10 分层强化学习

在本章中，我们将介绍分层强化学习。它是一种通过构建并利用认知和决策过程的底层结构来提高学习效果的方法。具体来说，首先我们将介绍了分层强化学习的背景和两个主要类别：选项框架（Options Framework）和封建制强化学习（Feudal Reinforcement Learning）。然后我们将详细介绍这些类别中的一些典型算法，包括战略专注作家（Strategic Attentive Writer）、选项批判者（Option-critic）和封建制网络（Feudal Networks）等。在本章的最后，我们对近年来关于分层强化学习的研究成果进行了总结。

10.1 简介

近年来，深度强化学习在许多领域取得了显著的成功 (Levine et al., 2018; Mnih et al., 2015; Schulman et al., 2015; Silver et al., 2016, 2017)。然而，长期规划对智能体来说仍然是一个挑战，特别是在一些奖励稀疏、大时间跨度的环境，例如 *Dota* (OpenAI, 2018) 和《星际争霸》 (Vinyals et al., 2019)。分层强化学习（Hierarchical Reinforcement Learning，HRL）提供了一种方法来寻找这种复杂控制问题中的时空抽象和行为模式 (Bacon et al., 2017; Barto et al., 2003; Dayan, 1993b; Dayan et al., 1993a; Dietterich, 1998, 2000; Hausknecht, 2000; Kaelbling, 1993; Nachum et al., 2018; Parr et al., 1998a; Sutton et al., 1999; Vezhnevets et al., 2016, 2017)。与人类认知的层次结构类似，HRL 具备抽象多层次控制的潜力，其中高层次的长期规划和元学习指导低层次的控制器。层次结构的模块化也提供了可移植性和可解释性，例如，理解地图和达到有利状态的技术通常在像 *grid-world* (Tamar et al., 2016) 或者 *Doom* (Bhatti et al., 2016; Kempka et al., 2016) 这样的游戏中十分有用。

以往对 HRL 的研究大多从 4 个主要方面展开：**选项框架**（Options Framework）(Sutton et al., 1999)、**封建制强化学习**（Feudal Reinforcement Learning，FRL）(Dayan et al., 1993a)、**MAXQ 分解**

（MAXQ Decomposition）(Dietterich, 2000) 和 **层次抽象机**（Hierarchical Abstract Machines，HAMs）(Parr et al., 1998a,b) 在选项框架中，高层策略会在特定的时间步上切换低层策略，以便在时间域上分解问题。在 FRL 智能体中，高层控制器负责为下层控制器提出明确的目标（如某些特定的状态），来实现状态空间的层次分解。MAXQ 分解也提出了一种将子任务的解与 Q 值函数相结合的状态抽象方法。HAMs 则考虑了一个学习过程来减少大型复杂问题中的搜索空间，其学习过程中智能体能执行的动作受限于有限状态机的层次。在本章中，我们将重点介绍在 HRL 中应用深度学习的最新研究成果。具体来说，我们讨论了分别属于选项框架和 FRL 的两种算法，并在本章结尾对深度 HRL 进行了简要的总结。

10.2 选项框架

选项框架 (Hausknecht, 2000; Sutton et al., 1999) 将动作在时间层面扩展。**选项**（Options），也被称为技能 (Da Silva et al., 2012) 或者宏操作 (Hauskrecht et al., 1998; Vezhnevets et al., 2016)，是一种具有终止条件的子策略。它观察环境并输出动作，直到满足终止条件为止。终止条件是一类时序上的隐式分割点，来表示相应的子策略已经完成了自己的工作，且顶层的选项策略（Policy-Over-Action）需要切换至另一个选项。给定一个状态集为 $\mathcal{S}$、动作集为 $\mathcal{A}$ 的 MDP，选项 $\omega \in \Omega$ 被定义为三元组 $(I_\omega, \pi_\omega, \beta_\omega)$，其中 $I_\omega \subseteq \mathcal{S}$ 为一组初始状态集，$\pi_\omega : \mathcal{S} \times \mathcal{A} \to [0,1]$ 是一个选项内置策略，而 $\beta_\omega : \mathcal{S} \to [0,1]$ 是一个通过伯努利分布提供随机终止条件的终止函数。一个选项 ω 只有在 $s \in I_\omega$ 时，才能用于状态 s。一个智能体通过其选项策略选择一个选项，并继续保持该策略直到终止条件满足，然后再次查询选项策略并重复该步骤。注意，若选项 ω 被执行，则动作将由相应的策略 π_ω 进行选择，直到选项根据 β_ω 被随机终止。比如说，一个名为“开门”的选项可能包含一个用于靠近、抓取和转动门把手的策略，以及一个确定门被打开概率的终止条件。

尤其特别的是，一个选项框架由两层结构组成：底层的每个元素是一个选项，而顶层则是一个选项策略，用来在片段开始或上个选项终结时候选择一个选项。选项策略从环境给出的奖励信息学习，而选项可通过明确的子目标来学习。例如，在表格情况下，每个状态可以被看作子目标的候选 (Schaul et al., 2015; Wiering et al., 1997)。一旦给出了选项，则顶层可以将其作为动作，通过标准技术来进行学习。

在选项框架中，顶层模块学习的是一个选项策略，而底层模块学习能完成各个选项目标的策略。这可以看成马尔可夫过程在时间层（几个时间步）上的分解。**半马尔可夫决策过程**（Semi-Markov Decision Process，SMDP）为动作间持续时间具备不确定性的选项框架提供了一个理论观点 (Sutton et al., 1999)，如图 10.1 所示。SMDP 是一个具备额外元素 $\mathcal{F}$: $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \mathcal{F})$ 的标准 MDP。其中 $\mathcal{F}(t|s,a)$ 给出在状态 s 下执行动作 a 时，转移时间为 t 的概率。不严谨地说，选项框架中的顶层控制可以被看成一个 SMDP 上的策略。对于多级选项的情况，更高层的选项可以看成低层选项在时间上进一步扩展的 SMDP (Riemer et al., 2018)。

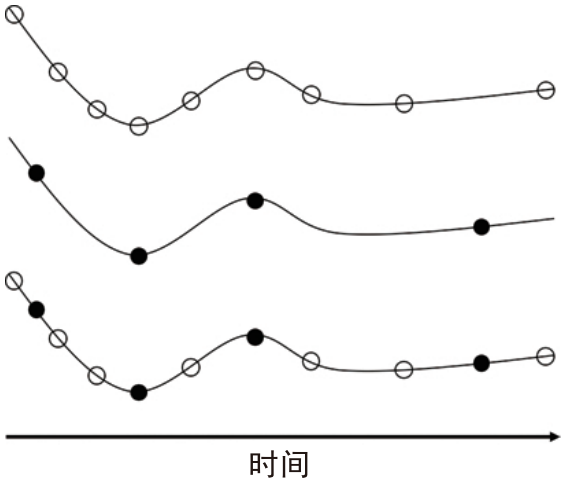

图 10.1　在 SMDP 视角下的选项，改编自文献 (Sutton et al., 1999). 顶部：一个马尔可夫决策过程（MDP）的状态轨迹。中部：一个半马尔可夫决策过程（SMDP）的状态轨迹。底部：一个两层结构上 MDP 的状态轨迹。实心圆表示 SMDP 的决策，而空心圆则是相应选项包含的原始动作

