3 强化学习算法分类

本章将介绍强化学习算法的常见分类方式和具体类别。图 3.1 总结了一些经典的强化学习算法，并从多个角度对强化学习算法进行分类，其中包括基于模型（Model-Based）和无模型的（Model-Free）学习方法，基于价值（Value-Based）和基于策略的（Policy-Based）学习方法（或两者相结合的 Actor-Critic 学习方法），蒙特卡罗（Monte Carlo）和时间差分（Temporal-Difference）

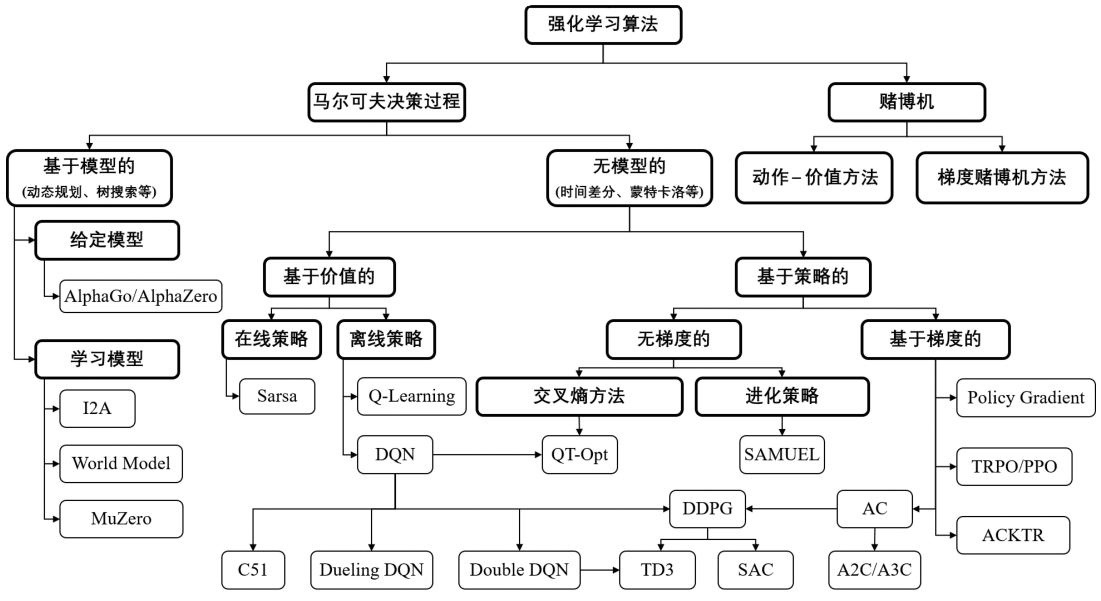

图 3.1　强化学习算法分类图。加粗方框代表不同分类，其他方框代表具体算法

学习方法，在线策略（On-Policy）和离线策略（Off-Policy）学习方法。大多数强化学习算法都可以根据以上类别进行划分，希望在介绍具体的强化学习算法之前，这些分类能帮助读者建立强化学习知识体系框架。其中，第 4、5 和 6 章分别具体介绍了基于价值的方法、基于策略的方法，以及两者的结合。

3.1 基于模型的方法和无模型的方法

我们首先讨论基于模型的方法和无模型的方法，如图 3.2 所示。什么是“模型”？在深度学习中，模型是指具有初始参数（预训练模型）或已习得参数（训练完毕的模型）的特定函数，例如全连接网络、卷积网络等。而在强化学习算法中，“模型”特指环境，即环境的动力学模型。回想一下，在马尔可夫决策过程（MDP）中，有五个关键元素：$\mathcal{S}, \mathcal{A}, P, R, \gamma$。$\mathcal{S}$ 和 $\mathcal{A}$ 表示环境的状态空间和动作空间；P 表示状态转移函数，$p(s'|s,a)$ 给出了智能体在状态 s 下执行动作 a，并转移到状态 s' 的概率；R 代表奖励函数，$r(s,a)$ 给出了智能体在状态 s 执行动作 a 时环境返回的奖励值；γ 表示奖励的折扣因子，用来给不同时刻的奖励赋予权重。如果所有这些环境相关的元素都是已知的，那么模型就是已知的。此时可以在环境模型上进行计算，而无须再与真实环境进行交互，例如第 2 章中介绍的值迭代、策略迭代等规划（Planning）方法。在通常情况下，智能体并不知道环境的奖励函数 R 和状态转移函数 $p(s'|s,a)$，所以需要通过和环境交互，不断试错（Trials and Errors），观察环境相关信息并利用反馈的奖励信号来不断学习。这个不断学习的过程既对基于模型的方法适用，也对无模型的方法适用。

