Large-Scale Study of Curiosity-Driven Learning

Yuri Burda* OpenAI
Harri Edwards* OpenAI
Deepak Pathak* UC Berkeley
Amos Storkey Univ. of Edinburgh
Trevor Darrell UC Berkeley
Alexei A. Efros UC Berkeley

【强化学习算法 17】Curiosity

张楚珩
清华大学 交叉信息院博士在读

原文的叫做 curiosity-driven learning，知乎文章标题打不了这么长，就叫Curiosity吧。

原文传送门：

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

特色：

这篇文章不使用外在的奖励，仅使用curiosity这种内在奖励，就能够在诸多Atari游戏上有很好的表现，甚至在超级马里奥上能顺利通过11关。

背景：

强化学习过程中reward的设计十分关键，好的reward最关键的是以下两点：

1. **密集（dense）**，即agent每走一步最好都有反馈告诉它这么走到底好不好，这就很像幼儿园里面的老师会不停的告诉小朋友“做得很好”一样。然而在复杂的强化学习问题里面自然的奖励方式常常是sparse and delayed的。
2. **形态好（well-shaped）**，好的奖励形态有助于算法快速地学习到好的策略。其实从某个角度来看，value-based RL算法其实就是在奖励之外构建一套价值函数，而这套价值函数具有好的形态，这样有助于策略的学习。

大家自然会想有没有办法在这些人为规定的奖励（extrinsic reward）之外定义一些密集并且形态好的内在奖励（dense and well-shaped intrinsic reward），通过它们的辅助让算法学的更好。其实已经有很多这方面的工作了，主要有以下几类

1. **Dynamic model based**：在算法中维护一个dynamic model $p_\theta(s_{t+1}|s_t, a_t)$ 。如果实际的transition和已有的这个模型差不多，那么就认为这是习以为常的，没啥意思，因此给的内在奖励就少；如果实际的transition和已有的这个模型差的很远，那么我们会觉得很新、很有意思，因此就给更多的奖励。这样的奖励方法能够刺激算法去探索新的状态空间或者动力学复杂的区域。如何判断实际的transition和模型相差多不多呢？一种方法是计算预测出来的误差（就是文中的curiosity方法）；一种方法是基于transition在模型在模型下的uncertainty（prediction uncertainty，不知道理解是否正确）；另一种是基于这个transition对模型产生的改进程度。
2. **count-based**：大体思路是如果访问到某些很少被访问到的状态就会给予较大的奖励。文中后面提到的同期工作diversity的方法就是使得访问到的状态空间的位置尽可能远。

本文就更进一步，就不只是把这些内在奖励作为辅助，而是完全就只用这些内在奖励，不使用人为规定的奖励，然后观察得到的agent能做到多好。

过程：

定义curiosity为模型预测的下一个状态的表示 $f(x_t, a_t)$ 和实际下一个状态的表示 $\phi(x_{t+1})$ 之间的L2 norm，即 $\|f(x_t, a_t) - \phi(x_{t+1})\|_2^2$ 。

一个好的状态表示 $\phi(\cdot)$ ，希望它

1. **compact**：即维度不要太高，并且不要有太多不相关的信息；
2. **sufficient**：即不要丢失太多有用的信息；
3. **stable**：不稳定的来源有二。一个是 $f(x_t, a_t)$ 中模型的不稳定，因为模型有一个逐渐学习的过程；另一个是 $\phi(\cdot)$ 中的不稳定。如果不稳定，得到的内在奖励前后就不一致了。

文章提出了几种可能的表示方法：

1. **Pixels**：即 $\phi(x) = x$ ，直接把图像当做其表示。
2. **Random Features（RF）**：即过一个权值随机的CNN，然后权值就固定不变了。结果发现在大多数问题上面这个简单的方式的表现就已经很不错了。
3. **Variational Autoencoder（VAE）**：VAE学到的隐变量作为其表示 $\phi(x) = p(x|x)$ ，随着样本的到来，学习的过程是逐步的，VAE的权值也是逐步变化的。
4. **Inverse Dynamics Features（IDF）**：训练一个神经网络拿到transition之后从 s_t 和 s_{t+1} 预测 a_t ，截取两端输入合并处的向量作为其表示。

接下来使用PPO算法来进行训练即可，同时用到了一些工程上的技巧，比如对于reward、advantage、observation进行normalization，并且做batch-norm，使用更多的parallel actors使得算法更稳定等。

