Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation

Tejas D. Kulkarni*
DeepMind, London
tejasdkulkarni@gmail.com

Karthik R. Narasimhan*
CSAIL, MIT
karthikn@mit.edu

Ardavan Saeedi
CSAIL, MIT
ardavans@mit.edu

Joshua B. Tenenbaum
BCS, MIT
jbt@mit.edu

【强化学习算法 15】h-DQN

张楚珩
清华大学 交叉信息院博士在读

这里的h-DQN是一种hierarchical deep reinforcement learning方法。

原文传送门：

Kulkarni, Tejas D., et al. "Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation." Advances in neural information processing systems. 2016.

特色：

有一类比较困难的强化学习问题，其环境反馈是sparse和delayed的。这里的解决方法是构造一个两个层级的算法。这很符合人类完成一个复杂任务的模式，遇到一个复杂任务的时候，我们会把它拆解成一系列的小目标，然后逐个去实现这些小目标。通过这样的算法，文章能够学习到Montezuma's Revenge游戏（一个简单的类似魔塔的小游戏，形式上也有点像超级马里奥）的策略。

分类：

model-free hierarchical RL, value-based, for specific task

过程：

考虑一个两层级的算法。

一个层级叫做**meta-controller**，它负责获取当前状态 s_t ，然后从可能的子任务里面选取一个子任务 $g_t \in \mathcal{G}$ 交代给下一个层级的控制器去完成。它是一个强化学习算法，其目标是最大化实际得到的extrinsic reward之和， $F_t = \sum_{t'=t}^{\infty} \gamma^{t'-t} f_{t'}$ 。在这里，这一层使用的是DQN方法，这一层Q-value的更新目标是

$$Q_2(s,g) = \max_{\pi_g} \mathrm{E}\Big[\sum_{t'=t}^{t+N} f_{t'} + \gamma \max_{g'} Q_2(s_{t+N}, g') \mid s_t = s, g_t = g, \pi_g\Big] \tag{2}$$

另一个层级叫做**controller**，它负责接收上一个层级的子任务 g 以及当前的状态 s_t，然后选择一个可能的行动 a_t 去执行。它也是一个强化学习算法，其目标是最大化一个人为规定的critic给出的intrinsic reward之和， $R_t(g) = \sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'}(g)$ 。这里也使用DQN方法，更新目标为

$$\begin{aligned} Q_1(s,a;g) &= \max_{\pi_{ag}} \mathrm{E}\Big[\sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'} \mid s_t = s, a_t = a, g_t = g, \pi_{ag}\Big] \\ &= \max_{\pi_{ag}} \mathrm{E}\big[r_t + \gamma \max_{a_{t+1}} Q_1(s_{t+1}, a_{t+1}; g) \mid s_t = s, a_t = a, g_t = g, \pi_{ag}\big] \end{aligned} \tag{1}$$

算法框架如图所示

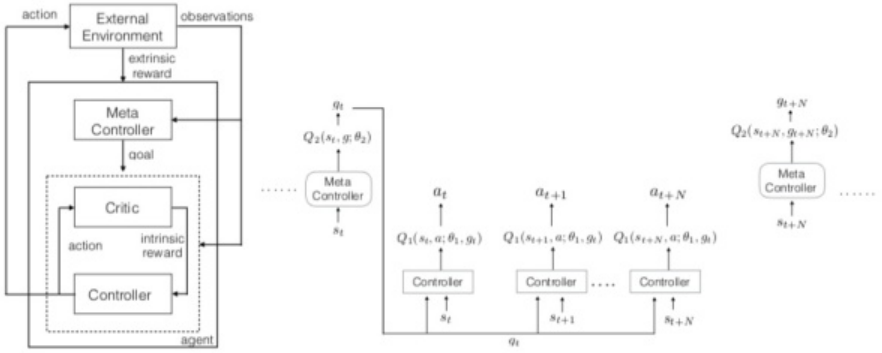

Figure 1: (**Overview**) The agent receives sensory observations and produces actions. Separate DQNs are used inside the *meta-controller* and *controller*. The meta-controller looks at the raw states and produces a policy over goals by estimating the action-value function $Q_2(s_t, g_t; \theta_2)$ (to maximize expected future extrinsic reward). The controller takes in states and the current goal, and produces a policy over actions by estimating the action-value function $Q_1(s_t, a_t; \theta_1, g_t)$ to solve the predicted goal (by maximizing expected future intrinsic reward). The internal critic provides a positive reward to the controller if and only if the goal is reached. The controller terminates either when the episode ends or when g is accomplished. The meta-controller then chooses a new g and the process repeats.

