Hindsight Experience Replay

Marcin Andrychowicz*, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong,
Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel†, Wojciech Zaremba†
OpenAI

【强化学习算法 34】HER

张楚珩
清华大学 交叉信息院博士在读

HER是Hindsight Experience Replay的简称，hindsight大概就是“事后诸葛亮”的意思。

原文传送门

Andrychowicz, Marcin, et al. "Hindsight experience replay." Advances in Neural Information Processing Systems. 2017.

特色

奖励稀疏是导致强化学习困难的一个重要的原因，本专栏前面讲过一些reward shaping方法或者intrinsic reward的设计方法，都是解决奖励稀疏的一个途径。（对此不熟悉的可以参考本专栏的【强化学习思想 22】Reward Shaping Invariance）

这篇文章开发了解决奖励稀疏的另一个途径，大致思想是奖励稀疏的时候可能绝大多数的探索都是失败的，这里考虑把这些失败的探索都利用起来。比如智能体想去食堂，某一次探索失败了，没去到食堂，但是走到操场了，与其认为这一次失败的探索，不如记下来这样走可以到操场，这样更加充分地利用了经历的失败。这篇文章就是在讲如何构建这样一个经验池。

文章只通过0-1奖励函数就训练取得了不错的性能，并且在实体机械臂的实验上取得了很好的效果。

过程

1. 附加目标的奖励和价值函数

对于强化学习常用的MDP设定 $(\mathcal{S},\mathcal{A},P,R,\gamma)$ ，不改变MDP的dynamics，但对于此MDP附加一个目标 $g\in\mathcal{G}$ 。对于目标稍加限制，这个目标描述了任务希望达到什么样的状态。

一个简单的想法就是直接把每个目标都定为相应的状态，即目标空间和状态空间做一一映射，如果智能体达到了状态 $s=g$ ，那么就是达到了目标，反之就没有达到目标。

文章中考虑到目标空间并不一定要跟状态空间一样（比如一般来说，目标空间可以更概括一些），

文章做了如下的形式化规定。

- 规定函数 $\forall g \in \mathcal{G}, f_g : \mathcal{S} \to \{0,1\}$ ，该函数表示在目标 g 下，任意一个状态是否达到该目标。
- 规定某个目标下的奖励函数 $r_g(s,a) = -[f_g(s) = 0]$ ，注意到这是一个二值奖励函数，只要没达到目标就返回-1，是一个很稀疏的奖励函数。
- 规定函数 $m : \mathcal{S} \to \mathcal{G}$ ，有 $\forall s \in \mathcal{S}, f_{m(s)}(s) = 1$ ，即对于任意一个状态都能找到其对应的目标。

上面的规定简言之就是1）任意一个目标都对应一套稀疏的奖励函数；2）任意一个状态都有与之对应的目标。

2. 附加目标的经验池

一个普通的经验池里面存放的是 (s_t, a_t, r_t, s_{t+1}) ，这里再附加地存放相应的目标，变为 $(s_t||g, a_t, r_{g,t}, s_{t+1}||g)$ ，其中 $||$ 表示直接concatenation。以此同时，行动策略（behavior policy）也是与目标相关的，其输入是当前状态和需要完成的目标，写为 $\pi_b(s_t||g)$ 。到此为止还没有很新的东西，基本上是一个普通的多任务的off-policy算法。

由于奖励的稀疏性，采集到的一条轨迹大概率地对于所需要完成的任务来说是失败的。本文最关键的一步就是在采样一条轨迹之后，对于轨迹上的每一个transition都去计算一下如果换作其他目标该transition会怎样。注意到由于多目标任务之间系统的dynamics是相同的，因此经历过的 (s_t, a_t, s_{t+1}) 都是不变的，只需要重新计算一下新目标 g' 下的奖励 $r' = r(s_t, a_t, g')$ 就可以了。

3. 算法

由此可以自然得到最后的算法。注意到最为核心的是里面第二个小for循环的部分。

Algorithm 1 Hindsight Experience Replay (HER)

Given:

- an off-policy RL algorithm $\mathbb{A}$, ▷ e.g. DQN, DDPG, NAF, SDQN
- a strategy $\mathbb{S}$ for sampling goals for replay, ▷ e.g. $\mathbb{S}(s_0, \ldots, s_T) = m(s_T)$
- a reward function $r : \mathcal{S} \times \mathcal{A} \times \mathcal{G} \to \mathbb{R}$. ▷ e.g. $r(s, a, g) = -[f_g(s) = 0]$