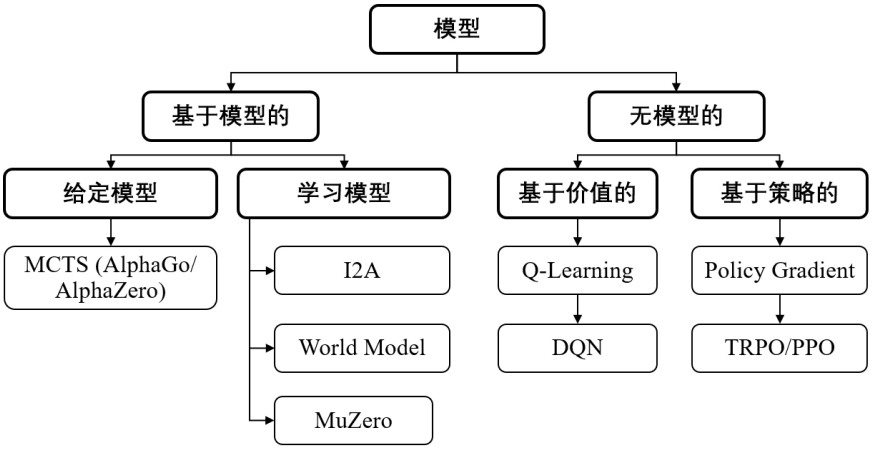

图 3.2 基于模型的方法和无模型的方法

在这个不断试错和学习的过程中，可能有某些环境元素是未知的，如奖励函数 R 和状态转移函数 P。此时，如果智能体尝试通过在环境中不断执行动作获取样本 (s,a,s',r) 来构建对 R 和 P 的估计，则 $p(s'|s,a)$ 和 r 的值可以通过监督学习进行拟合。习得奖励函数 R 和状态转移函数 P

之后，所有的环境元素都已知，则前文所述的规划方法可以直接用来求解该问题。这种方式即称为基于模型的方法。另一种称为无模型的方法则不尝试对环境建模，而是直接寻找最优策略。例如，Q-learning 算法对状态-动作对 (s, a) 的 Q 值进行估计，通常选择最大 Q 值对应的动作执行，并利用环境反馈更新 Q 值函数，随着 Q 值收敛，策略随之逐渐收敛达到最优；策略梯度（Policy Gradient）算法不对值函数进行估计，而是将策略参数化，直接在策略空间中搜索最优策略，最大化累积奖励。这两种算法都不关注环境模型，而是直接搜索能最大化奖励的策略。这种不需要对环境建模的方式称为无模型的方法。可以看到，基于模型和无模型的区别在于，智能体是否利用环境模型（或称为环境的动力学模型），例如状态转移函数和奖励函数。

通过上述介绍可知，基于模型的方法可以分为两类：一类是给定（环境）模型（Given the Model）的方法，另一类是学习（环境）模型（Learn the Model）的方法。对于给定模型的方法，智能体可以直接利用环境模型的奖励函数和状态转移函数。例如，在 AlphaGo 算法 (Silver et al., 2016) 中，围棋规则固定且容易用计算机语言进行描述，因此智能体可以直接利用已知的状态转移函数和奖励函数进行策略的评估和提升。而对于另一类学习模型的方法，由于环境的复杂性或不可知性，我们很难描述整个动力系统的规律。此时智能体无法直接获取模型，可行的替代方式是先通过与环境交互学习环境模型，然后将模型应用到策略评估和提升的过程中。