算法：

Algorithm 1 Learning algorithm for h-DQN

Initialize experience replay memories $\{\mathcal{D}_1, \mathcal{D}_2\}$ and parameters $\{\theta_1, \theta_2\}$ for the controller and meta-controller respectively.
Initialize exploration probability $\epsilon_{1,g} = 1$ for the controller for all goals g and $\epsilon_2 = 1$ for the meta-controller.
for $i = 1, num\_episodes$ **do**
    Initialize game and get start state description s
    $g \leftarrow \text{EPSGREEDY}(s, \mathcal{G}, \epsilon_2, Q_2)$
    while s is **not** terminal **do**
        $F \leftarrow 0$
        $s_0 \leftarrow s$
        while not (s is terminal **or** goal g reached) **do**
            $a \leftarrow \text{EPSGREEDY}(\{s, g\}, \mathcal{A}, \epsilon_{1,g}, Q_1)$
            Execute a and obtain next state s' and extrinsic reward f from environment
            Obtain intrinsic reward $r(s, a, s')$ from internal critic
            Store transition $(\{s, g\}, a, r, \{s', g\})$ in $\mathcal{D}_1$
            $\text{UPDATEPARAMS}(\mathcal{L}_1(\theta_{1,i}), \mathcal{D}_1)$
            $\text{UPDATEPARAMS}(\mathcal{L}_2(\theta_{2,i}), \mathcal{D}_2)$
            $F \leftarrow F + f$
            $s \leftarrow s'$
        end while
        Store transition (s_0, g, F, s') in $\mathcal{D}_2$
        if s is **not** terminal **then**
            $g \leftarrow \text{EPSGREEDY}(s, \mathcal{G}, \epsilon_2, Q_2)$
        end if
    end while
    Anneal ϵ_2 and ϵ_1.
end for

Algorithm 2 : $\text{EPSGREEDY}(x, \mathcal{B}, \epsilon, Q)$

if random() $< \epsilon$ **then**
    return random element from set $\mathcal{B}$
else
    return $\text{argmax}_{m \in \mathcal{B}} Q(x, m)$
end if

Algorithm 3 : $\text{UPDATEPARAMS}(\mathcal{L}, \mathcal{D})$

Randomly sample mini-batches from $\mathcal{D}$
Perform gradient descent on loss $\mathcal{L}(\theta)$ (cf. (3))

这个工作有哪些不足之处？

1. 对于特定的这个Montezuma’s Revenge任务做了太多特定的engineering，对于其他任务并不是普

遍适用。比如critic的判断规定为“小人是否到达某个位置”这样的yes-or-no的判断条件；再比如由于子任务是需要小人到达某个位置，因此就先训练子网络让子网络基本上能知道如何操作才能使得小人移动到规定的地点。个人感觉，文章选择的这个任务属于本身就具有明显层级结构的任务，只要做了合适的分层，效果应该都还有。

2. 文章并没有对比其他hierarchical RL的算法（当然，也有可能2016年还没有特别多这方面可以用于对比的算法）。文中的算法是面向特定任务有做特定工程优化的算法，用于对比的baseline算法只是简单的DQN，这样的对比也不公平。

3. sub-goal的选取和学习是否可以更普适呢？最近看的有些文章感觉做的更好，比如reward稀疏我们就更多地利用环境反馈的state，我们希望个体能更多样地探索状态空间[1]；再比如好奇心算法[2]（还没仔细看）。其他的这些方法之后有空再聊。

[1] Eysenbach, Benjamin, et al. "Diversity is All You Need: Learning Skills without a Reward Function."*arXiv preprint arXiv:1802.06070*(2018).

[2] Burda, Yuri, et al. "Large-Scale Study of Curiosity-Driven Learning."*arXiv preprint arXiv:1808.04355*(2018).

发布于 2018-10-08

算法　机器学习　强化学习 (Reinforcement Learning)