏上取得更好的成绩，登上 Nature 封面。但是在标准的强化学习算法 Q-learning 以及 DQN 上，使用相同的值来选择和评估动作，使得网络更容易选择过拟合值，导致次优估计值。DeepMind 研究团队针对 DQN 进行改进工作，提出各种优化算法。Van[26]等提出的双重 DQN，将选择从评价中独立出来，有效解决了估计偏差过高的问题；Wang 等[27]提出决斗网络(dueling network)架构，通过分别呈现状态值和动作优势函数，实现不同行为之间的泛化；Schaul 等[28]提出优先经验回放(prioritized experience replay)技术，用带有优先级的经验提升数据的效率；Bellemare 等[29]提出分类 DQN，通过建立反馈折扣的类别分布，让估计结果更细致可信；Mnih 等[30]提出异步优势行动者-评论家(asynchronous advantage actor-critic, A3C)算法，不再需要经验回放机制，极大提升了训练速度，并且它在 Atari 游戏上的平均成绩是 DQN 的 4 倍； 2017 年，Hessel 等[31]在论文中提出 Rainbow 方法，它将上述方法进行整合，在 57 个 Atari 2600 游戏组成的基准测试环境中在数据效率和最终结果上都达成了新的业界最佳水平，这是机器博弈发展的又一新高度。

3.3 围棋

围棋是所有棋类博弈中复杂度最高的，也是完全信息博弈的一个高峰。围棋的棋盘是 19×19，共有 361 个交叉点，第 i 步的走法就有 $362-i$ 种，状态空间复杂度为 2×10^{170}，是所有棋类博弈中最高的。1983 年，理论计算机专家证明了围棋属于探索-利用问题[32]，也就是说，评估盘面最优走步的计算复杂度必然随考虑步数成指数级增长。早期的研究者致力于通过缩小搜索空间来解决问题，但由于计算力的局限，这种做法并没有取得理想的成绩。即使在计算资源发达的今天，依靠蛮力搜索仍然不是解决计算机围棋的最佳选择。计算机围棋更大的障碍在于没有一个可靠的评估方法。围棋中棋子的组合状态复杂，任何两局博弈之间的差异性十分巨大，特征提取与整合过程更加困难，这导致围棋盘面不能通过一套泛化的规则体系进行评估。因此，如何进行巨空间搜索、如何设计局面评估，成了横亘在研究者面前的难题。2016 年，DeepMind 发布围棋博弈系统 AlphaGo[33]，将强化学习与蒙特卡洛树搜索创造性地结合起来，为复杂机器博弈探索出一条新的道路。接下来，我们将介绍 AlphaGo 如何巧妙地处理博弈中存在的探索-利用困境和局面评估难题的。

① 探索-利用困境。围棋是一个探索-利用问题，要求的计算量是海量的。AlphaGo 构造了相互独立的策略网络和价值网络，通过策略网络产生下一步

棋的概率分布，确定落子位置，通过价值网络评估落子位置，得出落子胜率。AlphaGo 在学习阶段利用深度卷积网络表示策略，再结合自我博弈使用强化学习得到策略网络，该策略网络将缩小高概率落子位置的搜索范围，提升搜索效率，同时通过强化学习过程中得到的约 3 000 万盘棋局的数据训练价值网络。这个价值网络用来估值当前盘面状态，并会预测该局对弈的胜方，有效降低了搜索深度。同时它采用蒙特卡洛树搜索，给出一个期望的置信区间，让当前期望不高的盘面也有机会被探索到，从而在探索与利用之间收获平衡。

② 动态评估。局面评估方法有静态评估和动态评估。静态评估通过完备的专家知识设计规则，对局面进行打分。而动态评估通过不断与环境交互，更新估值函数来估计局面。对于国际象棋这样棋子独立性较强的博弈，我们可以使用静态评估，但是围棋中静态评估并不可靠。因此，AlphaGo 使用了蒙特卡洛树搜索这种动态评估方法。该方法基于统计学原理，对当前盘面的所有可能结果进行反复抽样，并得到解的概率分布。同时蒙特卡洛树搜索会给出当前走步的估值 Z_L。AlphaGo 还设计一个独立的价值网络产生一个估值 $v_\theta(S_L)$，这两个值通过加权的方式得出当前走法的评估结果 $V(S_L)$。并将该值作为当前走步的 Q 值[34]：

