基于值函数的深度强化学习算法研究进展

摘　要 强化学习是当前机器学习的一个比较新的领域。跟监督学习和无监督学习相比，强化学习不是通过大量的静态数据学习一种映射关系，而是通过不断与环境进行交互，不断“试错”，动态的学习环境状态到所采取策略的映射关系。本文首先从强化学习的基本概率出发，主要以基于值函数的强化学习算法主线，阐述深度强化学习的研究发展状况，包括各种深度强化学习算法的演进，以及应用前景，最后简要介绍其发展方向。

关键词 强化学习，Value-base，Q-learning

第一章 引言

受行为驱动学科的启发，强化学习是关于智能体与环境交互的一种学习方式。环境处于一定的状态下，智能体通过一种行为与环境产生互动，然后环境给与智能体奖励，如何最大化这种奖励就是强化学习要解决的问题。20世纪末，伴随着机器人应用的研究发展，强化学习也逐渐称为热门的研究领域。特别是近十几年来随着深度学习的研究，越来越多的研究与深度学习相结合取得了突破性的进展，如[hoho：Hinton的Reducing the dimensionality of data with neural networks]用RBM深度神经网络实现对图像的降维，取得比PCA[hoho: PCA引用]更好的效果，被誉为首次深度学习兴起的成功实践。强化学习也不例外，通过与深度学习的结合，解决了很多复杂的连续状态空间、连续动作空间的问题，为机器人、金融、生物工程等领域开创了新局面。

强化学习可以分为基于模型的学习（model-base）和无模型的学习（model-free）两大类。基于模型的算法需要预先知道环境的状态转移函数和奖励函数，或者可以根据智能体与环境的交互采样数据学习到，譬如用动态规划的策略迭代和价值迭代算法，经典的Dyna-Q算法[hoho: 引用]等。无模型的算法则相反，不知道环境的模型参数，而是直接通过智能体与环境的交互数据直接学习策略或状态价值，如DQN、策略梯度、DDPG、PPO、SAC[hoho: 引用]等，都是这类型的算法。两种学习方法各有优缺点。通常在确定性环境中会使用基于模型的学习算法，如某些具有严格规则的棋牌类游戏。但这种白盒环境在现实中很少，很难对这种复杂环境建立良好的模型，这时使用无模型的学习算法会更加容易训练。 本文介绍的基于值函数的算法也都是属于无模型类型的学习算法。

本文只关注基于值函数的强化学习算法，是由于这类算法让人更容易进入强化学习这个领域，它更直接，更具代表性。以下从这几方面阐述基于值函数深度强化学习算法的研究进展：

- 第二章回顾强化学习的基本概念；
- 第三章列举近几十年来基于值的深度强化学习算法进展，并阐明算法的基本流程；
- 第四、五章阐述强化学习的应用领域与发展前景，总结当前强化学习领域需要解决的问题。

第二章 强化学习基本概念回顾

在介绍深度强化学习研究进展之前，我们先回顾一下强化学习的基本概念。

1. 马尔可夫决策过程（以下简称为MDP）

MDP是强化学习的基础理论。一个MDP描述为：在环境状态为s_t时，智能体采取动作a_t与环境进行交互，环境反馈智能体奖励r_t，并转移为新的状态s_{t+1}，如图1所示。

