VIME: Variational Information Maximizing Exploration

Rein Houthooft[§†‡], Xi Chen[†‡], Yan Duan[†‡], John Schulman[†‡], Filip De Turck[§], Pieter Abbeel[†‡]
[†] UC Berkeley, Department of Electrical Engineering and Computer Sciences
[§] Ghent University - imec, Department of Information Technology
[‡] OpenAI

【强化学习算法 24】VIME

张楚珩
清华大学 交叉信息院博士在读

28 人赞同了该文章

VIME是Variational Information Maximizing Exploration的缩写，目的是为了鼓励探索，形式上是一种intrinsic reward。

原文传送门：

Houthooft, Rein, et al. "Vime: Variational information maximizing exploration." Advances in Neural Information Processing Systems. 2016.

特色：

一种基于信息增益的内在奖励，目的是弥补原本稀疏的奖励，鼓励更好地探索，能够与各种其他标准的RL算法结合使用。对了，另外的特色就是用到的信息论和数学知识比较多，换句话说就是难。这里尽量讲明白其思想，而不被数学部分搞混乱了。

过程：

1. 探索的原则

目的是在不考虑外部奖励的情况下，决定如何探索，即如何选择下一步的行动 a_t 。这里维护一个对于环境动力学的建模 $p(s_{t+1}|s_t,a_t,\theta),\theta\in\Theta$ ，而该模型的参数 Θ 看做是一个随机变量， θ 是该随机变量的值。选择行动的目标就是最大化该行动之后对于模型来说获得的信息增益，也就是希望每一步行动都能够尽可能利用这次交互机会来更多地获取环境的信息。具体地，就是每一步要最大化

$$\left(H(\Theta|\xi_t,a_t)-H(\Theta|S_{t+1},\xi_t,a_t)\right), \tag{1}$$

其中 $\xi_t=\{s_1,a_1,\cdots,s_t\}$ 表示之前的轨迹。

2. 转为为内在奖励形式

考虑到[1][2]

$$
\begin{aligned}
\mathrm{I}(X;Y) &\equiv \mathrm{H}(X) - \mathrm{H}(X|Y) \\
&\equiv \mathrm{H}(Y) - \mathrm{H}(Y|X) \\
&\equiv \mathrm{H}(X) + \mathrm{H}(Y) - \mathrm{H}(X,Y) \\
&\equiv \mathrm{H}(X,Y) - \mathrm{H}(X|Y) - \mathrm{H}(Y|X)
\end{aligned}
$$

$$
\mathrm{I}(X;Y) = \int_{y} \int_{x} p(x,y) \log \left(\frac{p(x,y)}{p(x)\,p(y)} \right) dx\,dy,
$$

(1)式就等于

$$
I\left(S_{t+1}; \Theta | \xi_t, a_t\right) = \mathbb{E}_{s_{t+1} \sim \mathcal{P}(\cdot|\xi_t, a_t)} \left[D_{\mathrm{KL}} [p(\theta|\xi_t, a_t, s_{t+1}) \| p(\theta|\xi_t))] \right], \tag{2}
$$

要通过最大化这个互信息来选择行动实在有点困难，一般行动是由策略决定的，策略是由RL的算法结合奖励学出来的，因此，我们就想，干脆把它做成内在奖励（intrinsic reward）吧。于是有

$$
r'(s_t, a_t, s_{t+1}) = r(s_t, a_t) + \eta D_{\mathrm{KL}} [p(\theta|\xi_t, a_t, s_{t+1}) \| p(\theta|\xi_t)], \tag{3}
$$

3. 内在奖励计算的困难

内在奖励是一个KL散度，散度里面的分布就不是很好计算，比如（利用Bayes' rule，并且注意到 $p(\theta|\xi_t, a_t) = p(\theta|\xi_t)$ ）

$$
p(\theta|\xi_t, a_t, s_{t+1}) = \frac{p(\theta|\xi_t) p(s_{t+1}|\xi_t, a_t; \theta)}{p(s_{t+1}|\xi_t, a_t)}, \tag{4}
$$

而其分母部分

$$
p(s_{t+1}|\xi_t, a_t) = \int_{\Theta} p(s_{t+1}|\xi_t, a_t; \theta) p(\theta|\xi_t) d\theta. \tag{5}
$$