$$Q(s,a)=\frac{1}{N(s,a)}\sum_{i=1}^{n}L(s,a,i)V\left(s_L^i\right) \tag{10}$$

式中，$L(s,a,i)$ 为第 i 次抽样中是否抽到了状态动作对 (s,a)。程序在反复抽样中不断调整策略，最终将侧重于选择那些一定步数内更有胜率的下法。

AlphaGo 采用蒙特卡洛树搜索[35]，结合了广度优先搜索和深度优先搜索，让计算机围棋侧重那些一定步数内更有胜率的下法上，根据这种方法，AlphaGo 取得了超过人类顶尖大师的棋力，并在世界范围内掀起了研究强化学习的热潮。

随后，在 AlphaGo 的改进版本 AlphaGo Zero 中，DeepMind 团队使用了一种完全独立的强化学习算法[36]，取得了更惊艳的成果。该版本用一个神经网络代替了策略网络和价值网络，并且摒弃了蒙特卡洛方法，仅使用简化版的树搜索方式估计落子位置和落子胜率。训练过程中，AlphaGo Zero 从当前盘面状态 s_t 开始，使用基于深度神经网络 f_θ 的树搜索得出落子策略 π_t，其中 θ 是神经网络 f_θ 的参数，并通过神经网络输出策略函数 p_t 和估值 v_t。程序通过策略 π_t 选择动作，直到棋局结束获得最终结果 z。然后使用结果 z 和估值 v_t，以及策略 π_t 和 p_t 构成损失函数 l，不断更新神经网络的参数。

由此不断在自我博弈中迭代以增强搜索的结果。其损失函数如下：

$$(p_t,v_t)=f_\theta(s_t) \tag{11}$$

$$l=(z-v_t)^2-\pi_t^T\log p_t+c\|\theta\|^2 \tag{12}$$

AlphaGo Zero 作为一个解决了实际问题的成功案例，表明了强化学习算法在复杂机器博弈下的出色表现，同时 AlphaGo Zero 也给出了强化学习的稳定性和收敛性的有效探索。即在搜索过程中，根据预测结果进行学习，稳定提升强化学习的训练过程。但围棋属于完全信息的二人博弈游戏，此外还有包含众多未知性的非完全信息博弈游戏，例如，德州扑克游戏和多人博弈游戏，都期待着深度强化学习做出进一步突破。

4 强化学习在非完全信息博弈上的应用

4.1 德州扑克

德州扑克(Texas Hold'em poker)是一种多玩家贯续互动的游戏模型，属于扩展式博弈，在对弈过程中只能看到己方牌面和公共牌信息，而无法掌握对方牌面信息，又属于非完全信息博弈。扩展式博弈经常使用博弈树形式来展示，博弈树节点表示可能的博弈状态，博弈树的边表示进行的动作。而且博弈树的展开复杂度极高，扩展得到的信息集远远小于实际可能的游戏状态数量。因此必须设计估值算法，在经历有限游戏状态后近似扩展到最后的回报值。与完全信息博弈求解策略相比，非完全信息博弈最大的不同在于其信息隐藏，无法得知确切的游戏状态。

因此，在非完全信息决策上必须增加随机探索，并且在求解最优策略时加入这种影响。面对这种牌面隐藏，对方策略未知，不可能做出最佳决策的问题时，人们转向纳什均衡求解。2007 年，Zinkevich 等[16]利用 CFR 为两人零和博弈提出了近似纳什均衡解。2015 年，Bowling[37]利用改进的 CFR+完全解决了两人限制性德州扑克，这是人类解决的第一个非完全信息博弈。该方法是通过两个遗憾最小化算法自我博弈对近似纳什均衡迭代求解。卡耐基梅隆大学的 Libratus 依靠这种算法击败了人类顶级玩家。但该方法对多人博弈和非零和博弈并不适用。Heinrich 等[19]提出了一种引入了神经网络的强化学习方法 NFSP 算法，在自我博弈中近似均衡求解。NFSP 算法通过两个网络的训练，直接将随机探索环节剥离出来，使用 off-policy 的强化学习方法（如 Q-learning、DQN），来训练一个行动值神经网络 F_Q，