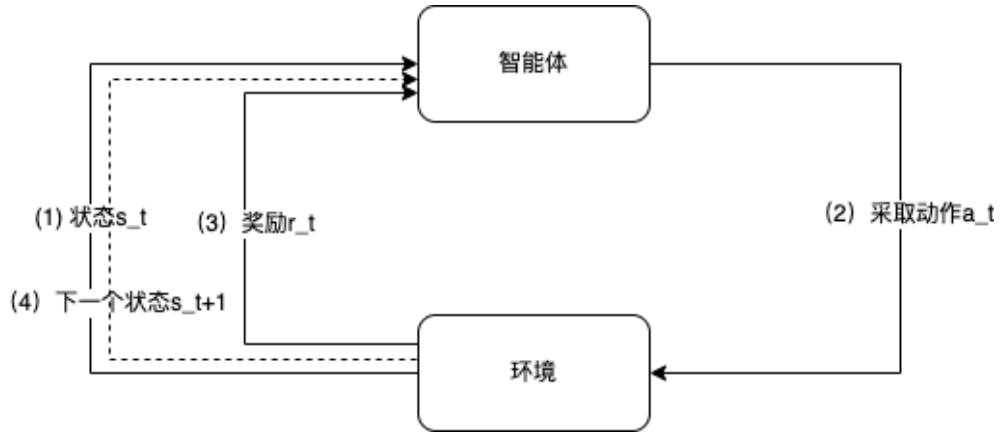

图1 一次马尔科夫决策过程

所以，MDP可描述为一个五元组：

$\{S, P, A, r, \gamma\}$，其中：

- S：状态的集合，$s_t \in S$；
- P：状态转移概率，如$p(s_{t+1}|s_t, a_t)$表示在状态为s_t（时间步t的状态）采取动作a_t的条件下，下一个时间步的状态为s_{t+1}的概率；
- A：动作的集合，$a_t \in A$；
- $r(s)$：某状态s下的奖励函数

- γ：折扣因子，$\tau \in [0,1]$

2. 策略π

策略表示智能体在状态s下所采取动作a的概率分布，一般表示为$\pi(a_t|s_t) = p(a_t|s_t)$。策略也分为：

- 确定性策略：每个状态下只能输出一个动作，只有该动作概率为1，其他动作为0；
- 随机策略：每个动作都服从一定的概率分布。譬如$\epsilon - greedy$的策略，智能体每次采取动作时有ϵ的概率（ϵ通常是一个很小的值）使用均匀分布采样一个动作，而有$1-\epsilon$的概率采取贪婪策略，采取价值最大的动作，从而一定程度平衡了智能体对环境的探索与策略的利用。实践时，随着智能体探索的步骤越来越多，通常ϵ会随时间步进行递减，符合智能体从一开始不知环境状态而增大探索能力，到后来逐渐熟悉环境而确定策略的过程。

3. 累积回报G

累积回报是智能体在每个状态下（一直到终结状态）获得的奖励的衰减之和：$G_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$，其中$\gamma$为折扣因子。折扣因子的存在是为了避免奖励之和的无穷大而使得智能体缺乏探索的动力。

4. 价值函数

状态的期望回报称为这个状态的价值函数（简称为V函数）：$V_\pi(s_t) = \mathbb{E}_\pi[G_t|s_t]$，其中$\pi$是当下所采取的策略。将状态价值函数的公式展开，就可以得到状态价值函数的递归表达：

$$
\begin{aligned}
V_\pi(s_t) &= \mathbb{E}_\pi[G_t|s_t] \\
&= \mathbb{E}_\pi[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + ...|s_t] \\
&= \mathbb{E}_\pi[r_t + \gamma(r_{t+1} + \gamma r_{t+2} + ...)|s_t] \\
&= \mathbb{E}_\pi[r_t + \gamma G_{t+1}|s_t] \\
&= \mathbb{E}_\pi[r_t + \gamma V(s_{t+1})|s_t] \\
&= r(s_t) + \gamma \sum_{s_{t+1} \in S} p(s_{t+1}|s_t) V(s_{t+1})
\end{aligned}
$$

还有另外一种价值函数：在特征状态下采取动作的的期望回报称为状态-动作价值函数（也简称为动作价值函数或Q函数）：$Q_\pi(s_t, a_t) = \mathbb{E}_\pi[G_t|s_t, a_t]$，两者之间的关系为：

$$V_\pi(s_t)=\sum_{a_t\in A}\pi(a_t|s_t)Q_\pi(s_t,a_t)$$

$$Q_\pi(s_t,a_t)=r(s_t,a_t)+\gamma\sum_{s_{t+1}\in S}p(s_{t+1}|s_t,a_t)V_\pi(s_{t+1})$$

强化学习的目的就是要最大化状态价值函数或动作价值函数。

5. 贝尔曼方程

贝尔曼是一位美国数学家，他提出并验证了著名的贝尔曼方程，成为了强化学习的基础理论。可以通过推导得出关于两个价值函数的贝尔曼期望方程：

$$\begin{aligned}Q_\pi(s_t,a_t)&=\mathbb{E}[r_t+\gamma Q_\pi(s_{t+1},a_{t+1})|s_t,a_t]\\&=r(s_t,a_t)+\gamma\sum_{s_{t+1}\in S}p(s_{t+1}|s_t,a_t)\sum_{a_{t+1}\in A}\pi(a_{t+1}|s_{t+1})Q_\pi(s_{t+1},a_{t+1})\end{aligned}$$

$$\begin{aligned}V_\pi(s_t)&=\mathbb{E}[r_t+\gamma V_\pi(s_{t+1})|s_t]\\&=\sum_{a_t\in A}\pi(a_t|s_t)(r(s_t,a_t)+\gamma\sum_{s_{t+1}\in S}p(s_{t+1}|s_t,a_t)V_\pi(s_{t+1}))\end{aligned}$$

