分布式强化学习入门

强化学习入门自用教程

强化学习

本文的主要学习资料，主要依赖于CSDN博客及Bilibili的WANG SHUSEN教学视频。参考的书籍为《动手学强化学习》

强化学习基础篇

强化学习综述

一、 Value-Based方法

Value-Based方法是指通过估计状态值来选择动作，其核心思想是通过估计状态值来选择动作。

DQN算法

DQN算法是Deep Q-Learning的简称，是DeepMind提出的一种基于深度学习的强化学习算法。DQN算法的核心思想是通过神经网络来学习一个Q函数，通过Q函数来选择下一个动作。DQN算法的核心思想是通过神经网络来学习一个Q函数，通过Q函数来选择下一个动作。

CSDN中DQN算法的介绍

DQN算法

DDQN算法

二、 Policy-Based方法

Policy-Based方法是指通过学习一个策略来选择动作，其核心思想是通过学习一个策略来选择动作。

policy gradient算法

policy gradient算法是一种基于策略梯度的强化学习算法。策略梯度算法的核心思想是通过策略来评估一个状态下的所有可能的动作，然后根据策略来选择一个最优的动作。

CSDN中policy gradient算法的介绍

policy gradient算法

Bilibili中policy gradient算法的介绍

policy gradient算法

Value-Based 与 Policy-Based 区别

value-based是根据计算出来的期望reward，选择最大的期望reward所对应的action。典型代表Q-Learning。policy-based是将计算出来的期望reward当作选择action的概率，期望的reward越大，对应的action被选中的概率也就越大，但不一定就会被选中，只是概率。典型代表Policy Gradient。

CSDN中对于两者关系的详细讲解

一文带你理清从Q-Learning到DDPG(Deep Deterministic Policy Gradient)算法思想

三、 On-Policy 与 Off-Policy 区别

1. On-Policy

“要训练的Agent” 跟 “和环境互动的Agent” 是同一个的话，这个叫做 on-policy。即：要训练的 Agent，它是一边跟环境互动，一边做学习这个叫 on-policy。比如：Policy Gradient。Policy Gradient 是一个会花很多时间来取样的算法。Policy Gradient 算法的大多数时间都在取样，Agent/Actor跟环境做互动取样后update参数一次(只能 update 参数一次)，接下来就要重新再去环境里取样，然后才能再次 update 参数一次，这非常花时间。

2. Off-Policy

“要训练的Agent” 跟 “和环境互动的Agent” 不是同一个的话，这个叫做 off-policy。如果它是在旁边看别人玩，透过看别人玩，来学习的话，这个叫做 Off-policy。

在迭代的过程中允许两个policy，π_θ与π'_θ。其中，π'_θ一个用于学习过程的动作，具有很强的探索性，另外一个π_θ是由值函数产生的最优策略，这个方法就叫做off-policy。

从on-policy到off-policy的好处就是：用π'_θ去跟环境互动，用π'_θ采集到数据去训练π_θ

用π_θ从环境中取样去训练π_θ意味着$\pi_{\theta'}$从环境中回去的样可以用多次，π_θ在做梯度上升时可以执行多次，比较有效率。

CSDN中对于PPO的讲解

PPO(Proximal Policy Optimization)

** 需要关注里面的PPO2算法。 **

四、 Actor-Critic方法

Bilibili讲解视频

讲解笔记

1. State_Value Function Approximate

State-Value Function 用来估计一个状态的价值，它是通过估计状态值来选择动作。

$$V_\pi = \sum_a \pi(a|s) * Q_\pi(s,a) \approx \sum_a \pi(a|s;\theta) * q(s,a;w)$$

** Policy network(actor):

利用神经网络$\pi(a|s;\theta)$去近似$\pi(a|s)$,其中θ为神经网络参数。

** Value network(critic):
利用神经网络$q(s,a;w)$去近似$Q_\pi(s,a)$,其中w为神经网络参数。

价值网络不控制agent运动，只负责给行为打分。类似于裁判给运动员打分；

2. Train the networks

利用神经网络近似得到的State-Value Fucntion结果如下：

$$V(s;\theta,w)=\sum_a \pi(a|s;\theta)*q(s,a;w)$$

** 训练：更新参数θ和w,
a. 更新策略网络参数θ目的是为了提升state-value $V(s;\theta,w)$

- 目的是为了让actor表现的更好
- 分数是由value network(critic)去评估的

b. 更新参数w是为了更好去评估真实的return

- 目的是为了让critic更接近正确的return
- 通过环境给的奖励去更新自己

训练网络的流程：

$$V(s;\theta,w)=\sum_a \pi(a|s;\theta)*q(s,a;w)$$

Training:Update the parameters θ and w

- i. 观测当前状态s_t
- ii. 根据$\pi(\cdot|s_t;\theta_t)$随机选择a_t
- iii. 执行a_t得到r_t和下一个状态s_{t+1}
- iv. 利用$\pi(\cdot|s_t;\theta_t)$求出$\tilde{a}_{t+1}$(但不执行)
- v. 算两次价值网络的输出：$q_t=q(s_t,a_t;w)$与$q_{t+1}=q(s_{t+1},\tilde{a}_{t+1};w_t)$
- vi. 计算TD error: $\delta_t=q_t-(r_t+\gamma*q_{t+1})$
  $q_t-(r_t+\gamma*q_{t+1})$这里作为base line
- vii. 对价值网络进行求导：$d_{w,t}=\frac{\partial q(s_t,s_t;w)}{\partial w}|_{w=w_t}$
- viii. 利用TD算法更新价值网络参数$w_{t+1}=w_t-\alpha*\delta_t*d_{w,t}$
- ix. 对策略函数进行求导：$d_{\theta,t}=\frac{\partial log\pi(a_t|s_t,\theta)}{\partial \theta}|_{\theta=\theta_t}$
- x. 利用梯度上升来更新策略网络：$\theta_{t+1}=\theta_t+\beta*q_t*d_{\theta,t}$
  q_t为裁判打的分数；一部分论文中会用δ_t来代替q_t

3. Actor-Critic方法总结

a. Policy Network and Value Network

目标(最大):

$$V_\pi(s) = \sum_a \pi(a|s) * q(s,a)$$

利用神经网络进行近似:

$$\pi(a|s;\theta) -> \pi(a|s)[actor]$$

$$q(s,a;w) -> Q_\pi(s,a)[critic]$$

b. During training

$$a_t = \pi(\cdot|s_t;\theta_t)$$

价值网络q(critic)作为裁判，给运动员打分；

c.After training

$$a_t = \pi(\cdot|s_t;\theta_t)$$

价值网络q(critic)此时不发挥作用。

备注（课后思考题）

基于值的好，还是基于策略的好?

可以简化模型时，利用值，利用Q-TABLE.