而分布 $p(\theta|\xi_t) = p(\theta|\mathcal{D})$ 可能是奇形怪状的分布，于是上面的这些东西都没法算了。

4. 使用Variational Bayes方法

对于这样的困难有一套比较成熟的解决思路了[3][4]，我们使用一个参数化的规则的分布 $q(\theta;\phi)$ 来近似复杂的分布 $p(\theta|\mathcal{D})$ 。在这里我们选择fully factorized Gaussian distribution

$$
q(\theta;\phi) = \prod_{i=1}^{|\Theta|} \mathcal{N}(\theta_i|\mu_i;\sigma_i^2). \tag{11}
$$

其中， $\phi = \{\mu, \sigma\}$ 。在这样的情况下，内在奖励就可以写成可以计算的形式了

$$
r'(s_t, a_t, s_{t+1}) = r(s_t, a_t) + \eta D_{\mathrm{KL}} [q(\theta;\phi_{t+1}) \| q(\theta;\phi_t)], \tag{7}
$$

5. 更新参数 ϕ

参数 ϕ 可以通过最大化以下函数（variational lower bound, ELBO）来得到

$$L[q(\theta;\phi),\mathcal{D}] = \mathbb{E}_{\theta\sim q(\cdot;\phi)}\left[\log p(\mathcal{D}|\theta)\right] - D_{\mathrm{KL}}[q(\theta;\phi)\,\|\,p(\theta)]. \tag{6}$$

最大化的具体方法就是做梯度下降。

其中第一项期望里面可以写作 $\log p(\mathcal{D}|\theta) = \frac{N}{M}\sum_{i=1}^{M}\log p(s_{t+1}^{(i)}|s_t^{(i)},a_t^{(i)},\theta)$ ，其中M是采样的数目，N是数据集的大小。那么又存在一个问题， θ 是通过 $q(\cdot;\phi)$ 采样得来的，经过了采样，在求梯度的时候没办法回溯到 ϕ 了。因此，这里需要使用到reparametrization trick[4]，相应的SGD方法也叫做stochastic gradeitn variational Bayes (SGVB)。

此外，观察到 $\phi\to\theta\to p(s_{t+1}^{(i)}|s_t^{(i)},a_t^{(i)},\theta)$ ，与其先采样然后再把采样的结果送到model里面去计算概率，不如直接把分送到model里面计算出来 $p(s_{t+1}^{(i)}|s_t^{(i)},a_t^{(i)},\theta)$ 的分布，然后再采样。后者的variance更小。这种方法叫做local reparametrization trick[4]。

另外，注意到这里参数的更新没必要每产生一个新样本都更新，可以是多步之后一块更新一次。不同于后面对于内在奖励的计算，内在奖励的计算需要每一个 (s_t,a_t,s_{t+1}) 都计算得到一个奖励。

6. 计算内在奖励

回忆内在奖励的形式

$$r'(s_t,a_t,s_{t+1}) = r(s_t,a_t) + \eta D_{\mathrm{KL}}[q(\theta;\phi_{t+1})\,\|\,q(\theta;\phi_t)], \tag{7}$$

其中 $\phi'=\phi_{t+1}$ 表示拿到新样本 (s_t,a_t,s_{t+1}) 之后更新的参数，参数的更新方法是

$$\phi' = \arg\min_{\phi}\Big[\overbrace{\underbrace{D_{\mathrm{KL}}[q(\theta;\phi)\,\|\,q(\theta;\phi_{t-1})]}_{\ell_{\mathrm{KL}}(q(\theta;\phi))} - \mathbb{E}_{\theta\sim q(\cdot;\phi)}\left[\log p(s_{t+1}|\xi_t,a_t;\theta)\right]}^{\ell(q(\theta;\phi),s_t)}\Big], \tag{12}$$

注意到这个奖励每一步都要算的，每一步都算一个 $\arg\min$ 实在是吃不消，于是我们简化一下，就 $\phi'=\phi_{t-1}+\lambda\Delta\phi$ ，其中 $\Delta\phi$ 使用牛顿法得到

$$\Delta\phi = H^{-1}(\ell)\nabla_\phi\ell(q(\theta;\phi),s_t), \tag{13}$$

下面最关键的来了，注意到奖励可以写成

$$D_{KL}[q(\theta;\phi+\lambda\Delta\phi)\|q(\theta;\phi)] \approx \frac{1}{2}\lambda^2\Delta\phi^T H\Delta\phi = \frac{1}{2}\lambda^2\nabla_\phi l^T H^{-1}(l_{KL})\nabla_\phi l$$