研究表明，人工定义的选项通过和深度学习的结合，即使在像《我的世界》和雅达利游戏这样很有挑战性的环境中，也可以取得显著的效果 (Kulkarni et al., 2016; Tessler et al., 2017)。然而，初始集和终结条件是选项框架的一个制约因素。例如，一个人工定义的策略 π_ω 是让移动机器人插上它的充电器，而它很有可能是只为充电器在视野范围内的状态而定制的。终结条件表明当机器人成功插上充电器或者状态在 I_ω 之外时，终结的概率为 1。因此，如何自动地发掘选项也曾是 HRL 的一个研究主题。我们将介绍两种算法，它们将选项发掘表示为优化问题，并用函数逼近的方式解决这类问题。第一个是一种深度递归神经网络，被称为战略专注作家（Strategic Attentive Writer，STRAW），它通过开环选项内置策略[1]（Open-Loop Intra-Option Policies）学习选项。第二个则是考虑闭环选项内置策略[2]（Close-Loop Intra-Option Policies）的选项-批判者（Option-Critic）结构。

10.2.1　战略专注作家

战略专注作家 (Vezhnevets et al., 2016) 是一种新奇的深度递归神经网络结构。它对常见的动作序列（宏动作）进行时域抽象，并通过这些动作进行端到端的学习。值得注意的是，宏动作是一个在神经网络中隐式表示的特定选项。其动作序列（或者在此之上的分布）是在宏动作被初始化的时候决定的。STRAW 分别包含短期动作分布和长期计划这两个模块。

第一个模块将环境的观测数据转化为一个**动作-计划**（Action-Plan），它是一个显式（Explicit）

[1]开环意即不将控制的结果反馈，进而影响当前控制的系统。

[2]闭环意即将控制的结果进行完全反馈，进而影响当前控制的系统。

的随机变量，用于表示接下来一段时间内计划执行的动作。当时间步为 t 时，动作-计划表示为矩阵 $\boldsymbol{A} \in \mathbb{R}^{|\mathcal{A}| \times T}$，其中 T 是计划的最大时间跨度，而在 $\boldsymbol{A}$ 中的第 τ 列对应动作在时间步 $t+\tau$ 的分对数。

第二个模块通过单行矩阵 $\boldsymbol{c}^t \in \mathbb{R}^{1 \times T}$ 维护**承诺-计划**（Commitment-Plan），即一个决定在哪一步网络结束一个宏动作并更新动作-计划的状态变量。在时间步为 t 时，$\boldsymbol{c}^{t-1}$ 的第一个元素提供了终止条件的伯努利分布的参数。在落实计划的期间，行动-计划 $\boldsymbol{A}^t$ 和承诺-计划 $\boldsymbol{c}^t$ 都被一个时间移位运算符 ρ 直接滑动至下一步，其中 ρ 通过移除矩阵的第一列并在末尾添 0 的形式来移动矩阵。

图 10.2 显示了一个包含动作-计划和承诺-计划的 STRAW 工作流示例。为了更新这两类计划，STRAW 在时间维度上使用了专注写作技术 (Gregor et al., 2015)，它使网络能够聚焦在当前部分。该技术将一个高斯滤波器矩阵沿着时间维度应用于计划。更准确地说，对于时间大小 K，一个 $|\mathcal{A}| \times K$ 一维高斯滤波器的网格通过指定网格中心的坐标和相邻过滤器之间的步幅来在计划中定位。注意，这里的步幅（Stride）和 CNN 中的相同术语相似。让 ψ^A 作为动作-计划的注意力参数，即高斯滤波器的网格位置、步幅和标准差。STRAW 将注意力操作定义如下：

$$\boldsymbol{D} = \text{write}(\boldsymbol{p}, \psi_t^A); \boldsymbol{\beta}_t = \text{read}(\boldsymbol{A}^t, \psi_t^A), \tag{10.1}$$

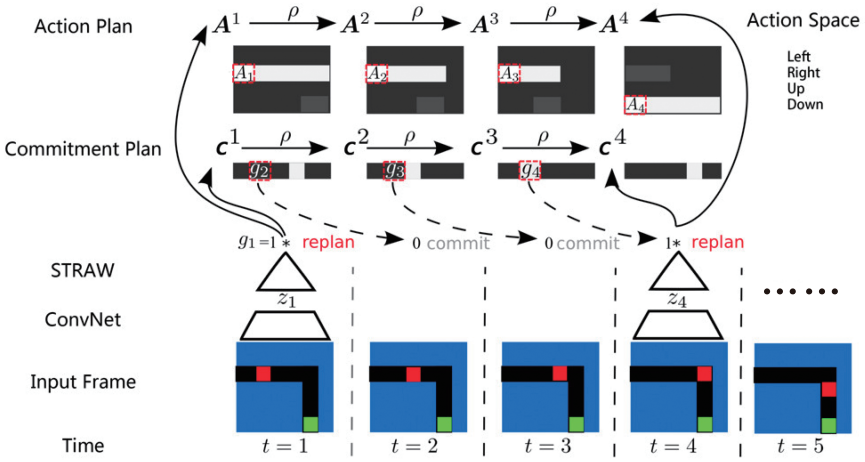

图 10.2 STRAW 在一个迷宫导航游戏中的工作流程，改编自文献 (Vezhnevets et al., 2016)。观测数据是原始像素，其中像素的颜色可以是蓝色、黑色、红色和绿色，分别代表墙、走廊、智能体和最终目的地。动作空间为上、下、左、右四个方向的移动。当 $t = 1$ 时，帧的特征被一个卷积神经网络提取后输入进 STRAW。STRAW 立刻产生两个计划。在紧接着的 2 个时间步中，这两个计划被 ρ 滑动。之后，智能体来到角落并由承诺-计划 $\boldsymbol{c}^t$ 给出一个重新计划的信号（见彩插）