预测最大化行动值，其迭代公式如下：

$$Q^i(s,a) \approx E_{\beta^i,\sigma^{-i}}\left[G_t^i \mid S_t = s, A_t = a\right] \tag{13}$$

该网络用来定义近似最优策略：

$$\beta = \varepsilon - \text{greedy}(F_Q)$$

同时训练另一个策略神经网络来定义扰动平均策略 π，最终从 $\sigma \equiv (1-\eta)\hat{\pi} + \eta\left(\hat{\beta}\right)$ 中选择动作，其中，$\eta \in \mathbf{R}$ 是预测参数(anticipatory parameter)。该方法可以在无先验知识条件下学习，不依赖局部搜索，并且收敛到近似纳什均衡。

4.2 DOTA2

DOTA2 是一种多人在线战术竞技类(multiplayer online battle arena, MOBA)游戏，类似于现实世界中的人类对抗，游戏任务是击败对方英雄并且摧毁敌方基地，涉及攻防对抗、资源分配、部队选择等具有现实意义的复杂博弈。电子竞技游戏一般都属于非完全信息博弈，每一方都只能掌握全局的部分视野，同时，游戏过程在实时动态下进行，拥有远多于围棋的规则和远超围棋的复杂度，这对机器学习的决策能力、局面评估提出了更高的要求。2018 年 8 月 OpenAI Five 在限定条件的 5V5 比赛中连胜两局，击败了排名在 99.5%之前的 Dota2 半职业选手队。OpenAI Five 使用了称为近端策略优化(proximal policy optimization, PPO)的强化学习算法[20]。该算法为强化学习提出了新的策略梯度法，在环境交互中进行采样，然后利用随机梯度上升法优化代理目标函数(“surrogate” objective function)，通过在两个步骤之间循环迭代进行学习。

PPO 使用了一种新的目标函数：

$$L^{clip}(\theta) = \hat{E}_t\left[\min\left(r_t(\theta)\hat{A}_t, clip\left(r_t(\theta), 1-\varepsilon, 1+\varepsilon\right)\hat{A}_t\right)\right] \tag{14}$$

式中，θ 为策略参数；$\hat{E}_t$ 为 timesteps 上的经验期望；r_t 为新旧策略下的概率比；$\hat{A}_t$ 为时间 t 下的优势估计；ε 为一个超参数；*clip* 项用来保证 r_t 不偏离其所定义的区间 $[1-\varepsilon, 1+\varepsilon]$，可以通过下式得到代理目标函数。

$$L_t^{clip+VF+S}(\theta) = \hat{E}_t\left[L_t^{clip}(\theta) - c_1 L_t^{VF}(\theta) + c_2 S\left[\pi_\theta\right](s_t)\right] \tag{15}$$

式中，c_1、c_2 为系数；L_t^{VF} 为平方误差 $\left(V_\theta(s_t) - V_t^{t\arg}\right)^2$；$S$ 为熵奖励(entropy bonus)，用来增加探索性。

在 PPO 算法的每一次迭代中，先通过 n 个并行的 actor 采集 T 个 timestep 的数据，再用这 nT 个数据构造代理损失函数。然后，与传统的策略梯度法不同，该算法会进行多 epoch 的小批量更新，优化代理函数并更新策略参数。

它实现了一种与随机梯度下降兼容的信任区域(trust region)[38]更新方法，为了简化算法，移除 KL(Kullback-Leibler)惩罚项及它的自适应升级功能。结果 PPO 算法对连续决策问题具有很好的表现。虽然 OpenAI Five 战胜了半职业玩家，但这远称不上是突破性的进展。在这场比赛中，OpenAI Five 限制了英雄池的数量，仅余 18 个英雄可选，使得人类玩家阵容可选择性大幅降低，同时在游戏中缺少了数项重要的游戏道具。

面对状态空间大，局面视野可见，动作空间大，时间尺度长的挑战，OpenAI Five 使用 PPO 的强化学习算法就取得了令人满意的成果。通过进一步将 PPO 算法与蒙特卡洛树搜索、分层强化学习等方法结合，OpenAI Five 还会有巨大的提升空间。

