Confidence-Aware Imitation Learning from Demonstrations with Varying Optimality

关键：

从具有多样最优性的演示中学习的常用方法是将其排序或给予演示置信度，手动输入的置信度阈值限制了性能，因此该文提出自适应的置信度调节置信度阈值。

关键在于能够评估模仿学习训练得到的模型的性能，这可以通过在估计数据集D_E上计算模型的loss得到。

细节

给每个pair<动作，状态>设定置信度，表示该pair出现在优秀策略中的概率，置信度可以理解成函数映射：$\beta : S \times A \rightarrow \mathbb{R}$。目标就是学习到好的$\beta$，从而重新赋予<动作，状态>对的权重，得到新的专家数据集，继续根据loss从新数据中进行模仿学习。

定义最优置信度

根据occupancy measure定义<状态，动作>访问分布：

$$\rho_\pi(s,a) = \pi(a|s) \sum_{t=0}^{\infty} \gamma^t P(s_t = s|\pi) \tag{1}$$

归一化：

$$p_\pi(s,a) = \frac{\rho_\pi(s,a)}{\sum_{s,a} \rho_\pi(s,a)} \tag{2}$$

初始专家演示来源π^d，包括一些最优、次优、失败的演示。根据置信度改变<状态，动作>分布后得到新的演示，可以认为该演示数据来源于π_{new}：$p_{\pi_{new}}(s,a) = \beta(s,a) p_{\pi^d}(s,a)$。目标是找到最优$\beta^*$确保$\pi_{new}$能够最大化return:

$$\beta^*(s,a) = \arg\max_\beta \eta_{\pi_{new}} \tag{3}$$

return被定义为：

$$\eta_\pi = \mathbb{E}_{s_0 \sim \rho_0, \pi}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)] \tag{4}$$

学习置信度

通过Inner Loss和Outer Loss 学习置信度：$\mathcal{L}_{in}$用来鼓励模仿，$\mathcal{L}_{out}$用来评估模仿性能，简单来说有点EM算法的味道：

1. 首先最小化$\mathcal{L}_{in}$得到最优模型θ^*：
$$\theta^*(\beta) = \arg\min_\theta \mathbb{E}_{(s,a) \sim \beta(s,a) p_{\pi^d(s,a)}} \mathcal{L}_{in}(s,a;\theta,\beta) \tag{5}$$
2. 然后最小化$\mathcal{L}_{out}$来优化β:
$$\beta^*_{out} = \arg\min_\beta \mathcal{L}_{out}(\theta^*(\beta)) \tag{6}$$

   $\mathcal{L}_{out}$的计算是在有限数据集D_E上进行估计的。

优化$\mathcal{L}_{in}$和$\mathcal{L}_{out}$

外部优化更新β

令β_τ为时间τ时的置信函数，β要被更新，首先需要确定当前最优θ^*，即先执行内部Loss更新θ，得到θ^*后，执行外部Loss更新即可得到β^*：

- **内部Loss更新θ**：

$$\theta'_{t+1} = \theta'_t - \mu\nabla_{\theta'}\mathcal{L}_{in}(s,a;\theta'_t,\beta_\tau) \tag{7}$$

其中μ为学习率，θ'_t中t表示第t轮更新，$\theta'_0 = \theta_\tau$，只有当上式经过多轮伪更新 得到收敛结果时，θ'才会被真正更新。

- **外部Loss更新β**：

$$\beta_{\tau+1} = \beta_\tau - \alpha\nabla_\beta\mathcal{L}_{out}(\theta') \tag{8}$$

α为学习率。在使用上式更新时，只采样mini-batch用于更新，且β本身也只是二元函数，因此计算复杂度是可接受的。

内部优化更新θ

当得到$\beta_{\tau+1}$后，我们利用内部Loss公式更新θ即可：

$$\theta_{t+1} = \theta_t - \mu\nabla_\theta\mathcal{L}_{in}(s,a;\theta_t,\beta_\tau) \tag{9}$$