第二类的典型例子包括 World Models 算法 (Ha et al., 2018)、I2A 算法 (Racanière et al., 2017) 等。例如在 World Models 算法中，智能体首先使用随机策略与环境交互收集数据 (S_t, A_t, S_{t+1})，再使用变分自编码器（Variational Autoencoder，VAE）(Baldi, 2012) 将状态编码为低维潜向量 $\boldsymbol{z}_t$。然后利用数据 (Z_t, A_t, Z_{t+1}) 学习潜向量 $\boldsymbol{z}$ 的预测模型。有了预测模型之后，智能体便可以通过习得的预测模型提升策略能力。

基于模型的方法的主要优点是，通过环境模型可以预测未来的状态和奖励，从而帮助智能体进行更好的规划。一些典型的方法包括朴素规划方法、专家迭代 (Sutton et al., 2018) 方法等。例如，MBMF 算法 (Nagabandi et al., 2018) 采用了朴素规划的算法；AlphaGo 算法 (Silver et al., 2016) 采用了专家迭代的算法。基于模型的方法的缺点在于，存在或构建模型的假设过强。现实问题中环境的动力学模型可能很复杂，甚至无法显式地表示出来，导致模型通常无法获取。另一方面，在实际应用中，学习得到的模型往往是不准确的，这给智能体训练引入了估计误差，基于带误差模型的策略的评估和提升往往会造成策略在真实环境中失效。

相较之下，无模型的方法不需要构建环境模型。智能体直接与环境交互，并基于探索得到的样本提升其策略性能。与基于模型的方法相比，无模型的方法由于不关心环境模型，无须学习环境模型，也就不存在环境拟合不准确的问题，相对更易于实现和训练。然而，无模型的方法也有其自身的问题。最常见的问题是，有时在真实环境中进行探索的代价是极高的，如巨大的时间消耗、不可逆的设备损耗及安全风险，等等。比如在自动驾驶中，我们不能在没有任何防护措施的情况下，让智能体用无模型的方法在现实世界中探索，因为任何交通事故的代价都将是难以承受的。

第 4、5 和 6 章中介绍的算法都是无模型算法，包括深度 Q 网络（Deep Q-Network，DQN）

算法 (Mnih et al., 2015)、策略梯度（Policy Gradient）方法 (Sutton et al., 2000)、深度确定性策略梯度（Deep Deterministic Policy Gradient，DDPG）算法 (Lillicrap et al., 2015) 等。虽然无模型方法仍然是现在的主流方法，但由于其采样效率（Sample Efficiency）低的缺点很难克服，天然具有高采样效率的基于模型的方法发挥着越来越重要的作用（详见第 7 章）。例如，第 15 章中介绍的 AlphaGo (Silver et al., 2016)、AlphaZero (Silver et al., 2017, 2018) 算法，以及最新的 MuZero 算法 (Schrittwieser et al., 2019) 都属于基于模型的方法。

3.2 基于价值的方法和基于策略的方法

回忆第 2 章，深度强化学习中的策略优化主要有两类：基于价值的方法和基于策略的方法。两者的结合产生了 Actor-Critic 类算法和 QT-Opt (Kalashnikov et al., 2018) 等其他算法，它们利用价值函数的估计来帮助更新策略。其分类关系如图 3.3 所示。基于价值的方法通常意味着对动作价值函数 $Q^\pi(s,a)$ 的优化。优化后的最优值函数表示为 $Q^{\pi^*}(s,a)=\max_a Q^\pi(s,a)$，最优策略通过选取最大值函数对应的动作得到 $\pi^* \approx \arg\max_\pi Q^\pi$（“$\approx$”由函数近似误差导致）。