5 结　论

机器博弈是当前人工智能研究的重点，也是人工智能发展的试金石。强化学习作为解决机器博弈问题的重要方法之一，从简单到复杂、从完全信息博弈到非完全信息博弈，它不断展现出自身的创造力，通过与其他方法的糅合攻克了诸多复杂的博弈问题。未来，非完全信息博弈将会成为机器博弈理论和实践发展的重要方向，其体现出的科研价值和商业价值会吸引越来越多的研究者投入到该领域的研究，例如军事智能博弈对抗、能源智能电网调度、农田水利管理、商务谈判与运营等诸多方面。当前，非完全信息博弈的研究还处于发展阶段，强化学习在其中的应用也面临着一些困境，例如，如何在复杂的局面下设置奖赏信号来得到最优策略，如何在巨搜索空间下处理计算复杂度问题，如何使用更少的数据和训练次数达到良好的性能等。同时，人们也迫切希望机器博弈的成果尽早地应用于现实问题中，服务人类。

参考文献(References)

[1] 尤树华, 周谊成, 王辉. 基于神经网络的强化学习研究概述[J]. 电脑知识与技术, 2012, 8(28): 6782-6786.
You S H, Zhou Y C, Wang H. Research on Reinforcement Learning Based on Neural Network: a Summary[J]. Computer Knowledge and Technology, 2012, 8(28): 6782-6786.

[2] 张汝波, 顾国昌, 刘照德, 等. 强化学习理论、算法及应用[J]. 控制理论与应用, 2000,17(5): 637-642.
Zhang R B, Gu G C, Liu Z D, et al. Reinforcement Learning Theory, Algorithms and Its Application[J]. Control Theory and Applications, 2000, 17(5): 637-642.

[3] 刘忠，李海红，刘全. 强化学习算法研究[J]. 计算机工程与设计, 2008, 22: 5805-5809.
Liu Z, Li H H, Liu Q. Research on Algorithm of Reinforcement Learning[J]. Computer Engineering and Design, 2008, 22: 5805-5809.

[4] Dong Y L, Tang X C, Yuan Y. Principled Reward Shaping for Reinforcement Learning via Lyapunov Stability Theory[J]. Neuro-computing, 2020, 393: 83-90.

[5] Goyal P, Niekum S, Mooney R J. Using Natural Language for Reward Shaping in Reinforcement Learning[J]. Twenty-eighth International Joint Conference on Artificial Intelligence, 2019: 2385-2391.

[6] Minsky M L. Theory of Neural Analog Reinforcement Systems and Its Application to the Brain Model Problem[D]. New Jersey: Princeton University, 1954.

[7] Bush R R, Mosteller F. Stochastic Models for Learning[M]. New York: Wiley, 1955: 128-152.

[8] Bellman R. Dynamic Programming and Lagrange Multipliers[J]. Proceedings of the National Academy of Sciences, 1956, 42(10): 767-769.

[9] Waltz M D, Fu K S. A Heuristic Approach to Reinforcement Learning Control Systems[J]. IEEE Transactions on Automatic Control, 1965, 4: 390-398.

[10] Werbos P J. Advanced Forecasting Methods for Global Crisis Warning and Models of Intelligence[J]. General Systems Yearbook, 1977, 22: 25-38.

[11] Sutton R S. Learning to Predict by the Methods of Temporal Differences[J]. Machine Learning, 1988, 3(1): 9-44.

[12] Watkins C J C H, Dayan P. Technical Note: Q-learning[J]. Machine Learning, 1992, 8(3,4): 279-292.

[13] Rummery G A, Niranjan M. On-line Q-learning Using Connectionist Systems[J]. Technical Report, 1994: 1-7.

[14] Bertsekas D P, Tsitsiklis J N. Neuro-dynamic Programming: an Overview[C]. New Orleans: the 34th IEEE Conference on Decision and Control, 1995.

[15] Kocsis L, Szepesvari C. Bandit Based Monte-carlo Planning[C]. Berlin: the European Conference on Machine Learning, 2006.

[16] Zinkevich M, Johanson M, Bowling M, et al. Regret Minimization in Games with Incomplete Information[C]. Kitakyushu: International Conference on Neural Information Processing Systems, 2007.