另外还有贝尔曼最优方程，可以求出最优的两个价值函数：

$$V^*(s_t)=\max_{a_t\in A}\{r(s_t,a_t)+\gamma\sum_{s_{t+1}\in S}p(s_{t+1}|s_t,a_t)V^*(s_{t+1})\}$$

$$Q^*(s_t,a_t)=r(s_t,a_t)+\gamma\sum_{s_{t+1}\in S}p(s_{t+1}|s_t,a_t)\max_{a_{t+1}\in A}Q^*(s_{t+1},a_{t+1})$$

有了贝尔曼方程，我们就可以用动态规划的思想求解马尔科夫决策问题了。

第三章 基于值函数的强化学习算法

1. DQN

最初关于值函数的表格型方法，譬如Q-learning[hoho:]，Sarsa[hoho:]，只是解决离散状态下的离散动作问题。随着状态空间的增大，需要引入深度神经网络来“代替”这个表格。所以DQN全称为Deep Q Network [hoho: **Playing Atari with Deep Reinforcement**

Learning]，可以用来解决连续状态下的离散动作问题。为了逼近真实的值函数，可以将平方误差作为DQN的目标函数：

$$L(\theta)=\frac{1}{2N}\sum_{i=1}^{N}[Q_\theta(s_i,a_i)-(r_i+\gamma\max_{a^{'}\in A}Q_\theta(s_i^{'},a_i^{'}))]^2$$

其中θ为神经网络的参数， $s^{'}$为下一个状态，$a^{'}$为下一个动作，Q_θ为网络的预测输出，$r+\gamma\max Q_\theta$是根据贝尔曼方程得出的目标输出，网络通过不断的减少它们之间的误差进行学习。

通过实践可以看到光引入深度神经网络，网络的训练会非常不稳定。因为网络的参数不断迭代变化，计算目标的Q值也是不断变化，试想想，要逼近一个随时在变化的目标是十分困难的。 于是，Volodymyr Mnih等[hoho: Human-level control through deep reinforcement learning]提出了使用两个相同结构的神经网络f_θ和f_{θ^-}架构来组织DQN：

- 训练用的网络f_θ用于计算预测Q_θ，并在每一步使用梯度下降更新参数；
- 目标网络f_{θ^-}用于计算$r+\gamma\max Q_{\theta^-}$;
- 每隔c步，目标网络f_{θ^-}的参数与训练网络f_θ的参数同步一次；

而且，Volodymyr Mnih等还使用了一种称为经验回放的技巧：将智能体与环境每一步交互的经验数据$\{s_t,a_t,r_t,s_{t+1},done\}$暂存在一个缓冲区，如果缓冲区已满，则抛弃旧的经验数据。当缓冲区收集到足够多的经验数据时，均匀的从其中随机采样出一个batch来进行网络的训练，这样做的好处是：

- 使训练样本满足独立假设。从马尔可夫决策过程可知经验数据通常是前后相关的，而前后相关的样本数据不能直接用来训练网络。通过随机采样可以一定程度的打破样本的相关性，时样本变得独立；
- 缓冲区的样本可以重复使用，从而提高样本的使用效率；

后来，Tom Schaul等【hoho:**Prioritized Experience Replay**】发现认为TD误差大（$TD_{error}=Q_\theta(s_i,a_i)-(r_i+\gamma\max_{a^{'}\in A}Q_\theta(s_i^{'},a_i^{'}))$）的样本应该给予更高的权重，提出了基于优先级的经验回放方法，使用TD误差作为样本采样的优先级，用非均匀采样替代了之前的均匀采样，获得了不错的训练效果。

2. DDQN

普通的DQN中选择动作和计算动作Q值都是用同一个网络，很容易造成Q值的高估。实际上因为强化学习中总是进行最大化价值的操作，Q值高估的现象十分常见。Hado van Hasselt等提出的Double DQN（简称DDQN）【hoho：**Deep Reinforcement Learning with Double Q-learning**】致力于解决这一问题。DDQN选择动作用的时训练用的网络f_θ，而计算该动作价值的则用目标网络f_{θ^-}，由于目标网络f_{θ^-}的参数更新不会