其中，$\boldsymbol{p} \in \mathbb{R}^{A \times K}$ 是一个时间窗口为 K 的计划补丁。write 操作生成了一个与 $\boldsymbol{A}^t$ 相同大小的平滑计划 $\boldsymbol{D}$，而 read 操作生成了一个读取补丁 $\boldsymbol{\beta}_t \in \mathbb{R}^{A \times K}$。此外，将 z_t 作为在时间步 t 下的观测数据的特征表示，并将相似的注意力技术应用于承诺-计划，计划的更新算法如算法 10.34 所示。其中 f^{ψ}、f^A 和 f^c 都是线性函数，h 是一个多层感知器，$\boldsymbol{b} \in \mathbb{R}^{1 \times T}$ 是一个具有相同标量参数 b 的偏差，而 e 是 固定为 40 的标量 (Vezhnevets et al., 2016)，以便经常重新做计划。

算法 10.34 STRAW 中的计划更新

if $g_t = 1$ **then**
　计算动作-计划的注意力参数 $\psi_t^A = f^{\psi}(z_t)$
　应用专注阅读：$\boldsymbol{\beta}_t = \text{read}(\boldsymbol{A}^{t-1}, \psi_t^A)$
　计算中间表示 $\epsilon_t = h(\text{concat}(\beta_t, z_t))$
　计算承诺-计划的注意力参数 $\psi_t^c = f^c(\text{concat}(\psi_t^A, \epsilon_t))$
　更新 $\boldsymbol{A}^t = \rho(\boldsymbol{A}^{t-1}) + \text{write}(f^A(\epsilon_t), \psi_t^A)$
　更新 $\boldsymbol{c}_t = \text{Sigmoid}(\boldsymbol{b} + \text{write}(e, \psi_t^c))$
else
　更新 $\boldsymbol{A}^t = \rho(\boldsymbol{A}^{t-1})$
　更新 $\boldsymbol{c}_t = \rho(\boldsymbol{c}_{t-1})$
end if

对于进一步的结构化搜索，STRAW 在对角高斯分布上使用了重参数技术 $Q(z_t|\zeta_t) = \mathcal{N}(\mu(\zeta_t), \sigma(\zeta_t))$，其中 ζ_t 是特征提取器的输出。STRAW 的训练 loss 被定义如下：

$$\mathcal{L} = \sum_{t=1}^{\mathrm{T}} (L(\boldsymbol{A}^t) + \alpha g_t \text{KL}(Q(z_t|\zeta_t)|P(z_t)) + \lambda \boldsymbol{c}_1^t), \tag{10.2}$$

其中，L 是一个领域特定损失函数（例如回报的负对数似然），$P(z_t)$ 是一个先决条件，而最后一项惩罚了重新计划并鼓励承诺。

要特别注意的是，STRAW 是一个网络结构。对于强化学习的任务，可以使用一系列的强化学习算法。文献 (Vezhnevets et al., 2016) 展示了在《2D 迷宫》和雅达利游戏上使用 A3C (Mnih et al., 2016) 算法的效果。《2D 迷宫》是由许多格子组成的一个 2D 网格世界，其中每个格子只可能是墙壁或者通道，而其中，某个通道会随机选择为目的地。智能体将完全观测到迷宫的状态，并需要通过结构化探索来到达目标。在本任务中，文献 (Vezhnevets et al., 2016) 展示了 STRAW 在策略上的表现优于 LSTM，并且很接近由 Dijkstra 算法给出的最优策略。在雅达利游戏领域中，文献 (Vezhnevets et al., 2016) 选出了 8 个需要一些规划和探索的游戏，其中 STRAW 及其变体在 8 个游戏中的 6 个里，比 LSTM 和简单前馈网络在游戏中获得了更高的分数。

10.2.2 选项-批判者结构

选项-批判者结构（Option-Critic Architecture）(Bacon et al., 2017) 将策略梯度定理扩展至选项，它提供一种端到端的对选项和选项策略的联合学习。它直接优化了折扣化回报。我们先考虑将选项-价值函数定义如下：

$$Q_\Omega(s,\omega) = \sum_a \pi_\omega(a|s) Q_U(s,\omega,a), \tag{10.3}$$

其中 $Q_U : \mathcal{S} \times \Omega \times \mathcal{A} \to \mathbb{R}$ 是在确定状态-选项对 (s,ω) 后执行某个动作的价值：

$$Q_U(s,\omega,a) = R(s,a) + \gamma \sum_{s'} p(s'|s,a) U(\omega,s'), \tag{10.4}$$

其中 $U : \Omega \times \mathcal{S} \to \mathbb{R}$ 是进入一个状态 s' 时，执行 ω 的价值：

$$U(\omega,s') = (1-\beta_\omega(s'))Q_\Omega(s',\omega) + \beta_\omega(s')V_\Omega(s'), \tag{10.5}$$

其中 $V_\Omega : \mathcal{S} \to \mathbb{R}$ 是选项的最优价值函数：

$$V_\Omega(s') = \max_{\omega\in\Omega} \mathbb{E}_\omega[\sum_{n=0}^{k-1} \gamma^n R_{t+n} + \gamma^k V_\Omega(S_{t+k}) | S_t = s'], \tag{10.6}$$

其中 k 是 ω 在状态 s' 中的预计持续时间。因此，我们可以定义 $A_\Omega : \mathcal{S} \times \Omega \to \mathbb{R}$ 为选项的优势函数：

$$A_\Omega(s,\omega) = Q_\Omega(s,\omega) - V_\Omega(s). \tag{10.7}$$

如果选项 ω_t 曾被初始化或者已经在状态 S_t 中执行了 t 个时间步，通过将状态-选项对视为马尔可夫链中的常规状态，那么在一步中状态转移至 (S_{t+1},ω_{t+1}) 的概率为

$$\sum_a \pi_{\omega_t}(a|S_t) p(S_{t+1}|S_t,a)[(1-\beta_{\omega_t}(S_{t+1}))\mathbf{1}_{\omega_t=\omega_{t+1}} + \beta_{\omega_t}(S_{t+1})\pi_\Omega(\omega_{t+1}|S_{t+1})]. \tag{10.8}$$

通过假设所有选项在任何地方都可用，上述转移是一个在状态-选项对的唯一稳态。

用于学习选项的随机梯度下降算法的结构如图 10.3 所示，其中梯度由定理 10.1 和定理 10.2 给出。然而，文献 (Bacon et al., 2017) 提出了通过一种基于两种时间尺度结构来学习价值，在更新选项内置策略使用更快的时间尺度，而更新终止函数时使用比例更小的时限 (Konda et al., 2000)。我们可以从行动者-批判者结构中看出，选项内置策略、终止函数和选项策略都属于行动者的部

