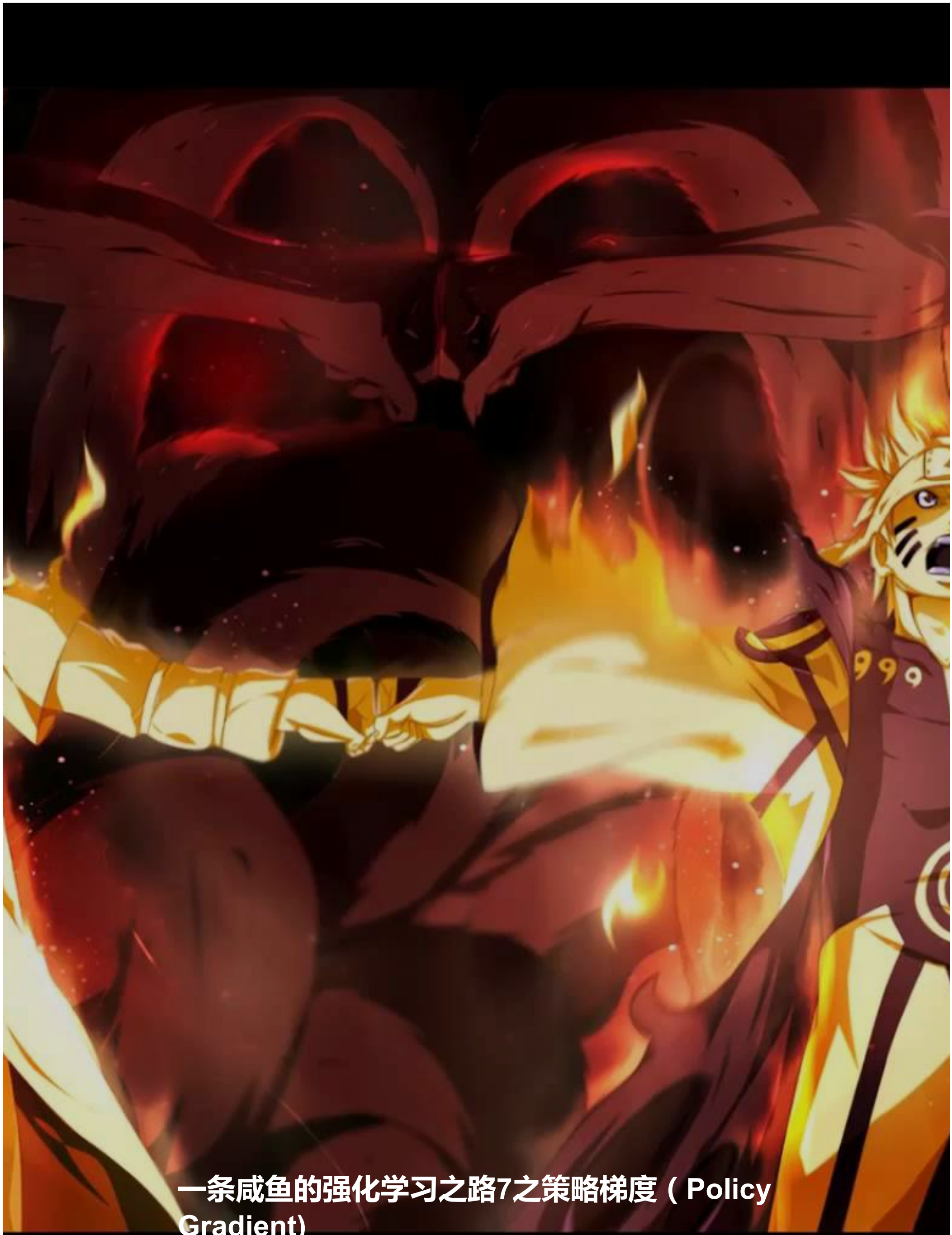
知

写文章 ooo



一条咸鱼的强化学习之路7之策略梯度（ Policy Gradient)