很频繁，一定程度抑制的Q值的过大。图2是论文给出的分别使用DQN和DDQN进行57个Atari游戏的得分情况，实践证明DDQN比普通的DQN性能有所提升。

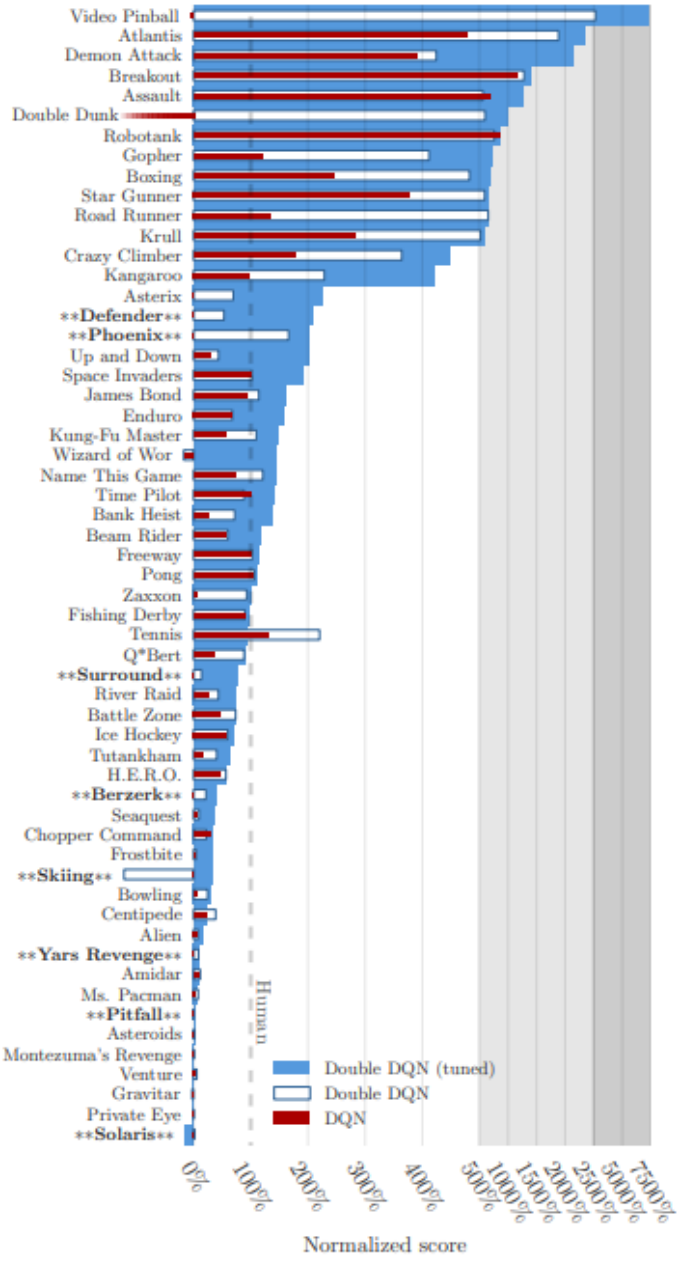

图2 用57个Atari游戏得分对比DQN和DDQN性能【hoho】

3. Dueling DQN

Wangzi Yu等提出的Dueling DQN【hoho：**Dueling Network Architectures for Deep Reinforcement Learning**】也是在DQN的基础上进行改进，以改善Q值高估的问题。论文指出，根据优势函数表示采取不同动作差异性的意义：$A(s,a) = Q(s,a) +$