分，而批判者则包括 Q_U 和 A_Ω。

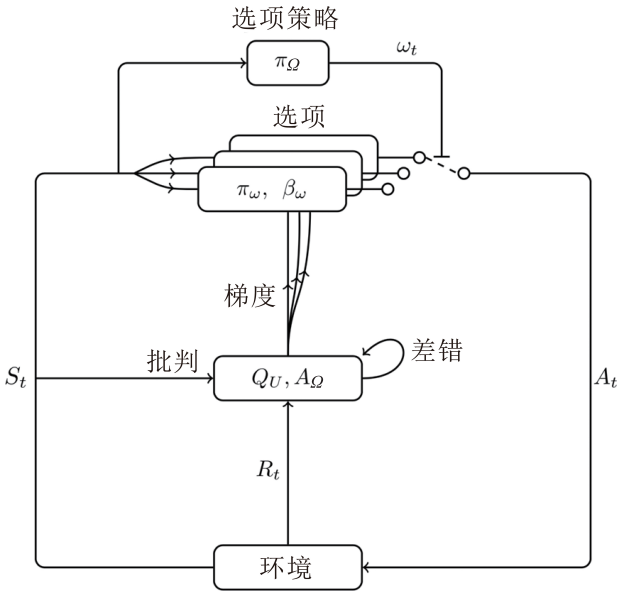

图 10.3　选项-评判家结构，改编自文献 (Bacon et al., 2017)

定理 10.1 选项内置策略梯度理论（Intra-Option Policy Gradient Theorem）(Bacon et al., 2017) 给定一组马尔可夫选项，随机选项内置策略对它们的参数 θ 可微。折扣化回报期望关于 θ 和初始条件 $(\hat{s},\hat{\omega})$ 的梯度为

$$\sum_{s,\omega}\mu_\Omega(s,\omega|\hat{s},\hat{\omega})\sum_a\frac{\partial\pi_{\omega,\theta}(a|s)}{\partial\theta}Q_U(s,\omega,a), \tag{10.9}$$

其中 $\mu_\Omega(s,\omega|\hat{s},\hat{\omega})=\sum_{t=0}^{\infty}\gamma^t p(S_t=s,\omega_t=\omega|S_0=\hat{s},\omega_0=\hat{\omega})$ 是一个沿着从 $(\hat{s},\hat{\omega})$ 开始的轨迹的状态-选项对的折扣化权重。

定理 10.2 终止梯度定理（Termination Gradient Theorem）(Bacon et al., 2017) 给定一组马尔可夫选项，选项的随机终止函数对其参数 φ 可微。折扣化回报目标期望对于 φ 和初始条件 $(\hat{s},\hat{\omega})$ 的梯度为

$$-\sum_{s',\omega}\mu_\Omega(s',\omega|\hat{s},\hat{\omega})\frac{\partial\beta_{\omega,\varphi}(s')}{\partial\varphi}A_\Omega(s',\omega), \tag{10.10}$$

其中 $\mu_\Omega(s,\omega|\hat{s},\hat{\omega})=\sum_{t=0}^{\infty}\gamma^t p(S_t=s,\omega_t=\omega|S_0=\hat{s},\omega_0=\hat{\omega})$ 是一个沿着从 $(\hat{s},\hat{\omega})$ 开始的轨迹的状态-选项对的折扣化权重。

文献 (Bacon et al., 2017) 提供了离散和连续环境下的实验。在离散环境中，文献 (Bacon et al., 2017) 在雅达利学习环境（Arcade Learning Environment，ALE）(Bellemare et al., 2013) 中训练了 4 个雅达利游戏，这些训练与文献 (Mnih et al., 2015) 采取了相同的设置。结果表明，选项-批判者能够在这全部 4 个游戏中学到结构选项。在连续环境中，文献 (Bacon et al., 2017) 选择了 Pinball 游戏 (Konidaris et al., 2009)，游戏中智能体控制一个小球在随机形状的多边形 2D 迷宫中进行移动，其目的地也随机生成。通过选项-批判者学习到的轨迹表明，智能体可以实现时域抽象。

10.3 封建制强化学习

封建制强化学习（Feudal Reinforcement Learning，FRL）(Dayan et al., 1993a) 提出了一种封建制等级结构。其中，管理者有着为他们工作的下级管理者和他们自己的上级管理者。它反映了封建等级制度，其中每层的各个管理者可以为他们的下级设置任务、奖励和惩罚。有两个保证封建制规则的关键原则需要被重视：**奖励隐藏**（Reward Hiding）和 **信息隐藏**（Information Hiding）。奖励隐藏指的是，无论某管理者做出的指令是否能使其上级满意，该管理者的下级都必须服从。而信息隐藏是指管理者的下级不知道该管理者被派予的任务，而管理者的上级也不知道该管理者给其下级安排了什么任务。顶层的封建智能体并非像选项框架那样学习一个选项的时间分解，而是通过为底层策略制定明确目标来分解状态空间的问题。这样的结构允许强化学习扩展到管理层之间具有明确分工到大型领域中。

在这种解耦学习的启发下，文献 (Vezhnevets et al., 2017) 引入了一种新的神经网络结构，称为 **封建制网络**（Feudal Networks，FuNs）。它可以自动发现子目标，并且具备奖励隐藏和信息隐藏的软条件。它跨越多层解耦了端到端学习，这使得它可以处理不同的时间分辨率。此外，**使用离线策略修正的分层强化学习**（Hierarchical Reinforcement Learning with Off-policy Correction，HIRO）可进一步提高了离线策略经验的样本效率 (Nachum et al., 2018)。实验显示，HIRO 取得了显著的进展，并且能解决非常复杂的结合运动和基本物体交互的问题。

10.3.1 封建制网络

封建制网络（Feudal Networks，FuNs）是一个完全可微模块化的 FRL 神经网络，它有两个模块：管理者和工作者。管理者在一个潜在状态空间中更低的时间分辨率上设定目标，而工作者则学习如何通过内在奖励达到目标。图 10.4 展示了 FuN 的结构，其中前向过程可以描述成以下等式：

$$\boldsymbol{z}_t = f^{\text{Percept}}(S_t) \tag{10.11}$$

$$\boldsymbol{m}_t = f^{\text{Mspace}}(\boldsymbol{z}_t) \tag{10.12}$$

$$h_t^{\text{M}}, \hat{\boldsymbol{g}}_t = f^{\text{Mrnn}}(\boldsymbol{m}_t, \boldsymbol{h}_{t-1}^{M}); \boldsymbol{g}_t = \hat{\boldsymbol{g}}_t / \|\hat{\boldsymbol{g}}_t\|; \tag{10.13}$$

$$\boldsymbol{w}_r = \phi\left(\sum_{i=t-c}^{t} \boldsymbol{g}_i\right) \tag{10.14}$$

$$h^{\mathrm{W}}, \boldsymbol{U}_t = f^{\mathrm{Wrnn}}(\boldsymbol{z}_t, h_{t-1}^{W}) \tag{10.15}$$

$$\boldsymbol{\pi}_t = \mathrm{SoftMax}(\boldsymbol{U}_t \boldsymbol{w}_t) \tag{10.16}$$