注意到(12)式里当 ϕ' 在 ϕ_{t-1} 附近的时候，后一项主要是线性的，只有前一项有二次型的形态，因此Hessian矩阵可以只算前一项的。考虑到高斯分布参数化之后，KL散度计算十分方便，因此(13)式的 $H(l)$ 很容易计算得到。同时注意到，Hessian只有对角项，因此其逆 $H^{-1}(l)$ 也很容易求。 $\nabla_\phi l$ 再次利用reparametrization trick一样可以求到。

算法：

Algorithm 1: Variational Information Maximizing Exploration (VIME)

for each epoch n **do**
  for each timestep t in each trajectory generated during n **do**
    Generate action $a_t \sim \pi_\alpha(s_t)$ and sample state $s_{t+1} \sim \mathcal{P}(\cdot|\xi_t, a_t)$, get $r(s_t, a_t)$.
    Add triplet (s_t, a_t, s_{t+1}) to FIFO replay pool $\mathcal{R}$.
    Compute $D_{\mathrm{KL}}[q(\theta; \phi'_{n+1}) \,\|\, q(\theta; \phi_{n+1})]$ by approximation $\nabla^\top H^{-1} \nabla$, following Eq. (16) for diagonal BNNs, or by optimizing Eq. (12) to obtain ϕ'_{n+1} for general BNNs.
    Divide $D_{\mathrm{KL}}[q(\theta; \phi'_{n+1}) \,\|\, q(\theta; \phi_{n+1})]$ by median of previous KL divergences.
    Construct $r'(s_t, a_t, s_{t+1}) \leftarrow r(s_t, a_t) + \eta D_{\mathrm{KL}}[q(\theta; \phi'_{n+1}) \,\|\, q(\theta; \phi_{n+1})]$, following Eq. (7).
  Minimize $D_{\mathrm{KL}}[q(\theta; \phi_n) \,\|\, p(\theta)] - \mathbb{E}_{\theta \sim q(\cdot; \phi_n)} [\log p(\mathcal{D}|\theta)]$ following Eq. (6), with $\mathcal{D}$ sampled randomly from $\mathcal{R}$, leading to updated posterior $q(\theta; \phi_{n+1})$.
  Use rewards $\{r'(s_t, a_t, s_{t+1})\}$ to update policy π_α using any standard RL method.

$$D_{\mathrm{KL}}[q(\theta; \phi + \lambda \Delta\phi) \,\|\, q(\theta; \phi)] \approx \overbrace{\tfrac{1}{2} \lambda^2 \nabla_\phi \ell^\top H^{-1}(\ell_{\mathrm{KL}}) \nabla_\phi \ell}^{\ell(q(\theta;\phi), s_t)}. \tag{16}$$

$$\phi' = \arg\min_\phi \Big[\underbrace{D_{\mathrm{KL}}[q(\theta; \phi) \,\|\, q(\theta; \phi_{t-1})]}_{\ell_{\mathrm{KL}}(q(\theta;\phi))} - \mathbb{E}_{\theta \sim q(\cdot;\phi)} [\log p(s_t|\xi_t, a_t; \theta)] \Big], \tag{12}$$

$$r'(s_t, a_t, s_{t+1}) = r(s_t, a_t) + \eta D_{\mathrm{KL}}[q(\theta; \phi_{t+1}) \,\|\, q(\theta; \phi_t)], \tag{7}$$

$$L[q(\theta; \phi), \mathcal{D}] = \mathbb{E}_{\theta \sim q(\cdot;\phi)} [\log p(\mathcal{D}|\theta)] - D_{\mathrm{KL}}[q(\theta; \phi) \,\|\, p(\theta)]. \tag{6}$$

知乎 @张楚珩

参考资料：

[1] 信息论各种概念的定义和关系：

wiki/Mutual_information

[2] KL散度的定义：

wiki/Kullback–Leibler_divergence

[3] Variational Lower Bound的推导（很短，十分钟读完）：

Yang, Xitong. "Understanding the Variational Lower Bound." (2017).

[4] Variational Inference for BNN（包括里面提到的Reparametrization trick等，很详细）：

Variational Inference for Bayesian Neural Networks

发布于 2018-10-30

算法 机器学习 强化学习 (Reinforcement Learning)

▲ 赞同 28 ▼ 3 条评论 分享 喜欢 收藏 ···

文章被以下专栏收录

强化学习前沿
读呀读paper

进入专栏