贾航天 · 4 天前

一条咸鱼的强化学习之路7

策略梯度（Policy Gradient）

Hello,大家好，距离上次发心得好像已经过了有一个月的时间了，这个月的时间呢主要是在琢磨DQN，尝试通过例子（Flappybird,Atari 的Tennis）去实现和理解算法。Flappybird用Deepmind论文《Playing Atari with Deep Reinforcement Learning, 2013》，《Human-level Control through Deep Reinforcement Learning, 2015》里网络结构和预处理方法能得到相当不错的结果，差不多训练二百多万次飞越七八十根柱子很随意，毕竟满打满算也就向上向下两个动作维度。然而事情到Atari的Tennis上就变得不是那么乐观了，Tennis是Atari的一款网球游戏，如下图所示：

还是有那么几分对抗性质的，有兴趣的童鞋可以去搜一下模拟机尝试玩一下，说实话，并不容易，我玩了几把被机器完虐...囧。这里边你控制的agent的动作维度有18个！18个！18个！

（分别是none，上，下，左，右，左上，右上，左下，右下，再分别在这些方位上加一个击打的动作）所以呢，可想而知，其实这基本上已经可以看成是一个有点小连续性质的控制问题了，用起DQN来确实有点吃力。而通过看上边human-level control 文章的附录2，即DQN玩各种游戏的一个比分与人类玩家的对比，你会发现一个有趣的现象，如下图红线标注所示，这几个游戏大部分是动作维度较高的对抗操控游戏，虽然计分规则不是非常清楚，但和其他动辄几千上万的score相比，一个负分放在那会至少让人感觉表现不佳，但即便这样，有些还是完爆所谓“职业游戏玩家”。

Alien	227.8	939.2	103.2	6875	3069 (±1093)	42.7%
Amidar	5.8	103.4	183.6	1676	739.5 (±3024)	43.9%
Assault	222.4	628	537	1496	3359(±775)	246.2%
Asterix	210	987.3	1332	8503	6012 (±1744)	70.0%
Asteroids	719.1	907.3	89	13157	1629 (±542)	7.3%
Atlantis	12850	62687	852.9	29028	85641(±17600)	449.9%
Bank Heist	14.2	190.8	67.4	734.4	429.7 (±650)	57.7%
Battle Zone	2360	15820	16.2	37800	26300 (±7725)	67.6%
Beam Rider	363.9	929.4	1743	5775	6846 (±1619)	119.8%
Bowling	23.1	43.9	36.4	154.8	42.4 (±88)	14.7%
Boxing	0.1	44	9.8	4.3	71.8 (±8.4)	1707.9%
Breakout	1.7	5.2	6.1	31.8	401.2 (±26.9)	1327.2%
Centipede	2091	8803	4647	11963	8309(±5237)	63.0%
Chopper Command	811	1582	16.9	9882	6687 (±2916)	64.8%
Crazy Climber	10781	23411	149.8	35411	114103 (±22797)	419.5%
Demon Attack	152.1	520.5	0	3401	9711 (±2406)	294.2%
Double Dunk	-18.6	-13.1	-16	-15.5	-18.1 (±2.6)	17.1%
Enduro	0	129.1	159.4	309.6	301.8 (±24.6)	97.5%
Fishing Derby	-91.7	-89.5	-85.1	5.5	-0.8 (±19.0)	93.5%
Freeway	0	19.1	19.7	29.6	30.3 (±0.7)	102.4%
Frostbite	65.2	216.9	180.9	4335	328.3 (±250.5)	6.2%
Gopher	257.6	1288	2368	2321	8520 (±3279)	400.4%
Gravitar	173	387.7	429	2672	306.7 (±223.9)	5.3%
H.E.R.O.	1027	6459	7295	25763	19950 (±158)	76.5%
Ice Hockey	-11.2	-9.5	-3.2	0.9	-1.6 (±2.5)	79.3%
James Bond	29	202.8	354.1	406.7	576.7 (±175.5)	145.0%
Kangaroo	52	1622	8.8	3035	6740 (±2959)	224.2%
Krull	1598	3372	3341	2395	3805 (±1033)	277.0%
Kung-Fu Master	258.5	19544	29151	22736	23270 (±5955)	102.4%
Montezuma's Revenge	0	10.7	259	4367	0 (±0)	0.0%
Ms. Pacman	307.3	1692	1227	15693	2311(±525)	13.0%
Name This Game	2292	2500	2247	4076	7257 (±547)	278.3%
Pong	-20.7	-19	-17.4	9.3	18.9 (±1.3)	132.0%
Private Eye	24.9	684.3	86	69571	1788 (±5473)	2.5%
Q*Bert	163.9	613.5	960.3	13455	10596 (±3294)	78.5%
River Raid	1339	1904	2650	13513	8316 (±1049)	57.3%
Road Runner	11.5	67.7	89.1	7845	18257 (±4268)	232.9%
Robotank	2.2	28.7	12.4	11.9	51.6 (±4.7)	509.0%
Seaquest	68.4	664.8	675.5	20182	5286(±1310)	25.9%
Space Invaders	148	250.1	267.9	1652	1976 (±893)	121.5%
Star Gunner	664	1070	9.4	10250	57997 (±3152)	598.1%
Tennis	-23.8	-0.1	0	-8.9	-2.5 (±1.9)	143.2%
Time Pilot	3568	3741	24.9	5925	5947 (±1600)	100.9%
Tutankham	11.4	114.3	98.2	167.6	186.7 (±41.9)	112.2%
Up and Down	533.4	3533	2449	9082	8456 (±3162)	92.7%
Venture	0	66	0.6	1188	380.0 (±238.6)	32.0%
Video Pinball	16257	16871	19761	17298	42684 (±16287)	2539.4%
Wizard of Wor	563.5	1981	36.9	4757	3393 (±2019)	67.5%
Zaxxon	32.5	3365	21.4	9173	4977 (±1235)	54.1%