同样等到更新收敛时，令$\theta_{\tau+1} = \theta$即可。

上述的两个更新步骤都需要循环梯度下降达到收敛(7式和9式)，这导致计算时间增加，计算效率低。该文提出优化，利用近似值只更新一次：

$$\begin{aligned}\theta'_{\tau+1} &= \theta_\tau - \mu\nabla_\theta\mathcal{L}_{in}(s,a;\theta_\tau,\beta_\tau)\\ \beta_{\tau+1} &= \beta_\tau - \alpha\nabla\mathcal{L}_{out}(\theta'_{\tau+1})\\ \theta_{\tau+1} &= \theta_\tau - \mu\nabla_\theta\mathcal{L}_{in}(s,a;\theta_\tau,\beta_{\tau+1})\end{aligned} \tag{10}$$

具体实现

CAIL框架需要内置模仿算法和外部损失$\mathcal{L}_{out}$，此文采用AIRL作为模仿算法：AIRL由对抗组件G和D组成：

$$\mathcal{L}_{\text{in}}^{D}\left(s,a;\theta^{D},\beta\right) = \mathbb{E}_{(s,a)\sim\beta(s,a)p_{\pi d_{(s,a)}}}[-\log D(s,a)] + \mathbb{E}_{(s,a)\sim\pi_{\theta G}}[-\log\left(1-D(s,a)\right)] \tag{11}$$

$$\mathcal{L}_{\text{in}}^{G}\left(s,a;\theta^{G}\right) = \mathbb{E}_{(s,a)\sim\pi_{\theta G}}[\log D(s,a) - \log\left(1-D(s,a)\right)] \tag{12}$$

其中$\mathcal{L}_{in}^{D}$是针对判别式D的内部损失，$\mathcal{L}_{in}^{G}$是针对生成式G的内部损失，π_{θ_G}是生成式G所表示的策略。D通过最小化$\mathcal{L}_{in}^{D}$来分辨<状态，动作>对是来自专家还是生成式G；G通过最小化$\mathcal{L}_{in}^{G}$来生成与专家数据相似的样本。

对于**外部损失**，AIRL通过判别式D来估计奖励函数：R_{θ^D}，使用$\eta'_{\xi_i} = \sum_{t=0}^{N}\gamma^t R_{\theta^D}(s_t,a_t)$作为预期回报。然后设定由$\eta'_{\xi_i}$产生的排名与真实排名之间不同的损失：

$$\mathcal{L}_{out}(\theta_D) = \sum_i\sum_{j>i} RK\left[\eta'_{\xi_i},\eta'_{\xi_j};\mathbb{I}[\eta_{\xi_i} > \eta_{\xi_j}]\right] \tag{13}$$

其中，当$\eta_{\xi_i} > \eta_{\xi_j}$时$\mathbb{I}[\eta_{\xi_i} > \eta_{\xi_j}] = 1$，否则为-1。RK被定义为margin=0的margin ranking loss：

$$\text{RK}\left[\eta'_{\xi_i};\eta'_{\xi_j},\eta_{\xi_i},\eta_{\xi_j}\right] = \begin{cases}\max\left(0,-\mathbb{I}\left[\eta_{\xi_i} > \eta_{\xi_j}\right]\left(\eta'_{\xi_i} - \eta'_{\xi_j}\right)\right), & \left|\eta'_{\xi_i} - \eta'_{\xi_j}\right| > \epsilon\\ \max\left(0,\frac{1}{4\epsilon}\left(\mathbb{I}\left[\eta_{\xi_i} > \eta_{\xi_j}\right]\left(\eta'_{\xi_i} - \eta'_{\xi_j}\right) - \epsilon\right)^2\right), & \left|\eta'_{\xi_i} - \eta'_{\xi_j}\right| \le \epsilon\end{cases} \tag{14}$$

作者在ϵ的范围内修正了$(\eta'_{\xi_i} - \eta'_{\xi_j}) = 0$的情况从而保证Lipschit光滑。