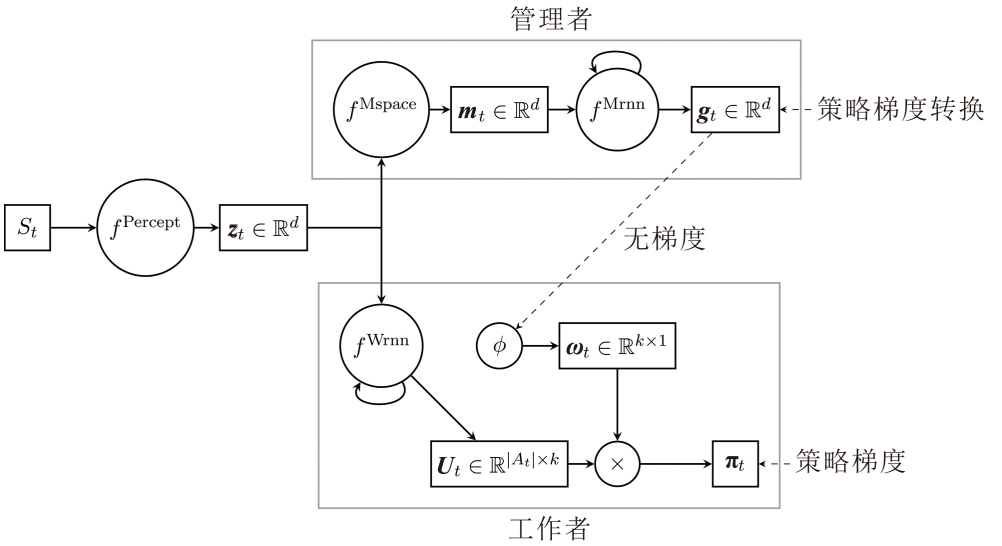

图 10.4　FuN 的结构，改编自文献 (Vezhnevets et al., 2017)。在文献 (Vezhnevets et al., 2017) 中，超参数 k 和 d 被定为 $k = 16 \ll d = 256$

其中 $\boldsymbol{z}_t$ 是 S_t 的表示，f^{Mspace} 向管理者提供状态 $\boldsymbol{m}_t$，而 $\boldsymbol{g}_t$ 表示管理者输出的目标。在 FRL 中需要注意以下两个原则：管理者和工作者之间没有梯度传播；但接收观测数据的感知机模块 f^{Percept} 共享。管理者的 f^{Mrnn} 和工作者的 f^{Wrnn} 都是循环模块，f^{Mspace} 是全连接的。h^{M} 和 h^{W} 分别对应管理者和工作者各自的内部状态。ϕ 是一个无偏线性变换，将目标 $\boldsymbol{g}_t$ 映射成一个嵌入向量 $\boldsymbol{w}_t$。$\boldsymbol{U}_t$ 表示动作的嵌入矩阵，它通过矩阵与 $\boldsymbol{w}_t$ 的积输出工作者动作策略的分对数。

考虑到标准强化学习的设置是最大化折扣化回报 $G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k}$。一个自然而然的学习整个结构的方法就是通过策略梯度算法进行端到端训练，因为 FuNs 全部可微。然而这样会导致梯度会被工作者通过任务目标传播给管理者，这可能导致目标会变成一个内部潜在变量，而不是分层标志。因此，FuN 分别训练管理者和工作者。对于管理者，更新规则遵循预测优势方向：

$$\nabla \boldsymbol{g}_t = (G_t - V_t^{\mathrm{M}}(S_t, \theta)) \nabla_\theta d_{\cos}(\boldsymbol{m}_{t+c} - \boldsymbol{m}_t, g_t(\theta)) \tag{10.17}$$

其中，V_t^{M} 是管理者的值函数，而 $d_{\cos}(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{\beta}/(|\boldsymbol{\alpha}||\boldsymbol{\beta}|)$ 是余弦相似度。另一方面，工作者可以通过任意现成的深度强化学习方式训练，其内在奖励定义如下：

$$R_t^{\mathrm{I}} = \frac{1}{c}\sum_{i=1}^{c} d_{\cos}(\boldsymbol{m}_t - \boldsymbol{m}_{t-i}, \boldsymbol{g}_{t-i}) \tag{10.18}$$

其中，状态空间中的方向偏移为目标提供了结构不变性。在实践中，FuN 通过使用 $R_t + \alpha R_t^{\mathrm{I}}$ 训练工作者，软化了原始 FRL 中的奖励隐藏条件，其中 α 是一个正则化内在奖励影响的超参数。

文献 (Vezhnevets et al., 2017) 也提供了一个关于管理者训练规则的理论分析。考虑到有高层跨策略的策略 $o(S_t, \theta)$，它在固定时长 c 下，在几个子策略中进行选择。对每个子策略来说，转移分布 $p(S_{t+c}|S_t, o)$ 可以被看作一个转移策略 $\pi^{\mathrm{T}}(S_{t+c}|S_t, \theta)$。和选项框架的 SMDP 视角类似，我们可以在高层 MDP 对 $\pi^{\mathrm{T}}(S_{t+c}|S_t, \theta)$ 应用策略梯度理论。

$$\nabla_\theta \pi^{\mathrm{T}}(S_{t+c}|S_t, \theta) = \mathbb{E}[(G_t - V_t^{\mathrm{M}}(S_t, \theta))\nabla_\theta \log p(S_{t+c}|S_t, o)] \tag{10.19}$$

这也被称为**转移策略梯度**（Transition Policy Gradients）。假设方向 $S_{t+c} - S_t$ 遵循 Mises-Fisher 分布，我们可以得到 $\log p(S_{t+c}|S_t, o) \propto d_{\cos}(S_{t+c} - S_t, \boldsymbol{g}_t)$。

此外，文献 (Vezhnevets et al., 2017) 提出了用于管理者的 Dilated LSTM，与空洞卷积一样，可以在分辨率无损的情况下获取更大的感受野。Dilated LSTM 维持了几个内部 LSTM 单元的状态。在任意时间步中，只有一个单元状态被更新，而输出的是最近 c 个被更新的状态进行池化后的结果。