$V(s)$，神经网络不直接输出Q值，Dueling DQN改为输出状态价值（V值）和优势值（A值），再求和得出Q值，网络架构改动如图3所示。

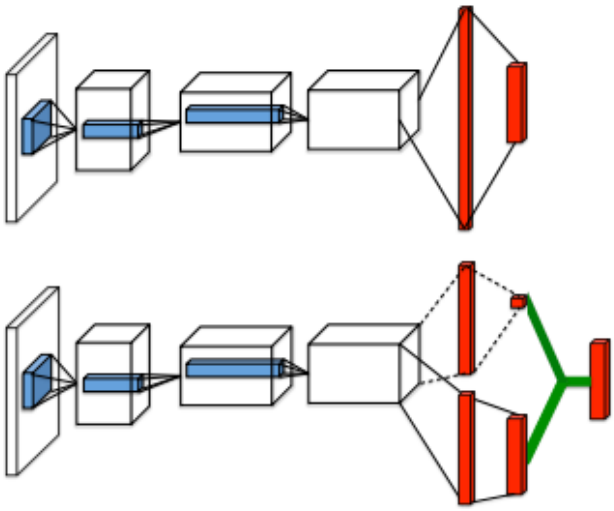

图3 普通DQN（上）和Dueling DQN（下）架构对比【hoho】

$V(s)$相当于网络对当前状态的一个基本判断，然后再根据当前状态决定采取哪个动作具有优势增益，即$A(s,a)$，所以这种拆分也是十分符合人的习惯。

然而，这里还有一个网络输出的唯一性问题要解决。试想想，令$V^{'}=V+10, A^{'}=A-10$，则$Q=V+A=V^{'}+A^{'}$，表明无法通过学习Q来唯一确定V和A，这样会导致训练的不稳定性。论文中增加一个$\max_{a^{'}} A(s,a^{'})$约束，于是模型变为$Q=V+A-\max A$，因为所有动作的优势之和必为0，这不改变模型的输出。这时，

$$\begin{aligned} Q &= V+A-\max A \\ &= (V^{'}-10)+(A^{'}+10)-\max(A^{'}-10) \\ &= V^{'}+A^{'}-10 \end{aligned}$$

可见网络学习到的V和A是唯一的。

4. noise DQN

在进行强化学习时为了保证智能体的探索能力，通常会使用如$\epsilon-greedy$策略进行探索。但是探索是与状态有关联的，看到同样的状态，往往会采取同样的探索方法，这比较符合人类的直觉。Merie Fortunato等【hoho:Noisy networks for exploration】认为之前都是在动作上添加噪声，探索效率较低，于是提出了NoiseNet方法，在神经网络的权重

上添加参数化的噪声，以此提高算法探索的效率。论文指出，在神经网络权重加入噪声带来的不确定性，比在策略上的更加大，而且噪声的参数也是需要学习的，网络可以通过学习来调整噪声的大小。

假设通常的网络建模为$y = \omega x + b$，NoiseNet则增加网络权重ω和b的均值μ和标准差σ参数，它们也是网络学习的参数，于是建模改为$y = (\mu^{\omega} + \sigma^{\omega} \odot \epsilon)x + (\mu^{b} + \sigma^{b} \odot \epsilon)$，其中$\epsilon$为噪声。

5. DQN与RNN、Attention的结合

普通的DQN对状态序列割裂得比较严重，往往很少考虑之前状态对现在的影响。Matthew Hausknecht等提出的LSTM+DQN【hoho: Deep Recurrent Q-Learning for Partially Observable MDPs】的DRQN方法用于解决这种部分观测的马尔科夫决策过程问题。作者将DQN网络最后一层全连接层改为LSTM，最终网络结构如图4。