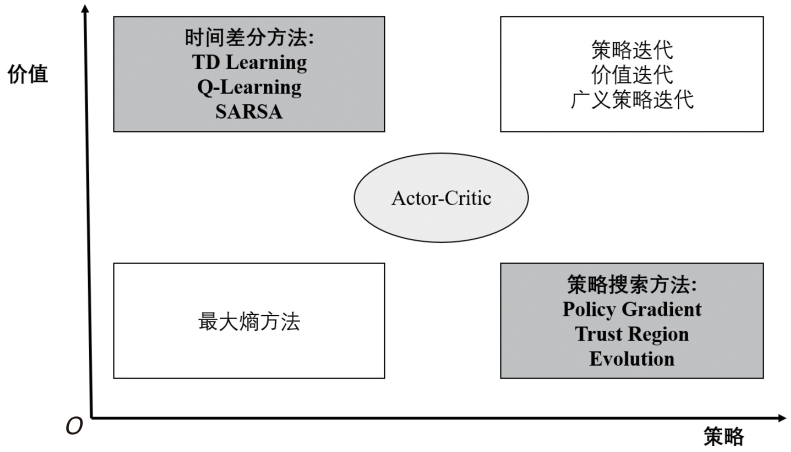

图 3.3 基于价值的方法和基于策略的方法。图片参考文献 (Li, 2017)

基于价值的方法的优点在于采样效率相对较高，值函数估计方差小，不易陷入局部最优；缺点是它通常不能处理连续动作空间问题，且最终的策略通常为确定性策略而不是概率分布的形式。此外，深度 Q 网络等算法中的 ϵ-贪心策略（ϵ-greedy）和 max 算子容易导致过估计的问题。

常见的基于价值的算法包括 Q-learning (Watkins et al., 1992)、深度 Q 网络（Deep Q-Network，DQN）(Mnih et al., 2015) 及其变体：（1）优先经验回放（Prioritized Experience Replay，PER）(Schaul et al., 2015) 基于 TD 误差对数据进行加权采样，以提高学习效率；（2）Dueling DQN (Wang et al., 2016) 改进了网络结构，将动作价值函数 Q 分解为状态值函数 V 和优势函数 A 以提高函数近似能力；（3）Double DQN (Van Hasselt et al., 2016) 使用不同的网络参数对动作进行选择和评估，以

解决过估计的问题；（4）Retrace (Munos et al., 2016) 修正了 Q 值的计算方法，减少了估计的方差；（5）Noisy DQN (Fortunato et al., 2017) 给网络参数添加噪声，增加了智能体的探索能力；（6）Distributed DQN (Bellemare et al., 2017) 将状态-动作值估计细化为对状态-动作值分布的估计。

基于策略的方法直接对策略进行优化，通过对策略迭代更新，实现累积奖励最大化。与基于价值的方法相比，基于策略的方法具有策略参数化简单、收敛速度快的优点，且适用于连续或高维的动作空间。一些常见的基于策略的算法包括策略梯度算法（Policy Gradient，PG）(Sutton et al., 2000)、信赖域策略优化算法（Trust Region Policy Optimization，TRPO）(Schulman et al., 2015)、近端策略优化算法（Proximal Policy Optimization，PPO）(Heess et al., 2017; Schulman et al., 2017) 等，信赖域策略优化算法和近端策略优化算法在策略梯度算法的基础上限制了更新步长，以防止策略崩溃（Collapse），使算法更加稳定。

除了基于价值的方法和基于策略的方法，更流行的是两者的结合，这衍生出了 Actor-Critic 方法。Actor-Critic 方法结合了两种方法的优点，利用基于价值的方法学习 Q 值函数或状态价值函数 V 来提高采样效率（Critic），并利用基于策略的方法学习策略函数（Actor），从而适用于连续或高维的动作空间。Actor-Critic 方法可以看作是基于价值的方法在连续动作空间中的扩展，也可以看作是基于策略的方法在减少样本方差和提升采样效率方面的改进。虽然 Actor-Critic 方法吸收了上述两种方法的优点，但同时也继承了相应的缺点。比如，Critic 存在过估计的问题，Actor 存在探索不足的问题等。一些常见的 Actor-Critic 类的算法包括 Actor-Critic（AC）算法 (Sutton et al., 2018) 和一系列改进：（1）异步优势 Actor-Critic 算法（A3C）(Mnih et al., 2016) 将 Actor-Critic 方法扩展到异步并行学习，打乱数据之间的相关性，提高了样本收集速度和训练效率；（2）深度确定性策略梯度算法（Deep Deterministic Policy Gradient，DDPG）(Lillicrap et al., 2015) 沿用了深度 Q 网络算法的目标网络，同时 Actor 是一个确定性策略；（3）孪生延迟 DDPG 算法（Twin Delayed Deep Deterministic Policy Gradient，TD3）(Fujimoto et al., 2018) 引入了截断的（Clipped）Double Q-Learning 解决过估计问题，同时延迟 Actor 更新频率以优先提高 Critic 拟合准确度；（4）柔性 Actor-Critic 算法（Soft Actor-Critic，SAC）(Haarnoja et al., 2018) 在 Q 值函数估计中引入熵正则化，以提高智能体探索能力。