需要注意的是，与 STRAW 相类似，FuN 也是一个用于 HRL 的神经网络结构。文献 (Vezhnevets et al., 2017) 选择了 A3C 作为强化学习算法，并设计了一系列的实验来显示 FuN 相对于 LSTM 的有效性。首先，它展示了对 FuN 应用在 *Montezuma's Revenge* 游戏的分析。*Montezuma's Revenge* 是一个雅达利游戏，它在强化学习领域是个难题。它需要通过许多技巧来控制角色躲开致命的陷阱，并且从稀疏奖励中进行学习。实验结果显示，FuN 在采样效率上有着显著的提高。此外，它在另外 10 款雅达利游戏中也有效果提升，其中 FuN 的分数明显高于选项-批判者结构。同样，文献 (Vezhnevets et al., 2017) 使用了 4 个不同等级的 DeepMind 实验室 3D 游戏平台 (Beattie et al., 2016) 来验证 FuN。它证明 FuN 学习了更加有意义的子策略，之后将这些子策略在内存中高效地结合起来能产生更有价值的行为。

10.3.2 离线策略修正

HRL 方法提出训练多层策略来对时间和行为进行抽象。在前几节中，我们讨论了 STRAW 和 FuN 使用神经网络结构来学习一个分层策略，而选项-批判者结构则端到端地同时学习内部策略和选项的终止条件。HRL 还存在许多问题，例如通用性、可迁移性和采样效率等。在本节中，我们

将介绍离线策略修正分层强化学习（Hierarchical Reinforcement Learning with Off-policy Correction，HIRO）(Nachum et al., 2018)。它为训练 HRL 智能体提供了一种普遍适用且数据效率很高的方法。

一般来说，HIRO 考虑了高层控制器通过自动提出一些目标来监督低层控制器的方案。更准确地说，在每个时间步 t 中，HIRO 通过一个目标 g_t 来驱动智能体。给定一个用户指定的参数 c，若 t 是 c 的倍数，则目标 g_t 由高层策略 μ^h 产生，否则 g_t 由目标转移函数 h: $g_t = h(S_{t-1}, g_{t-1}, S_t)$ 通过之前的目标 g_{t-1} 提供。和 FuN 类似，目标是指包含所需位置和方向信息在内的高层决策。实验发现，与在嵌入空间中表示目标不同，HIRO 直接使用原始观测数据更为有效。需要注意的是，我们可以根据特定任务的领域知识设计内在奖励和目标转移函数。具体来说，在最简单的情况下，内在奖励被定义如下：

$$R_t^{\mathrm{I}} = -\|S_t + g_t - S_{t+1}\|_2, \tag{10.20}$$

目标转移函数被定义为

$$h(S_{t-1}, g_{t-1}, S_t) = S_{t-1} + g_{t-1} - S_t \tag{10.21}$$

来维持目标方向。

为了提高数据效率，HIRO 将离线策略技术扩展到高层和低层训练。HIRO 让低层策略 μ^l 存储经验 $(S_t, g_t, A_t, R_t^{\mathrm{I}}, S_{t+1}, h(S_t, g_t, S_{t+1}))$，并将 g_t 视为模型的额外输入，以支持任意离线算法训练这些策略。对于高层策略，转移元组 $(S_{t:t+c}, g_{t:t+c}, A_{t:t+c}, R_{t:t+c}, S_{t+c})$（':' 在 Python 中表示切片操作。这里的切片不包括最后的元素）也可以通过任意的离线策略算法进行训练，这里只需将 g_t 视为一个动作并累加 $R_{t:t+c}$ 作为奖励。然而过去的低层控制器观测的转移数据并不能反映动作。为了解决这个问题，HIRO 提出使用重标记（Re-label）技术来纠正高层转移数据。旧的转移数据 $(S_t, g_t, \sum R_{t:t+c}, S_{t+c})$ 将被重新标记一个不同的目标 $\hat{g}_t$ 使得 $\hat{g}_t$ 能最大化 $\mu^l(A_{t:t+c}|S_{t:t+c}, \hat{g}_{t:t+c})$ 概率，其中 $\hat{g}_{t+1:t+c}$ 通过目标转移函数 h 计算。对于随机行为策略，其对数概率 $\log \mu^l(A_{t:t+c}|S_{t:t+c}, \hat{g}_{t:t+c})$ 可以通如下方式计算：

$$\log \mu^l(A_{t:t+c}|S_{t:t+c}, \hat{g}_{t:t+c}) \propto -\frac{1}{2} \sum_{i=t}^{t+c-1} \|A_t - \mu^l(S_i, \hat{g}_i)\|_2^2 + \text{const.} \tag{10.22}$$

在实践中，HIRO 从一个包括原始目标的目标候选集中选择能最大化对数概率的目标。该目标对应 $S_{t+c} - S_t$ 的差，并来自一个对角高斯分布的采样。分布中每个平均项随机对应向量 $S_{t+c} - S_t$ 中的元素，其中减号表示一个元素运算符。

HIRO 的结构如图 10.5 所示。Nachum 等人 (Nachum et al., 2018) 在文献 (Duan et al., 2016) 中通过 4 个挑战性的任务验证了 HIRO。实验表明，离线策略修正具有显著的优势，并且对低层控制器的重标记可以对初始训练进行加速。

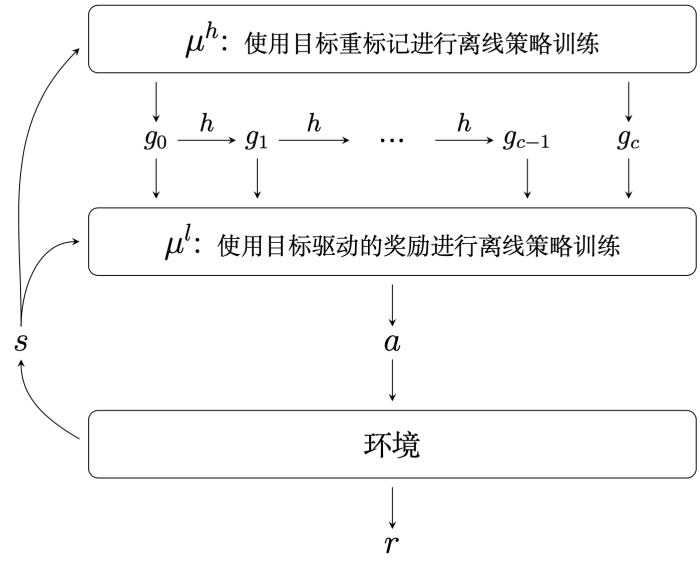

图 10.5 HIRO 的结构，改编自文献 (Nachum et al., 2018)。低层策略接收高层目标，并直接与环境交互。其中目标是由高层策略或者目标转移函数产生的