Best Linear Learner is the best result obtained by a linear function approximator on different types of hand designed features[23]. Contingency (SARSA) agent figures are the results obtained in ref. 15. Note the figures in the last column indicate the performance of DQN relative to the human games tester, expressed as a percentage, that is, 100 × (DQN score − random play score)/(human score − random play score).

Anyway, 说这些，无非也就是想表述一下不同算法都有其优劣势，DQN在低维离散动作空间的表现还是可圈可点的，而要向着高维和连续控制问题进军，可能就有点力不从心了，毕竟遍历空间会指数级的增加，最后陷到某个local optimum胡同里可能就出不来了或者也许根本就找不到好么……

对了，在正式侃策略梯度问题之前，这个月也在琢磨着怎么样去把监督式学习和强化学习进行结合。毕竟，在我看来，监督式的学习代表着对人类经验的一个学习与理解，而强化学习本是弱监督式的，充满创新的存在，通过trail and error形式不断去计算探索最优的解决方法，这两个结合的话会颇有一种站在巨人的肩膀上的感觉，不是么？感觉比较好的一个办法就是监督式学习的权重直接作为强化学习的一个初始值，再在这个基础上进行探索吧～也就是Alphago 1.0版本采取的思路，先用三千万盘走棋先训练一个策略网络，然后在此基础上自我对弈强化，再通过MCTS将另外一个价值网络与策略网络进行整合，最终输出结果。这些事情其实也只是说起来容易，监督学习训练本身就不是一个一蹴而就的事情，像我这种咸鱼，收集了游戏数据进行训练，效果也是差得要命，数据本来就是个头疼事，调参就更蛋疼了，逼急了我就去上天桥——找CNN代调参的，5块一层的话也费不了几个钱嘛→ _→….其实群里有几个下棋的大神已经搞定了监督学习，各种砍瓜切菜，但强化嘛，嘻嘻嘻，挣扎在通往AGI的康庄大道上，共勉哈～**@强化学习小分队**