3.3　蒙特卡罗方法和时间差分方法

蒙特卡罗（Monte Carlo，MC）方法和时间差分（Temporal Difference，TD）方法的区别已经在第 2 章中讨论过，一些算法如图 3.4 所示。这里我们再次总结它们的特点以保证本章的完整性。时间差分方法是动态规划（Dynamic Programming，DP）方法和蒙特卡罗方法的一种中间形式。首先，时间差分方法和动态规划方法都使用自举法（Bootstrapping）进行估计，其次，时间差分方法和蒙特卡罗方法都不需要获取环境模型。这两种方法最大的不同之处在于如何进行参数更新，蒙特卡罗方法必须等到一条轨迹生成（真实值）后才能更新，而时间差分方法在每一步动作执行都可以通过自举法（估计值）及时更新。这种差异将使时间差分方法方法具有更大的偏差，而使蒙

特卡罗方法方法具有更大的方差。

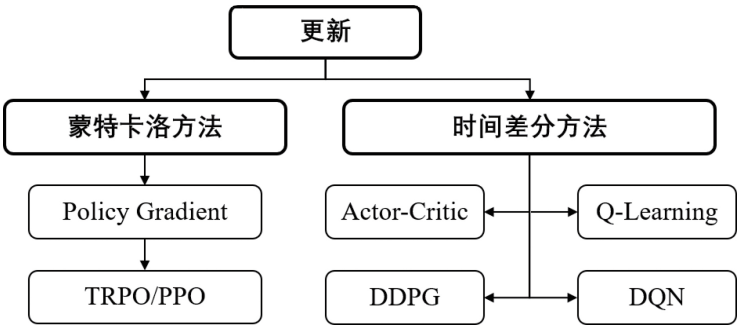

图 3.4 蒙特卡罗方法和时间差分方法

3.4 在线策略方法和离线策略方法

在线策略（On-Policy）方法和离线策略（Off-Policy）方法依据策略学习的方式对强化学习算法进行划分（图 3.5）。在线策略方法试图评估并提升和环境交互生成数据的策略，而离线策略方法评估和提升的策略与生成数据的策略是不同的。这表明在线策略方法要求智能体与环境交互的策略和要提升的策略必须是相同的。而离线策略方法不需要遵循这个约束，它可以利用其他智能体与环境交互得到的数据来提升自己的策略。常见的在线策略方法是 Sarsa，它根据当前策略选择一个动作并执行，然后使用环境反馈的数据更新当前策略。因此，Sarsa 与环境交互的策略和更新的策略是同一个策略。它的 Q 函数更新公式如下：

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_t + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]. \tag{3.1}$$

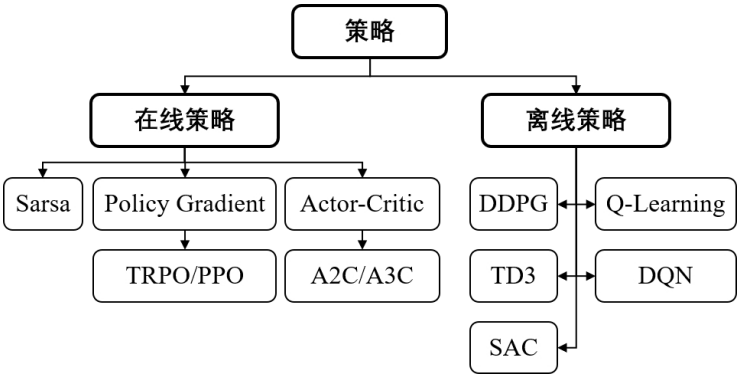

图 3.5 在线策略方法和离线策略方法

Q-learning 是一种典型的离线策略方法。它在选择动作时采用 max 操作和 ϵ-贪心策略，使得与环境交互的策略和更新的策略不是同一个策略。它的 Q 函数更新公式如下：

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_t + \gamma \max_a Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]. \tag{3.2}$$

参考文献

BALDI P, 2012. Autoencoders, Unsupervised Learning, and Deep Architectures[C]//Proceedings oast the International Conference on Machine Learning (ICML). 37-50.