10.4 其他工作

在本节中，我们对近年来 HRL 方面的工作进行了简要的总结。图 10.6 显示了两个视角。先从低层策略奖励信号这个视角看，通常有两种观点，第一种观点是提出直接用端到端的通过环境学习低层策略，例如前文介绍的 STRAW (Vezhnevets et al., 2016) 和选项-批判者结构 (Bacon et al., 2017)。第二种观点认为通过辅助奖励进行学习可以获得更好的分层效果，例如前文提到过的 FuN (Vezhnevets et al., 2017) 和 HIRO (Nachum et al., 2018)。

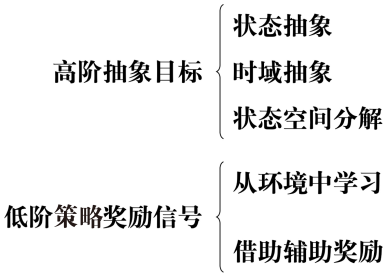

图 10.6 HRL 算法的两个视角

一般来说，第一种观点可以从端到端学习中获得更为有效的效果。这个分支下的主要工作聚焦在选项上。对于选项的发现方法，STRAW (Vezhnevets et al., 2016) 和选项-评判者结构 (Bacon

et al., 2017) 都可以被视为自上而下的方法，这种方法先通过探索获得一些奖励信号，随后对动作进行拆解，从而组成选项。与之不同的是，文献 (Machado et al., 2017) 介绍了一种自下而上的方法，该方法使用了在一个 Laplacian 图框架下的原始值函数（Proto-Value Functions，PVFs）来对环境进行表示学习，为任务无关的选项提供了理论基础。文献 (Riemer et al., 2018) 扩展了选项-批判者结构，并得出了一个深度分层选项的策略梯度定理。实验结果表明，分层选项-批判者在离散和连续的环境中都十分有效。文献 (Harutyunyan et al., 2018) 仿照离线策略学习中的做法，将终止条件解耦为行为终止和目标终止。该方法在文中的实验里表现出了更快的收敛速度。文献 (Sharma et al., 2017) 受到 SMDP 视角下的选项的启发，提出了细粒度动作重复（Fine Grained Action Repetition，FiGAR）。它能通过学习来预测选择出的动作要被重复执行的时间步数。

此外，另外一种直观的方法是将元学习与这种端到端的方法结合起来形成一个层次结构。文献 (Frans et al., 2017) 开发了一个能提升未知任务采样效率的元学习算法，该算法共享了分层结构中的基础策略，并在 3D 仿人机器人上取得了显著的成果。然而，由于对最终任务具有唯一依赖性，如何将该方法扩展到复杂领域仍然是一个问题 (Bacon et al., 2017; Frans et al., 2017; Nachum et al., 2018)。

第二种观点是使用辅助奖励。FuN (Vezhnevets et al., 2017) 和 HIRO (Nachum et al., 2018) 都为低层策略建立了目标导向的内在奖励。有许多其他的工作聚焦于能在一系列领域上有效的目标导向奖励。通用价值函数逼近器（Universal Value Function Approximators, UVFAs）(Schaul et al., 2015) 在目标上泛化价值函数。文献 (Levy et al., 2018) 进一步引入了后见之明目标转移，扩展了后见之明经验回放（Hindsight Experience Replay，HER）(Andrychowicz et al., 2017) 的思想，并取得了显著的稳定性。文献 (Kulkarni et al., 2016) 介绍了 h-DQN 算法，它学习不同时间尺度下的分层动作价值函数。其中顶层的动作价值函数学习选项策略，低层的动作价值函数学习如何达到给定的子目标。

另外也可以利用领域知识来构建手工辅助奖励。文献 (Heess et al., 2016) 介绍了一个用于移动任务的结构，它会先在相关简单任务上进行预训练。文献 (Tessler et al., 2017) 提出了应用在《我的世界》游戏领域的终生学习系统。它会有选择地将学到的技能转移到新任务上。文献 (Florensa et al., 2017) 引入了随机神经网络结构，它通过预训练的技能来学习高层策略，需要最少的下游任务领域知识，并可以很好利用学到的技能的可迁移性。然而，无论是目标导向奖励和手工奖励都很难简单地将任务扩展到其他领域，比如像素级观测的领域。

我们也可以从抽象目标的视角来理解 HRL 算法。选项框架通常学习时域抽象，而 FuN 则考虑状态抽象。HIRO 可以被认为既考虑了状态抽象又考虑了时域抽象。其目标提供了状态方向和目标转移函数模型的时间信息。对于时域抽象，与选项框架相比，文献 (Haarnoja et al., 2018) 使用了图模型来实现另一个分层思想。在该分层中，若当前任务没有完全成功，则每一层解决自己当前的任务。这会使上层的工作更为简单。在状态抽象和时域抽象之外，文献 (Mnih et al., 2014) 提供了一个利用注意力机制对状态空间进行分解的方法。更准确地说，这项工作在选择动作前，在状态空间增加了一个视觉注意力机制。此处的注意力完成了在状态空间上的高层规划 (Sahni

et al., 2017; Schulman, 2016)。对于选择抽象对象，其核心是回答高层策略如何指导低层策略的问题。对于有足够先验知识的领域，通过元学习进行技能组合学习的方式可以取得更好的效果。对于长期规划，顶层的时域抽象是十分必要的。

我们可以看出，HRL 仍然是强化学习的一个高级课题，还有许多问题需要解决。回想起 HRL 的动机是通过分层抽象来提高采样效率和通过重用学到的技巧来处理大时间跨度的问题。实验结果表明，分层架构带来了一些效果提升，但并没有足够的证据表明，它是确实实现了分层抽象，或者只是进行了更有效的探索 (Nachum et al., 2018)。未来，在概率规划、相关理论研究，以及在其他强化学习领域进行分层等方向上，可能会带来新的突破。

参考文献

ANDRYCHOWICZ M, WOLSKI F, RAY A, et al., 2017. Hindsight experience replay[C]//Advances in Neural Information Processing Systems. 5048-5058.

BACON P L, HARB J, PRECUP D, 2017. The option-critic architecture[C]//Thirty-First AAAI Conference on Artificial Intelligence.

BARTO A G, MAHADEVAN S, 2003. Recent advances in hierarchical reinforcement learning[J]. Discrete event dynamic systems, 13(1-2): 41-77.

BEATTIE C, LEIBO J Z, TEPLYASHIN D, et al., 2016. DeepMind Lab[J]. arXiv:1612.03801.

BELLEMARE M G, NADDAF Y, VENESS J, et al., 2013. The Arcade Learning Environment: An evaluation platform for general agents[J]. Journal of Artificial Intelligence Research, 47: 253-279.

