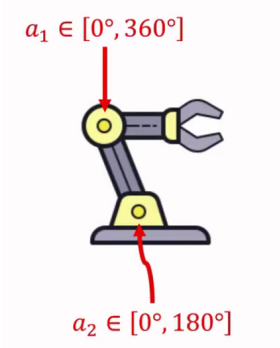
DPG确定性策略梯度

问题介绍

DPG模型用来解决连续性控制问题，比如一个这样的问题：



有一个机器臂，它有两个关节，上面的关节能够选择0~360度，下面的关节能够旋转0~180度。所以它有两个变量，自由度等于2，动作是一个2为向量，动作空间是一个连续的集合，有无穷多个动作

确定性Actor-Critic

DPG模型通过Actor-Critic算法建立，有一个策略网络控制智能体做运动，它根据状态s进行决策a。价值网络不控制智能体，它只是基于状态s对动作a进行打分，从而指导策略网络进行改进

不过对于该策略网络是确定性的函数，记作 **a = π(s;θ)**，θ为神经网络的参数，它的输入是状态s，不过它的输出不是状态分布，而是一个具体的动作a

给定状态s，输出的动作a是确定的，没有随机性。这就是名字中为什么被称为**确定性策略**，该网络输出的动作a可以是实数也可以是向量。

以机器手臂为例，该问题策略网络的输出是一个二维向量，二维指的是自由度为2，并不是指动作空间就两个动作。动作空间是无穷的，**注意离散动作的区别**

价值网络又称Critic，记作**q(s,a;w)**，其中w为价值网络的参数。价值网络有两个输入，一个状态s，另一个是动作a，基于状态s价值网络评价动作a的好坏程度。价值网络的输出是一个实数，它是对动作好坏的评估，动作越好，这个输出的实数就越大。

我们要训练两个神经网络，让神经网络共同进步，让决策越来越好，让评估打分越来越准确

DPG模型训练

TD更新价值网络：

每次观测到一个Transition(s_t，a_t，r_t，s_{t+1})，让价值网络预测**t**时刻的价值，记作$\mathbf{q_t(s_t,a_t;w)}$

在让价值网络预测**t+1**时刻的价值：$\mathbf{q_{t+1}(s_{t+1},a'_{t+1};w)}$，我们知道这个动作$a'_{t+1} = \pi(s_{t+1};\theta)$是由策略网络决策的，并不是智能体真正执行的动作。$a'_{t+1}$只是用来更新价值网络而已

用以上两个值计算时序差分TD有：

$$\text{TD error: } \delta_t = q_t - (r_t + \gamma \cdot q_{t+1}).$$

后面减掉的部分为TD目标，一部分是真实观测到的奖励$\mathbf{r_t}$，另一部分网络自己做出的预测$\mathbf{q_{t+1}}$

所以预测TD目标，比单纯的预测q_t更接近真相。所以鼓励qt更接近TD Target，即——**使TD error尽量小**

通过梯度下降来更新网络参数**w**：

$$\text{Update: } \mathbf{w} \leftarrow \mathbf{w} - \alpha \cdot \delta_t \cdot \frac{\partial\, q(s_t, a_t; \mathbf{w})}{\partial \mathbf{w}}.$$

- Loss: $L = \frac{1}{2}(q - y)^2.$
- Gradient: $\frac{\partial\, L}{\partial\, \mathbf{w}} = \frac{\partial\, q}{\partial\, \mathbf{w}} \cdot \frac{\partial\, L}{\partial\, q} = (q - y) \cdot \frac{\partial\, Q(\mathbf{w})}{\partial\, \mathbf{w}}.$

即，让价值网络的预测更接近TD Target

DPG更新策略网络：

确定性策略梯度和之前学习的策略梯度不太一样，下面对其进行推导

训练策略网络，需要价值网络的帮忙，价值网络用来评估动作a的好坏，从而指导策略网络进行改进

策略网络的参数为θ，θ越好则说明决策越正确，输出的动作a就更好。由于策略网络自己不知道动作的好坏，好坏取决于价值网络的评价，价值网络的输出越大就意味着动作越好。

所以目标清晰了，我们通过改变策略网络的θ使得价值网络的输出越大越好

对于确定性策略，对于某个鼓捣的状态s就会输出确定的动作a，所以s确定，价值网络也确定，那么唯一影响价值网络的输出的因素就是策略网络的参数θ，我们想要更新θ使得Q值变大，所以我们只需计算Q关于θ的梯度，然后做梯度上升，这样就可以使价值Q变大，这个梯度就称为确定性策略梯度**DPG**：

$$\text{DPG: } \mathbf{g} = \frac{\partial\, q(s, \pi(s;\boldsymbol{\theta}); \mathbf{w})}{\partial\, \boldsymbol{\theta}} = \frac{\partial\, a}{\partial\, \boldsymbol{\theta}} \cdot \frac{\partial\, q(s, a; \mathbf{w})}{\partial\, a}.$$

通过链式法则可计算，本质上就是**让梯度从价值q传播到动作a，然后从a传播到策略网络。**这样计算的梯度就是确定性策略梯度，用它来更新策略网络，然后做梯度上升来跟新θ：

$$\text{Gradient ascent: } \boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \beta \cdot \mathbf{g}.$$

g为刚刚计算出的梯度，β为学习率，这样更新θ可以使价值Q变大，也就是说这样就能使策略网络变得更好

DDPG算法

DPG模型改进

DDPG，顾名思义，它是一个深度的DPG模型（深度确定性策略梯度）

在实际应用中，这种带有神经函数近似器的actor-critic方法在面对有挑战性的问题时是不稳定的，这是很正常的事情。原因在于比如训练策略的θ时，它沿着某个方向去走梯度这个的**长短**，其实也会被Q值左右，而Q值在学习时，假设Q模型的参数为φ的话，这个参数φ只是竭尽全力的去逼近它的**TD**目标（时序差分），它不会关注我们的动作a对应点的斜率是否非常非常大，但在神经网络中，这种斜率是有可能非常大的。在Q学习更新的过程中，它根本不会去关注这些东西，所以导致学习的梯度非常大，因此不稳定

所以对于DDPG，深度确定性策略梯度为了解决这个问题，它分别从4个方面在DPG（确定性策略梯度）上进行相应的改善：

- 经验重放（离线策略）

第一个方面就是经验重放，对于要训练当前神经网络的DPG模型，刚刚采样到的数据是肯定不够的，肯定是需要把以前甚至很久之前采样的数据拿出来使用。而DPG学习只需要把当前状态s采样出来即可，接下来的动作或者价值都是不需要历史的数据的，所以只需要对状态进行采样。利用经验重放所以影响真的不大

- 目标网络

第二点就是使用了目标网络，在学习DQN的时候，如果将学习好的网络即放在目标上，又不断更新，就会非常的不稳定，DDPG中同样的道理，我们在策略模型网络和Q值模型网络上都去建立**另外一份拷贝**，去做目标网络。这样就能保证在学习中，目标这一部分是稳定的。

- 在动作输入前批标准化Q网络

在输入前对数据做相应的正则化，使得网络在学习中，每次输入都有一个**正定的分布**。使得过拟合这种情况就不太会出现

- 添加连续噪音

为了环境训练不稳定而必须要添加的东西，在某个点的周围加入噪音，帮助网络在某个点能够进行平滑的估计

通过这四点改进，就能对DPG模型有了本质的提升