BELLEMARE M G, DABNEY W, MUNOS R, 2017. A distributional perspective on reinforcement learning[C]//Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org: 449-458.

FORTUNATO M, AZAR M G, PIOT B, et al., 2017. Noisy networks for exploration[J]. arXiv preprint arXiv:1706.10295.

FUJIMOTO S, VAN HOOF H, MEGER D, 2018. Addressing function approximation error in actor-critic methods[J]. arXiv preprint arXiv:1802.09477.

HA D, SCHMIDHUBER J, 2018. Recurrent world models facilitate policy evolution[C]//Advances in Neural Information Processing Systems. 2450-2462.

HAARNOJA T, ZHOU A, ABBEEL P, et al., 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor[J]. arXiv preprint arXiv:1801.01290.

HEESS N, SRIRAM S, LEMMON J, et al., 2017. Emergence of locomotion behaviours in rich environments[J]. arXiv:1707.02286.

KALASHNIKOV D, IRPAN A, PASTOR P, et al., 2018. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation[J]. arXiv preprint arXiv:1806.10293.

LI Y, 2017. Deep reinforcement learning: An overview[J]. arXiv preprint arXiv:1701.07274.

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

MNIH V, KAVUKCUOGLU K, SILVER D, et al., 2015. Human-level control through deep reinforcement learning[J]. Nature.

MNIH V, BADIA A P, MIRZA M, et al., 2016. Asynchronous methods for deep reinforcement learning[C]//International Conference on Machine Learning (ICML). 1928-1937.

MUNOS R, STEPLETON T, HARUTYUNYAN A, et al., 2016. Safe and efficient off-policy reinforcement learning[C]//Advances in Neural Information Processing Systems. 1054-1062.

NAGABANDI A, KAHN G, FEARING R S, et al., 2018. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning[C]//2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE: 7559-7566.

RACANIÈRE S, WEBER T, REICHERT D, et al., 2017. Imagination-augmented agents for deep reinforcement learning[C]//Advances in Neural Information Processing Systems. 5690-5701.

SCHAUL T, QUAN J, ANTONOGLOU I, et al., 2015. Prioritized experience replay[C]//arXiv preprint arXiv:1511.05952.

SCHRITTWIESER J, ANTONOGLOU I, HUBERT T, et al., 2019. Mastering atari, go, chess and shogi by planning with a learned model[Z].

SCHULMAN J, LEVINE S, ABBEEL P, et al., 2015. Trust region policy optimization[C]//International Conference on Machine Learning (ICML). 1889-1897.

SCHULMAN J, WOLSKI F, DHARIWAL P, et al., 2017. Proximal policy optimization algorithms[J]. arXiv:1707.06347.

SILVER D, HUANG A, MADDISON C J, et al., 2016. Mastering the game of go with deep neural networks and tree search[J]. Nature.

SILVER D, HUBERT T, SCHRITTWIESER J, et al., 2017. Mastering chess and shogi by self-play with a general reinforcement learning algorithm[J]. arXiv preprint arXiv:1712.01815.

SILVER D, HUBERT T, SCHRITTWIESER J, et al., 2018. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play[J]. Science, 362(6419): 1140-1144.

SUTTON R S, BARTO A G, 2018. Reinforcement learning: An introduction[M]. MIT press.

SUTTON R S, MCALLESTER D A, SINGH S P, et al., 2000. Policy gradient methods for reinforcement learning with function approximation[C]//Advances in Neural Information Processing Systems. 1057-1063.

VAN HASSELT H, GUEZ A, SILVER D, 2016. Deep reinforcement learning with double Q-learning[C]// Thirtieth AAAI conference on artificial intelligence.

WANG Z, SCHAUL T, HESSEL M, et al., 2016. Dueling network architectures for deep reinforcement learning[C]//International Conference on Machine Learning. 1995-2003.

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