BHATTI S, DESMAISON A, MIKSIK O, et al., 2016. Playing Doom with slam-augmented deep reinforcement learning[J]. arXiv preprint arXiv:1612.00380.

DA SILVA B, KONIDARIS G, BARTO A, 2012. Learning parameterized skills[J]. arXiv preprint arXiv:1206.6398.

DAYAN P, 1993b. Improving generalization for temporal difference learning: The successor representation[J]. Neural Computation, 5(4): 613-624.

DAYAN P, HINTON G E, 1993a. Feudal reinforcement learning[C]//Advances in Neural Information Processing Systems. 271-278.

DIETTERICH T G, 1998. The MAXQ method for hierarchical reinforcement learning.[C]//Proceedings of the International Conference on Machine Learning (ICML): volume 98. Citeseer: 118-126.

DIETTERICH T G, 2000. Hierarchical reinforcement learning with the MAXQ value function decomposition[J]. Journal of Artificial Intelligence Research, 13: 227-303.

DUAN Y, CHEN X, HOUTHOOFT R, et al., 2016. Benchmarking deep reinforcement learning for continuous control[C]//International Conference on Machine Learning. 1329-1338.

FLORENSA C, DUAN Y, ABBEEL P, 2017. Stochastic neural networks for hierarchical reinforcement learning[J]. arXiv preprint arXiv:1704.03012.

FRANS K, HO J, CHEN X, et al., 2017. Meta learning shared hierarchies[J]. arXiv preprint arXiv:1710.09767.

GREGOR K, DANIHELKA I, GRAVES A, et al., 2015. Stochastic backpropagation and approximate inference in deep generative models[C]//Proceedings of the International Conference on Machine Learning (ICML).

HAARNOJA T, HARTIKAINEN K, ABBEEL P, et al., 2018. Latent space policies for hierarchical reinforcement learning[J]. arXiv preprint arXiv:1804.02808.

HARUTYUNYAN A, VRANCX P, BACON P L, et al., 2018. Learning with options that terminate off-policy[C]//Thirty-Second AAAI Conference on Artificial Intelligence.

HAUSKNECHT M J, 2000. Temporal abstraction in reinforcement learning[D]. University of Massachusetts, Amherst.

HAUSKRECHT M, MEULEAU N, KAELBLING L P, et al., 1998. Hierarchical solution of Markov decision processes using macro-actions[C]//Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence. Morgan Kaufmann Publishers Inc.: 220-229.

HEESS N, WAYNE G, TASSA Y, et al., 2016. Learning and transfer of modulated locomotor controllers[J]. arXiv preprint arXiv:1610.05182.

KAELBLING L P, 1993. Hierarchical learning in stochastic domains: Preliminary results[C]//Proceedings of the tenth International Conference on Machine Learning (ICML): volume 951. 167-173.

KEMPKA M, WYDMUCH M, RUNC G, et al., 2016. ViZDoom: A Doom-based AI research platform for visual reinforcement learning[C]//2016 IEEE Conference on Computational Intelligence and Games (CIG). IEEE: 1-8.

KONDA V R, TSITSIKLIS J N, 2000. Actor-critic algorithms[C]//Advances in Neural Information Processing Systems. 1008-1014.

KONIDARIS G, BARTO A G, 2009. Skill discovery in continuous reinforcement learning domains using skill chaining[C]//Advances in Neural Information Processing Systems. 1015-1023.

KULKARNI T D, NARASIMHAN K, SAEEDI A, et al., 2016. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation[C]//Advances in Neural Information Processing Systems. 3675-3683.

LEVINE S, PASTOR P, KRIZHEVSKY A, et al., 2018. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection[J]. The International Journal of Robotics Research, 37(4-5): 421-436.

LEVY A, PLATT R, SAENKO K, 2018. Hierarchical reinforcement learning with hindsight[J]. arXiv preprint arXiv:1805.08180.

MACHADO M C, BELLEMARE M G, BOWLING M, 2017. A Laplacian framework for option discovery in reinforcement learning[C]//Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org: 2295-2304.

MNIH V, HEESS N, GRAVES A, et al., 2014. Recurrent models of visual attention[C]//Advances in Neural Information Processing Systems. 2204-2212.

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.

NACHUM O, GU S S, LEE H, et al., 2018. Data-efficient hierarchical reinforcement learning[C]// Advances in Neural Information Processing Systems. 3303-3313.

OPENAI, 2018. Openai five[Z].

PARR R, RUSSELL S J, 1998a. Reinforcement learning with hierarchies of machines[C]//Advances in Neural Information Processing Systems. 1043-1049.

PARR R E, RUSSELL S, 1998b. Hierarchical control and learning for Markov decision processes[M]. University of California, Berkeley Berkeley, CA.

RIEMER M, LIU M, TESAURO G, 2018. Learning abstract options[C]//Advances in Neural Information Processing Systems. 10424-10434.

SAHNI H, KUMAR S, TEJANI F, et al., 2017. State space decomposition and subgoal creation for transfer in deep reinforcement learning[J]. arXiv preprint arXiv:1705.08997.

SCHAUL T, HORGAN D, GREGOR K, et al., 2015. Universal value function approximators[C]// International Conference on Machine Learning. 1312-1320.

SCHULMAN J, 2016. Optimizing expectations: From deep reinforcement learning to stochastic computation graphs[D]. UC Berkeley.

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

SHARMA S, LAKSHMINARAYANAN A S, RAVINDRAN B, 2017. Learning to repeat: Fine grained action repetition for deep reinforcement learning[J]. arXiv preprint arXiv:1702.06054.

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.

SUTTON R S, PRECUP D, SINGH S, 1999. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning[J]. Artificial intelligence, 112(1-2): 181-211.

TAMAR A, WU Y, THOMAS G, et al., 2016. Value iteration networks[C]//Advances in Neural Information Processing Systems. 2154-2162.

TESSLER C, GIVONY S, ZAHAVY T, et al., 2017. A deep hierarchical approach to lifelong learning in Minecraft[C]//Thirty-First AAAI Conference on Artificial Intelligence.

VEZHNEVETS A, MNIH V, OSINDERO S, et al., 2016. Strategic attentive writer for learning macro-actions[C]//Advances in Neural Information Processing Systems. 3486-3494.

VEZHNEVETS A S, OSINDERO S, SCHAUL T, et al., 2017. Feudal networks for hierarchical reinforcement learning[C]//Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org: 3540-3549.

VINYALS O, BABUSCHKIN I, CZARNECKI W M, et al., 2019. Grandmaster level in starcraft ii using multi-agent reinforcement learning[J]. Nature, 575(7782): 350-354.

WIERING M, SCHMIDHUBER J, 1997. HQ-learning[J]. Adaptive Behavior, 6(2): 219-246.