好了，接下来我应该要好好琢磨琢磨policy gradient了，毕竟在实际高维连续场景下梯度的运用还是更靠谱一点，我们还从DS的第七节课件谈起～

David Sivler

我们知道，强化学习基本上可以分为基于值函数的（Value based）和基于策略的（Policy based），当然还有进一步两者结合的（Actor Critic），前边我们比较熟悉的DQN就是基于值函数的，数值什么的相比较于梯度什么的而言当然是更加直观和容易理解，但离散的数据在去尝试解决高维和连续动作时候就显得吃力了，这时候也就到了使用梯度概念的时候，刚开始听上去也许蛮神秘，但其实也就像是其描述的那样，我们要直接将我们的策略函数参数化，即πθ(s,a) = P [a | s, θ]，通过求导，然后在某个梯度方向上一点点不断地更新并最终达到最优。所以，和value based方法相比，策略梯度的收敛性会好一点，不会持续震荡，但同时容易陷到局部最优里边，并且单纯的策略梯度方法会比较慢，哦不...很慢，非常慢...

我们的目标肯定是要找到最优策略，所以我们一般会通过什么标准来对我们现在的策略函数进行评估比较？DS课程罗列了三种情况，第一是在episode环境下，可以比较从一个状态开始到episode结束所获得的奖励；第二是在连续环境下，可以计算平均奖励，即用某个状态的概率乘以从该状态起获得的奖励；第三是在以time step计算的环境下，可以用某状态的概率乘以此时某个动作的概率再乘以获得的瞬时奖励。这三种方法其实本质含义是一致的。

比如游戏一局就是一个episode

说完评估，我们再看如何优化的问题，毕竟我们引入策略函数的参数θ后就是希望看看怎么通过搞一搞θ来找到策略函数的最大值。课程里提到两大方法，其一是不用梯度概念的，如什么hill climbing，遗传算法等，第二种是要引入梯度概念的，如**梯度下降**，联合梯度等，一般情况下用梯度概念的效率会更高，我们这里也主要关注梯度下降方法（Gradient descent）。其实这个名词大家一定比较熟悉了，在神经网络里边进行参数的更新用的也是梯度下降的方法，我感觉这里边思想都是一致的，只不过我们的目标函数不一样，DNN里边我们要找到合适参数使得loss function最小，而这里我们要使policy function 最大，所以求得了梯度以后，前者要顺着向下找最小（descend），而后者要顺着梯度向上找最大（ascend）。具体一下，在求得n个方向上的梯度∇θJ(θ)后，我们就来一步一个脚印的来用公式Δθ= α∇θJ(θ)尽情朝着梯度最大的方向前进吧（a：step size）！

喂。。。等等，你说啥？让我在n个方向上都分别计算梯度比较么？疯了么？要n无穷大岂不是作死？恩，方法虽然是个笨方法，但对于任意的策略都可行，甚至有些不可微情况下依然坚挺，但我们毕竟还有更好的方法，对吧？是的，我们接下来尝试分析式来计算策略的梯度，也就是具体怎么确定梯度更新方向的问题。首先，我们先引入两个概念，第一是**似然比**（Likelihood ratios），我们可以通过利用似然比来计算策略梯度，这里有一个利用对数导数的性质对策略梯度进行的巧妙转换，∇θπθ(s,a) = πθ(s, a) *[∇θπθ(s, a)/ πθ(s, a)] = πθ(s, a)∇θ logπθ(s, a)，而这个∇θlog πθ(s, a)我们称之为**得分函数**（Score function），这是第二个概念。在有了这些概念后，DS课程里边提到了两种策略，即Softmax Policy和Gaussian Policy。其中，Softmax policy 常应用于离散动作，其采取某行动的概率和用一些特征组成的线性组合算出的指数值成比例。其Scorefunction表示的是[(实际执行的动作的特征)–(我们可能才采取的所有动作的特征的平均值)]，即如果某个feature出现的频率较高并且能得到较高的reward，那我们就调整策略来更多的执行这个动作。而Gaussian Policy则主要用于连续的动作空间，其均值就是状态特征的线性结合，score function表示的含义是我采取的某个动作比平时多做了些什么，如果效果好的话就朝着该方向进行更新。