Initialize $\mathbb{A}$ ▷ e.g. initialize neural networks
Initialize replay buffer R
for episode = $1, M$ **do**
  Sample a goal g and an initial state s_0.
  for $t = 0, T-1$ **do**
    Sample an action a_t using the behavioral policy from $\mathbb{A}$:
      $a_t \leftarrow \pi_b(s_t||g)$ ▷ $||$ denotes concatenation
    Execute the action a_t and observe a new state s_{t+1}
  end for
  for $t = 0, T-1$ **do**
    $r_t := r(s_t, a_t, g)$
    Store the transition $(s_t||g,\ a_t,\ r_t,\ s_{t+1}||g)$ in R ▷ standard experience replay
    Sample a set of additional goals for replay $G := \mathbb{S}$(**current episode**)
    for $g' \in G$ **do**
      $r' := r(s_t, a_t, g')$
      Store the transition $(s_t||g',\ a_t,\ r',\ s_{t+1}||g')$ in R ▷ HER
    end for
  end for
  for $t = 1, N$ **do**
    Sample a minibatch B from the replay buffer R
    Perform one step of optimization using $\mathbb{A}$ and minibatch B
  end for
end for

4. 如何采样得到其他的目标？

注意到算法中第二个小for循环里面，对于每一个transition都会选择若干个新的目标 g' ，并且记录再此新目标下的奖励 r' 。那么如何选择这个新目标呢？文章尝试了以下几种方法

- 每次只选择这一条轨迹的最后一个状态对应的目标作为新目标，即 $g(s_0,\cdots,s_T)=m(s_T)$ ；这种情况下只选择一个新目标，加上原目标产生的transition，即经验池里面存放2倍于真实采样到的样本。
- 每次随机随机选择 k 个在这个轨迹上并且在这个transition之后的状态作为新目标，即如果现在的样本为 (s_t,a_t,s_{t+1}) ，那么会在 $s_{t+1},\cdots,s_T$ 之间选择 k 个状态对应的目标作为新目标（标记为future）；这种情况下经验池里面存放的样本数目是真实采样到样本数目的 $k+1$ 倍（下同）。
- 每次选择 k 个在这个轨迹上的状态作为新目标，即如果现在的样本为 (s_t,a_t,s_{t+1}) ，那么会在 $s_1,\cdots,s_T$ 之间选择 k 个状态对应的目标作为新目标（标记为episode）。
- 每次在所有出现过的状态里面选择 k 个状态作为新目标，这里甚至不再要求在这个episode内的状态里面采样了（标记为random）

最后的实验可以看到，前两个方案效果比较好，其中第二个方案效果最好。实验中比较好的取值为 $k=8$ 。

实验

实验选择了三个任务

- **pushing**：要求机械臂把一个小方块推动到平面上的指定位置（可以翻滚着推好几下）
- **sliding**：要求机械臂把一个小圆球推动到平面上的指定位置（基本上就像打台球一样推动一下）
- **pick-and-place**：要求机械臂捡起一个方块并且放置到指定的某个半空中的位置

实验结果有如下几个看点

- **HER可以结合任意的off-policy算法**：文中使用了DDPG
- **效果好**：由于这个任务需要机械臂把物体挪动到随机产生的不同位置，其本身就是一个多目标的任务，如果只设置一个稀疏的奖励，本身就很难学习到有效的东西。因此实验中DDPG+HER相比于纯DDPG的结果有了本质上的提升。（相比于其他count-based exploration策略也有很大的提升）
- **对于单目标任务也适用**：虽然这个方法自然地适用于多目标强化学习问题，但是如果对于单目标任务来说，在训练中只使用这个单一的目标也对于性能提升有帮助；不过文章还是推荐即使最后的任务是单目标，在训练的时候也使用多目标来训练；

- **效果甚至好于一些reward shaping的尝试**：文中简单尝试了几种手工的reward shaping，效果没有直接这样使用系数的目标更好。文章给出的解释是稀疏的奖励最能够反映任务目标，加了shaping之后其实是为了引导智能体学习从而牺牲了对于目标的正确表述；同时，shaping其实惩罚了先验规定的一些不想要的行动，这可能阻碍了智能体的探索。

评论

这篇文章成功的有一个重要原因是神经网络对于输入其中的目标具有泛化能力，这就是为什么有可能所需要的目标一次都没遇到过，但是最后执行的时候仍然有不错的效果。

这篇文章的主要限制在于规定了目标和状态之间的对应关系，这里的状态维度很低，并且有明确的语义。如果状态维度高或者语义不明确，就不方便基于状态来制定有语义的目标了，这一点可能限制了HER往其他任务上面的拓展。

文章由 @小红菌 推荐。

发布于 2018-12-02

强化学习 (Reinforcement Learning)

文章被以下专栏收录

强化学习前沿
读呀读paper

进入专栏