【ICLR2020】通过强化学习和稀疏奖励进行模仿学习

小小何先生
东北大学 信息科学与工程学院硕士在读

- **论文题目：SQIL**: Imitation Learning via Reinforcement Learning with Sparse Rewards

SQIL: IMITATION LEARNING VIA REINFORCEMENT LEARNING WITH SPARSE REWARDS

Siddharth Reddy, Anca D. Dragan, Sergey Levine
Department of Electrical Engineering and Computer Science
University of California, Berkeley
{sgr,anca,svlevine}@berkeley.edu

所解决的问题？

从高维的状态动作空间中进行模仿学习是比较困难的，以往的行为克隆算法(`behavioral cloning BC`)算法容易产生分布漂移(`distribution shift`)，而最近做得比较好的就是生成对抗模仿学习算法(generative adversarial imitation learning (`GAIL`))，是逆强化(`Inverse RL`)学习算法与生成对抗网络结合的一种模仿学习算法，这个算法使用 `adversarial training` 技术学 `reward function`，而作者提出的算法不需要 `reward function` 。整篇文章是在证明 `constant reward` 的 `RL` 方法与复杂的学习 `reward function` 的强化学习算法一样有效。

文章的主要贡献在于提出了一种简单易于实现版本的模仿学习算法，用于高维、连续、动态环境中。能够很好克服模仿学习中的 `distribution shift` 问题。

背景

模仿学习的问题在于 `behavior shift`，并且误差会累计，因为智能体并不知道如何回

到 expert 的轨迹状态上来。最近做地比较好的就是 GAIL ， GAIL 做模仿学习最大的好处就是 encourage long-horizon imitation 。那为什么 GAIL 能够做到 long-horizon imitation 呢？模型学习一般分为两步，在某个 state 下采取某个 action ，一般的 BC 算法都这么做的，而 GAIL 除此之外还考虑了采取这个 action 之后还回到 expert 轨迹的下一个状态上。而作者也采纳了 GAIL 的上述两点优势，但是并未使用 GAIL 算法中的 adversarial training 技术，而是使用一个 constant reward 。如果matching the demonstrated action in a demonstrated state，reward = +1；对于其他的情况 reward =0。整个问题就变成了一个奖励稀疏的强化学习问题。

所采用的方法？

作者引入 soft-q-learning 算法，将 expert demonstrations 的奖励设置为1，而与环境互动得到的新的 experiences 奖励设置为0。由于 soft Q-Learning 算法是 off-policy 的算法，因此有 data 就可以训练了。整个算法作者命名为 soft Q imitation learning (SQIL)。

Soft Q Imitation Learning算法

SQIL 在 soft q learning 算法上面做了三个小的修正：

1. 用 expert demonstration 初始化填入 agent 的 experience replay buffer ，其 reward 设置为 +1 ；
2. agent 与环境互动得到新的 data 也加入到 experience replay buffer 里面，其 reward 设置为 0 ；
3. 平衡 demonstration experiences 和 new experiences 各 50% 。这个方法在 GAIL 和 adversarial IRL 算法上面也都有应用。

SQIL 算法如下所示：

Algorithm 1 Soft Q Imitation Learning (SQIL)

1: Require $\lambda_{\text{samp}} \in \mathbb{R}_{\geq 0}, \mathcal{D}_{\text{demo}}$
2: Initialize $\mathcal{D}_{\text{samp}} \leftarrow \emptyset$
3: **while** Q_θ not converged **do**
4: $\quad \theta \leftarrow \theta - \eta \nabla_\theta (\delta^2(\mathcal{D}_{\text{demo}}, 1) + \lambda_{\text{samp}} \delta^2(\mathcal{D}_{\text{samp}}, 0))$ {See Equation 1}
5: $\quad$ Sample transition (s, a, s') with imitation policy $\pi(a|s) \propto \exp(Q_\theta(s, a))$
6: $\quad \mathcal{D}_{\text{samp}} \leftarrow \mathcal{D}_{\text{samp}} \cup \{(s, a, s')\}$
7: **end while**

其中 Q_θ 表示的是 soft q function ， $\mathcal{D}_{demo}$ 是 demonstrations ， δ^2 表示的是 soft bellman error 。 Equation 1 表示为：

$$\delta^2(\mathcal{D}, r) \triangleq \frac{1}{|\mathcal{D}|} \sum_{(s,a,s') \in \mathcal{D}} \left(Q_\theta(s,a) - \left(r + \gamma \log \left(\sum_{a' \in \mathcal{A}} \exp(Q_\theta(s', a')) \right) \right) \right)^2$$

其中奖励 r 只有 0 ， 1 两个取值。上述公式的理解就是希望 demonstrated action 能够获得比较高的 Q 值，而周围的 nearby state 的 action 分布就不期望那么突出，期望均匀一点，这里就跟熵联系起来了。

取得的效果？

	Domain Shift ($\mathcal{S}_0^{\text{train}}$)	No Shift ($\mathcal{S}_0^{\text{demo}}$)
Random	-21 ± 56	-68 ± 4
BC	-45 ± 18	$\mathbf{698 \pm 10}$
GAIL-DQL	-97 ± 3	-66 ± 8
SQIL (Ours)	$\mathbf{375 \pm 19}$	$\mathbf{704 \pm 6}$
Expert	480 ± 11	704 ± 79

Figure 1: Average reward on 100 episodes after training. Standard error on three random seeds.