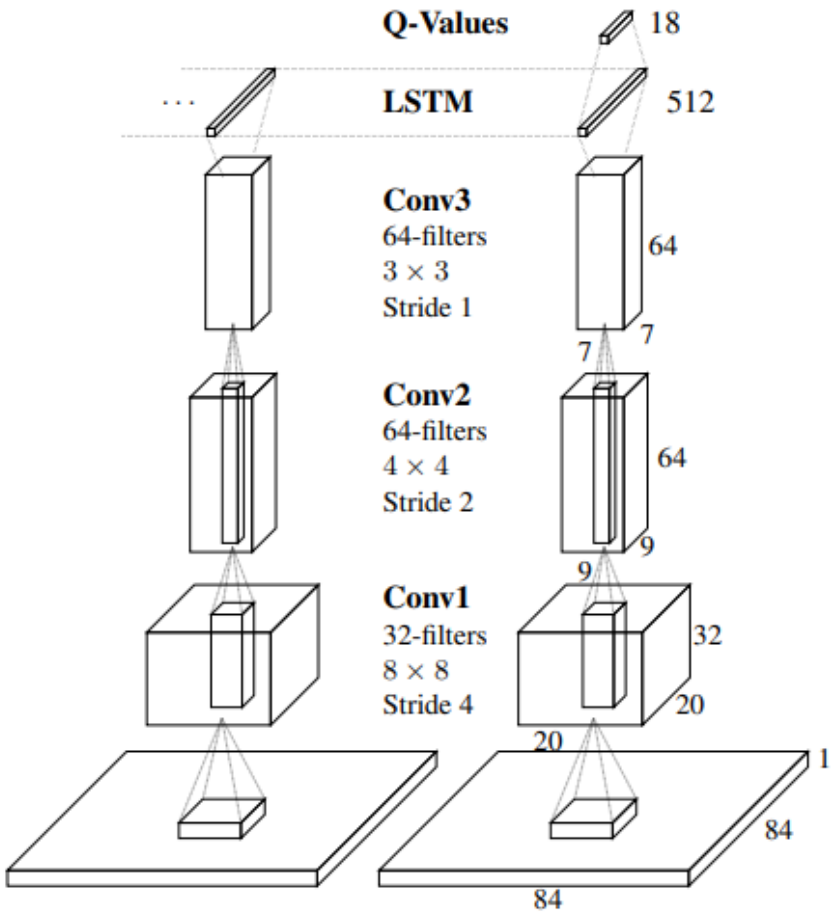

图4 DRQN网络结构【hoho】

作者采用两种方式更新网络：

- Bootstrapped Sequential Updates：每次抽取一整个游戏回合（episode）进行学习。
- Bootstrapped Random Updates：每次随机选取一次游戏回合（episode），再从这个游戏回合中随机选择一个时刻点，选择若干时间步进行学习。

Ivan Sorokin等【hoho:**Deep Attention Recurrent Q-Network**】在此基础上增加注意力机制，提出了DARQN，使得LSTM层不仅要选择下一个动作，还要选择下一个动作要关注的区域，进一步增强了时序强化学习的可解释性。

6. H-DQN

Deepmind和MIT【hoho: Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation】提出一种分层强化学习方法，旨在解决环境反馈奖励的稀疏和延迟问题。当环境反馈的奖励过于稀疏时，智能体可能因为长期都没有办法获得正奖励的样本，给学习带来困难，使得模型难以收敛。通过分层把策略分为不同层级的子策略，每个子策略在学习的过程中会得到来自上一层级传递来的奖励，这样可以大大提升网络学习的效率。图5是H-DQN的基本架构。

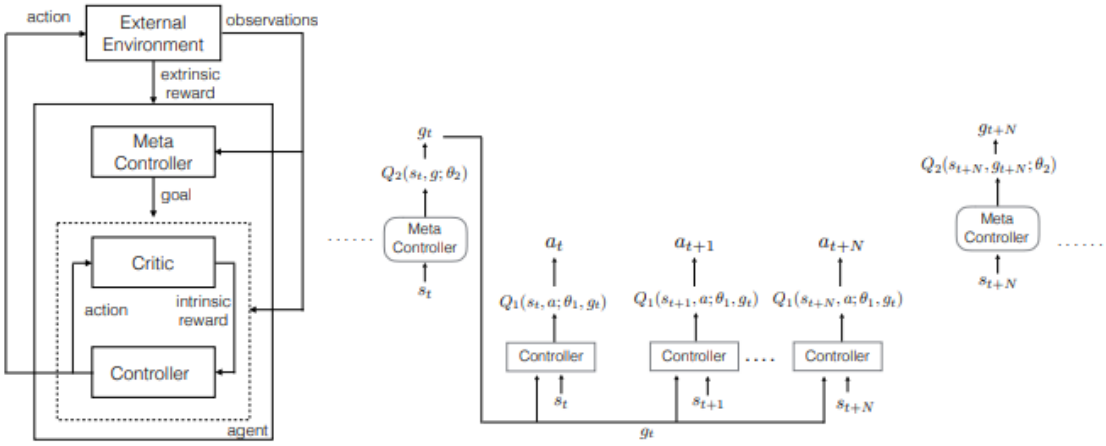

图5 分层强化学习的网络架构【hoho】

作者建立了双层网络架构，第一层称为Meta Controller，会接收外部的奖励，以及负责制定一个小目标，第二层是一个低级别的Controller，根据Meta Controller制定的小目标采取动作，小目标达到或者到达规定时间后，会给予内部奖励， 然后Meta Controller重复制定新的小目标。

7. Rainbow

第四章 应用与前景

第五章 总结