[17] Silver D, Lever G, Heess N, et al. Deterministic Policy Gradient Algorithms[C]. Beijing: the International Conference on Machine Learning, 2014.

[18] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level Control Through Deep Reinforcement Learning[J]. Nature, 2015, 518: 529-533.

[19] Heinrich J, Silver D. Deep Reinforcement Learning from Self-play in Imperfect-information Games[J]. E-print arXiv, 2016, 2: 1603-1604.

[20] Schulman J, Wolski F, Dhariwal P, et al. Proximal Policy Optimization Algorithms[J]. E-print arXiv, 2017, 2: 1707-1709.

[21] Tesauro G. Temporal Difference Learning and TD-gammon[J]. Communications of the ACM, 1995, 38(18): 88.

[22] 郭潇逍，李程，梅俏竹. 深度学习在游戏中的应用[J]. 自动化学报, 2016, 42(5): 676-684.
Guo X X, Li C, Mei Q Z. Deep Learning Applied to Games[J]. Acta Automatica Sinica, 2016, 42(5): 676-684.

[23] Lin L J. Reinforcement Learning for Robots Using Neural Networks [J]. Ph.d.thesis Carnegie Mellon University, 1992: 52-74.

[24] 石征锦，王康. 深度强化学习在 Atari 视频游戏上的应用[J]. 电子世界, 2017(16): 105-106,109.
Shi Z J, Wang K. The Application of Depth of Reinforcement Learning in the Vedio Game[J]. Electronics World, 2017(16): 105-106,109.

[25] Hochreiter S, Schmidhuber J. Long Short-term Memory[J]. Neural Computation, 1997, 9(8): 1735-1780.

[26] Van Hasselt H, Guez A, Silver D. Deep Reinforcement Learning with Double Q-Learning[C]. Phoenix: the 30th AAAI Conference on Artificial Intelligence, 2016.

[27] Wang Z, Schaul T, Hessel M, et al. Dueling Network Architectures for Deep Reinforcement Learning[C]. New York: the 33rd International Conference on Machine Learning (ICML), 2016.

[28] Schaul T, Quan J, Antonoglou I, et al. Prioritized Experience Replay[J]. Computer Science, 2015, 4: 1-4.

[29] Bellemare M G, Dabney W, Munos R. A Distributional Perspective on Reinforcement Learning[C]. Sydney: the 34th International Confer-ence on Machine Learning (ICML), 2017.

[30] Mnih V, Badia A P, Mirza M, et al. Asynchronous Methods for Deep Reinforcement Learning[C]. New York: the 33rd International Confer-ence on Machine Learning (ICML), 2016.

[31] Hessel M, Modayil J, Van Hasselt H, et al. Rainbow: Combining Improvements in Deep Reinforcement Learning[J]. The Thirty-second AAAI Conference on Artificial Intelligence, 2017(18): 3215-3222.

[32] Robson J M. The Complexity of Go[J]. Proc Ifip, 1983, 9: 413-417.

[33] Silver D, Huang A, Maddison C J, et al. Mastering the Game of Go with Deep Neural Networks and Tree Search[J]. Nature, 2016, 529: 484-489.

[34] 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[J]. 控制理论与应用, 2016, 33(6): 701-717.
Zhao D B, Shao K, Zhu Y H, et al. Review of Deep Reinforcement Learning and Discussions on the Development of Computer Go[J]. Control Theory & Applications, 2016, 33(6): 701-717.

[35] Kocsis L. Bandit Based Monte-carlo Planning[C]. Heidelberg: the 17th European Conference on Machine Learning, 2006.

[36] Silver D, Schrittwieser J, Simonyan K, et al. Mastering the Game of Go Without Human Knowledge[J]. Nature, 2017, 550: 354-359.

[37] Bowling M, Burch N, Johanson M, et al. Heads-up Limit Hold'em Poker is Solved[J]. Science, 2015, 347(6218): 145-149.

[38] Bellemare M G, Naddaf Y, Veness J, et al. The Arcade Learning Environment: an Evaluation Platform for General Agents[J]. The Journal of Artificial Intelligence Research, 2013, 47(1): 253-279.