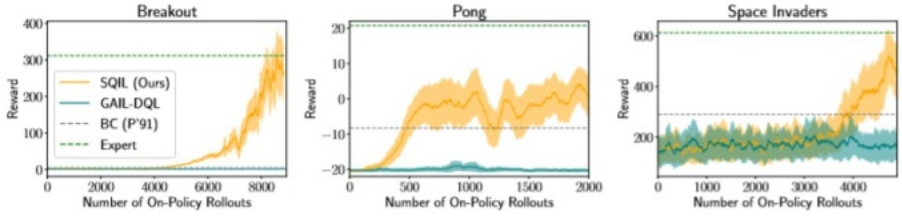

Figure 2: Image-based Atari. Smoothed with a rolling window of 100 episodes. Standard error on three random seeds. X-axis represents amount of interaction with the environment (not expert demonstrations).

所出版信息？作者信息？

作者是来自加利福尼亚伯克利大学的博士生 `Siddharth Reddy` 。

参考链接

- **export-demonstration**：drive.google.com/drive/...

扩展阅读

- **Maximum entropy model of expert behavior**：

`Maximum entropy model of expert behavior`： `SQIL` 是基于最大熵 `expert behavior` 所得出来的算法。策略 π 服从 `Boltzmann distribution`：

$$\pi(a|s) \triangleq \frac{\exp(Q(s,a))}{\sum_{a' \in \mathcal{A}} \exp(Q(s,a'))}$$

`Soft Q values` 可通过 `soft Bellman equation` 得到：

$$Q(s,a) \triangleq R(s,a) + \gamma \mathbb{E}_{s'} \left[\log \left(\sum_{a' \in \mathcal{A}} \exp(Q(s',a')) \right) \right]$$

在我们的模仿学习设置中， `rewards` 和 `dynamic` 是未知的，专家 `demonstration` $\mathcal{D}_{demo}$ 是一个固定的集合。通过在 `environment` 中 `rolling out` 策略 π 可以得到 `state transitions` $(s,a,s') \in \mathcal{D}_{demo}$。

- **Behavioral cloning (BC)**：

在 `behavior clone` 中是去拟合一个参数化的 `model` π_θ，最小化负的 `log-likelihood loss`：

$$\ell_{BC}(\theta) \triangleq \sum_{(s,a) \in \mathcal{D}_{demo}} -\log \pi_\theta(a|s)$$

本文中作者采用的是 `soft q function`，所以最大化的 `likelihood` 目标方程如下所示：

$$\ell_{BC}(\theta) \triangleq \sum_{(s,a) \in \mathcal{D}_{demo}} -\left(Q_\theta(s,a) - \log \left(\sum_{a' \in \mathcal{A}} \exp(Q_\theta(s,a')) \right) \right)$$

从这里可以看出作者的目标函数中相比较于行为克隆算法好处在于：后面那一项基于能量的式子是考虑了 `state transitions`。

- **Regularized Behavior Clone**

`SQIL` 可以看作是 a sparsity(稀疏) prior on the implicitly-represented rewards的行为克隆算法。

Sparsity regularization：当 `agent` 遇见了一个未见过的 `state` 的时候， Q_θ 也许会输出任意值。(Piot et al., 2014) 等人有通过引入a sparsity prior on the implied rewards 的正则化项。

- Bilal Piot, Matthieu Geist, and Olivier Pietquin. **Boosted and reward-regularized classification for apprenticeship learning**. In Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems, pp. 1249–1256. International Foundation for Autonomous Agents and Multiagent Systems, 2014.

作者与上述这篇文章的不同点在于有将其应用于连续的状态空间，还有加了 `latest imitation policy` 进行 `rollouts` 采样。

基于上文的 `soft Bellman equation`

$$Q(s,a) \triangleq R(s,a) + \gamma \mathbb{E}_{s'} \left[\log \left(\sum_{a' \in \mathcal{A}} \exp(Q(s',a')) \right) \right]$$

我们可以得到 `reward` 的表达式子：

$$R_\theta(s,a) \triangleq Q_\theta(s,a) - \gamma \mathbb{E}_{s'} \left[\log \left(\sum_{a' \in \mathcal{A}} \exp(Q_\theta(s',a')) \right) \right]$$

从中也可以发现其会考虑下一个状态 s'，而不像 `BC` 那样只 `maximization action likelihood`。最终的 `Regularized BC` 算法可表示为：

$$\ell_{RBC}(\theta) \triangleq \ell_{BC}(\theta) + \lambda \delta^2 \left(\mathcal{D}_{demo} \cup \mathcal{D}_{samp}, 0 \right)$$

其中 λ 是超参数， δ^2 是 `soft bellman error` 的平方。可以看出 `RBC` 算法与 `SQIL` 有异曲同工之妙。

- **Connection Between SQIL and Regularized Behavioral Clone**

$$\nabla_\theta \mathcal{L}_{\mathrm{RBC}}(\theta) \propto \nabla_\theta \left(\delta^2 \left(\mathcal{D}_{\mathrm{demo}}, 1 \right) + \lambda_{\mathrm{samp}} \delta^2 \left(\mathcal{D}_{\mathrm{samp}}, 0 \right) + V\left(s_0 \right) \right)$$

`SQIL` 相比与 `RBC` 算法引入了 `+1` 和 `0` 的 `reward`，相当于是**加强了奖励稀疏的先验知识**。

发布于 2020-03-14

强化学习 (Reinforcement Learning)　机器学习　深度学习（Deep Learning）

文章被以下专栏收录

强化学习
人工智能；深度强化学习；多智能体；

进入专栏

强化学习前沿
读呀读paper

进入专栏