接下来课程又从one-step MDP过程的策略梯度扩展到了策略梯度理论：

Theorem

For any differentiable policy $\pi_\theta(s,a)$,
for any of the policy objective functions $J = J_1, J_{avR}$, *or* $\frac{1}{1-\gamma}J_{avV}$,
the policy gradient is

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\ Q^{\pi_\theta}(s,a)\right]$$

上式就是用求score function和long term reward的乘积的期望，并以此来调整策略。这样也就产生了Monte-Carlo Policy Gradient即最简单的REINFORCE算法：

function REINFORCE
Initialise θ arbitrarily
for each episode $\{s_1, a_1, r_2, ..., s_{T-1}, a_{T-1}, r_T\} \sim \pi_\theta$ **do**
for $t = 1$ to $T - 1$ **do**
$\theta \leftarrow \theta + \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) v_t$
end for
end for
return θ
end function

但是这种方法真的很慢，并且方差很大。下面我们要做的事情就是如何让它更好更有效。我们考虑加进来一个Critic，即一个对策略进行评估的价值函数Qw(s,a)。这也就是传说中的AC算法，其中C代表Critic，也就是价值，用以更新行动价值函数的参数；A代表Actor，也就是策略，来向价值指向的方向更新策略的参数。所以这就又涉及到了之前几章的有关价值函数的计算问题，我们可以用MC，TD(0)，TD(λ)进行计算。下边是基于action-value critic的简单QAC算法：

function QAC
Initialise s, θ
Sample $a \sim \pi_\theta$
for each step **do**
Sample reward $r = \mathcal{R}_s^a$; sample transition $s' \sim \mathcal{P}_{s,\cdot}^a$
Sample action $a' \sim \pi_\theta(s', a')$
$\delta = r + \gamma Q_w(s', a') - Q_w(s, a)$
$\theta = \theta + \alpha \nabla_\theta \log \pi_\theta(s, a) Q_w(s, a)$
$w \leftarrow w + \beta \delta \phi(s, a)$
$a \leftarrow a', s \leftarrow s'$
end for
end function

注意，这里边这个α还是step size，step size决定了你更新的暴力程度，如果你的α相当的大，那其实也有几分greedy的意味了，所以policy和value之间的壁垒实际上也没有概念上相差的那么大。

话又说回来，上面的简单QAC算法还是有不足的，比如，方差可能还会非常之大，我们有没有什么方法来减小方差呢？答案是肯定的，我们可以引入一个叫基准线（Baseline）的概念，它只减小方差而不会改变期望。一个常用的baseline就是状态价值函数Vπθ(s)，如果我们把critic里减去状态价值函数，那么我们就可以了解到采取一个行动会比常规的能好多少。继而可以促使我们能一直朝着比平时表现更好的方向去更新策略参数。这个好的部分我们就称为优势（Advantage）。事实证明，采用Advantage可以significantly减小策略梯度的方差。这里要提一下，可以用TD error来表示Advantage，因为他们形式上是一致的...可喜可贺啊...这样一来，我们的critic参数减少到了一个。

这节课的内容差不多也就这么多了，总结下，我们首先要考虑计算策略梯度Actor，你可以用score function乘以奖励；你也可以用score function乘以价值函数/Advantage；你还可以考虑用资格迹来积累得分而不仅仅是使用当前得分，我们也可以在不同的时间步上积累资格迹然后用其来代替当前得分等。至于Critic，还是我们之前学的MC和TD。然后就是不管用什么AC方法，我们的目标都是让策略向能获得更多价值的方向进行移动。

写的貌似挺乱……凑乎看吧……最后，打个广告，对强化学习感兴趣童鞋来加Q群595176373！

咸鱼天~

aHR0cDovL3FtLnFxLmNvbS9jZ2ktYmluL3FtL3FyP2s9NFRjSERyYll5Tk81TzN1WHNMenA1NmczWnUwM1dhcWs= (二维码自动识别)

强化学习 (Reinforcement Learning)

咸鱼 梯度下降

还没有评论