局限性：

基于curiosity的定义，当环境变得随机的时候，这里的curiosity就更像是环境的熵了（这么说是基于 $r_t = -\log p(\phi(x_{t+1})|x_t, a_t)$ 的定义），因而不是更多地探索状态空间了，而是探索环境中熵比较大的位置了。

文章举了一个很有意思的例子，在走迷宫的时候如果在墙上放完全随机的电视画面，那么agent就会更倾向于停在那里看电视而不继续探索了。

是否有办法让算法不被随机电视机卡住？

注意到这里关于curiosity是定义是 $\|f(s_t, a_t) - s_{t+1}\|_2^2$ （如果不考虑做embedding）。原本预料的是如果某个地方的动力学很复杂（熵很大），那么在此处动力学模型预测会比较不准，因此这里的curiosity会比较大，这样就激励算法在这里多做探索。随机电视机这里确实有很高的熵，但是我们更多地学习也并不会带来动力学模型变得更准。

我发现有一个办法可能能够解决这一问题，就是不仅仅学习一个前项的动力学模型 $f: S_t \times A_t \to S_{t+1}$ ，还学习另外一个预测模型 $k: S_t \times A_t \times A_{t+1} \to S_{t+1}$ 。定义curiosity为 $\|f(s_t, a_t) - s_{t+1}\|_2^2 - \|k(s_t, a_t, a_{t+1}) - s_{t+1}\|_2^2$ 。对于随机电视机的情况来说，随机电视机出现之后，agent接下来的行动跟电视机的画面没啥关系，也就是说接下来的行动 a_{t+1} 不包含多少关于 s_{t+1} 的信息。这样模型 f 和模型 k 对于状态 s_{t+1} 的预测程度应该差不多，这样如果把两者相减作为奖励的话，就不会卡在这种动力学复杂，但是实际上对于模型帮助不大的地方了。

以上curiosity的定义来自于这一篇文章

de Abril, Ildefons Magrans, and Ryota Kanai. "Curiosity-driven reinforcement learning with

homeostatic regulation." arXiv preprint arXiv:1801.07440 (2018).

其中提到内在的激励可以划分为

1. agent到environment的信息流：即希望agent对环境的控制越大越好，相应的内在奖励就用于激励”采取某个行动之后能够获得和预想中差不多的结果“，或者”到达某个操作性更强的状态“（empowerment）。个人理解这些奖励类似于exploitation。
2. environment到agent的信息流：即希望环境给到agent的信息量越大于好，比如鼓励环境发生某个transition让算法内部的模型有较大的改善。个人理解这一部分类似于exploration。

这篇文章就把内在的激励定义为了环境给agent的信息流，具体地说叫causally conditioned directed mutual information，如图所示

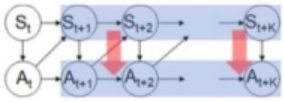

Figure 1: Conceptual diagram of the information gain process: dark thin arrows are causal dependencies and large arrows show how information flows from a sequence of observed states to the corresponding future sequence of agent actions.

$$
\begin{aligned}
&InformationGain(s_t) = I(S_{t+1:t+K} \rightarrow A_{t+1:t+K} || S_{t:t+K-1}, A_t) \\
&= I(S_{t+1}; A_{t+1} | S_t, At) \\
&+ < I(S_{t+2:t+K} \rightarrow A_{t+2:t+K} || S_{t+1:t+K-1}, A_{t+1}) >_{P(S_{t+1}, A_{t+1} | S_t, A_t)},
\end{aligned} \tag{1}
$$

这可以看成一种Bellman Equation，那么单步的奖励就是其中的第一项，第一项可以近似写为

$$
\begin{aligned}
&I(S_{t+1}; A_{t+1} | s_t, a_t) = H(S_{t+1} | S_t, A_t) - H(S_{t+1} | S_t, A_t, A_{t+1}) \\
&\approx H(S_{t+1} | s_t, a_t) - H(S_{t+1} | s_t, a_t, a_{t+1}) \\
&\approx ||s_{t+1} - \hat{s}_f||_2 - ||s_{t+1} - \hat{s}_k||_2
\end{aligned} \tag{2}
$$

对于信息论的公式和定义不熟悉的话可以参考维基百科：Mutual Information。

注：从机器之心专栏上面了解到的这篇文章

编辑于 2018-10-10

机器学习 算法 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