Atari-lunar-lander

看到代码训练了800 episode 实现了 平均得分200。  
请问每个episode大约多久，总计几个小时？  
  
我是每episode 1.6分钟，跑了半天300回合，平均值-100的样子，  
实在太久跑不动了

然后，这个tiled离散化对于训练速度，影响不大

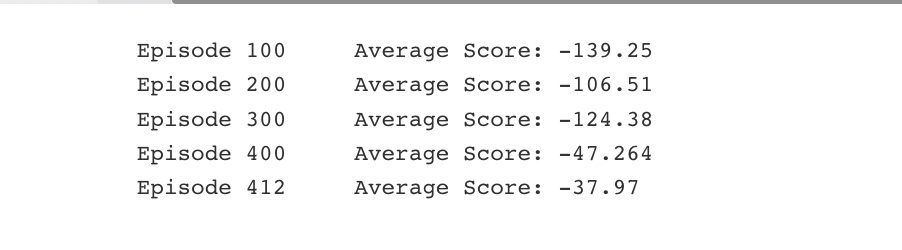
@丁涛(D.T)你说lecture4的问题吗？具体时间我忘记了 当时应该是好几个小时、

3070卡，16G

你需要在本地设置 remote connect

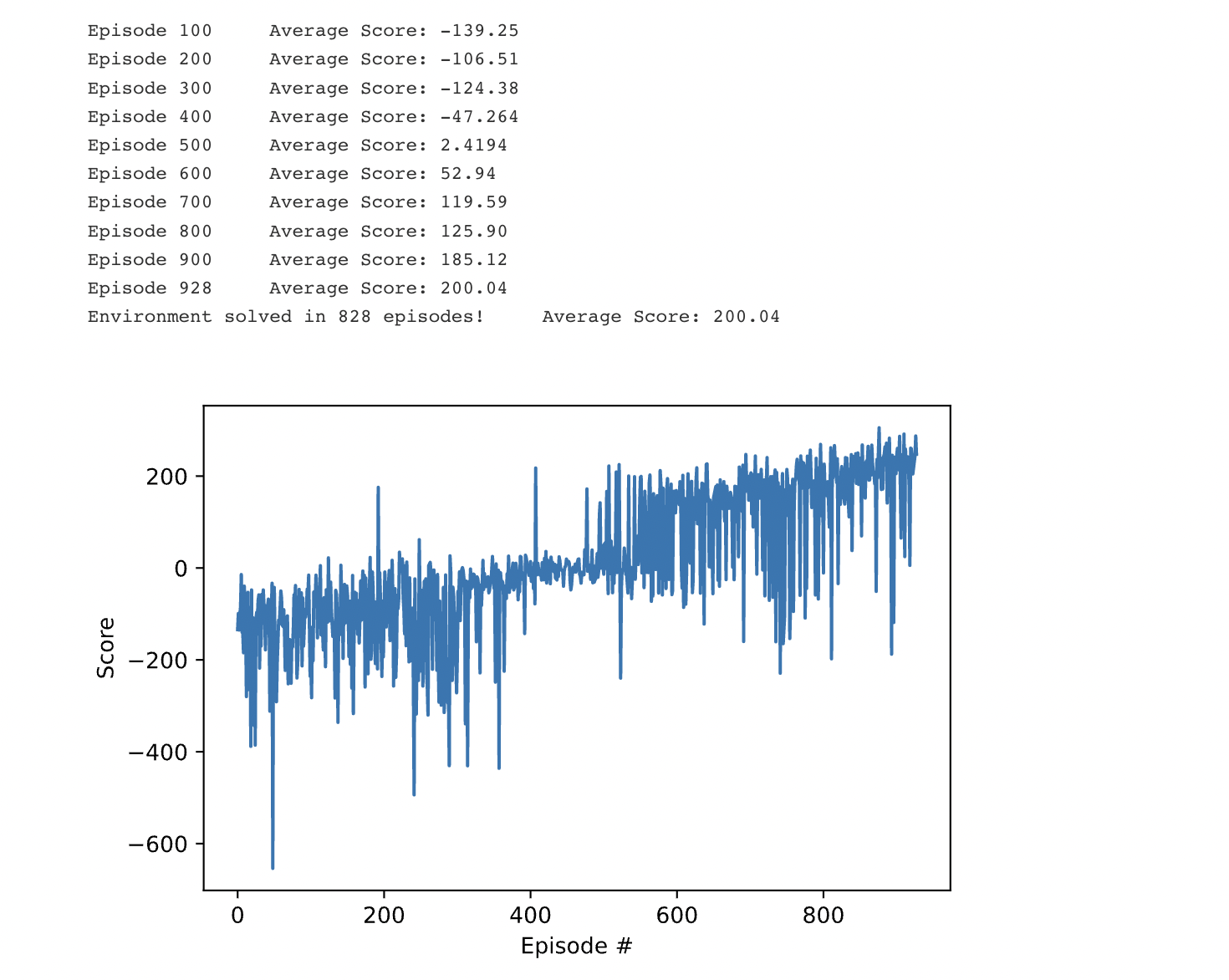
然后通过IP地址，连接到远程主机即可

我一个episode 不到1秒

训练到后面会变慢，原因是说明这个“agent” 可以进行更长时间的运行了，steps更长了

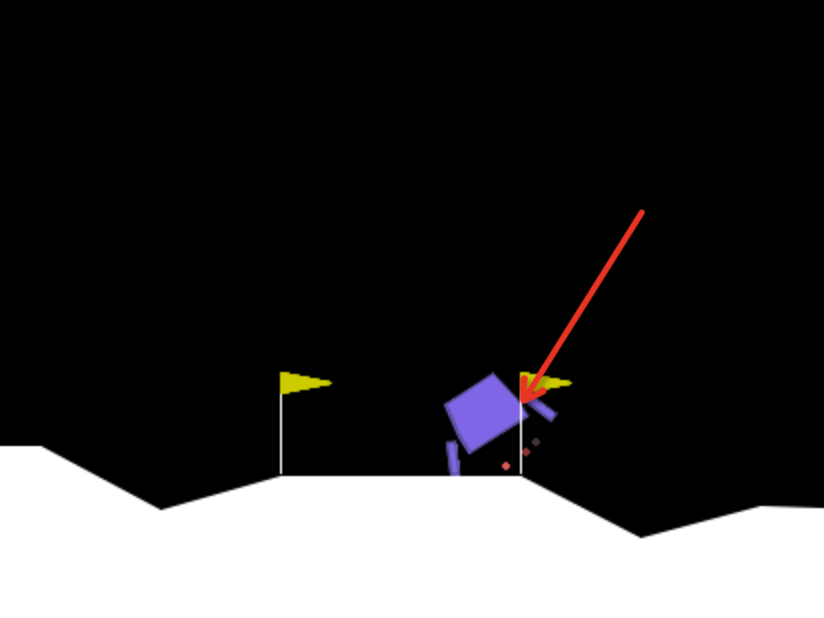
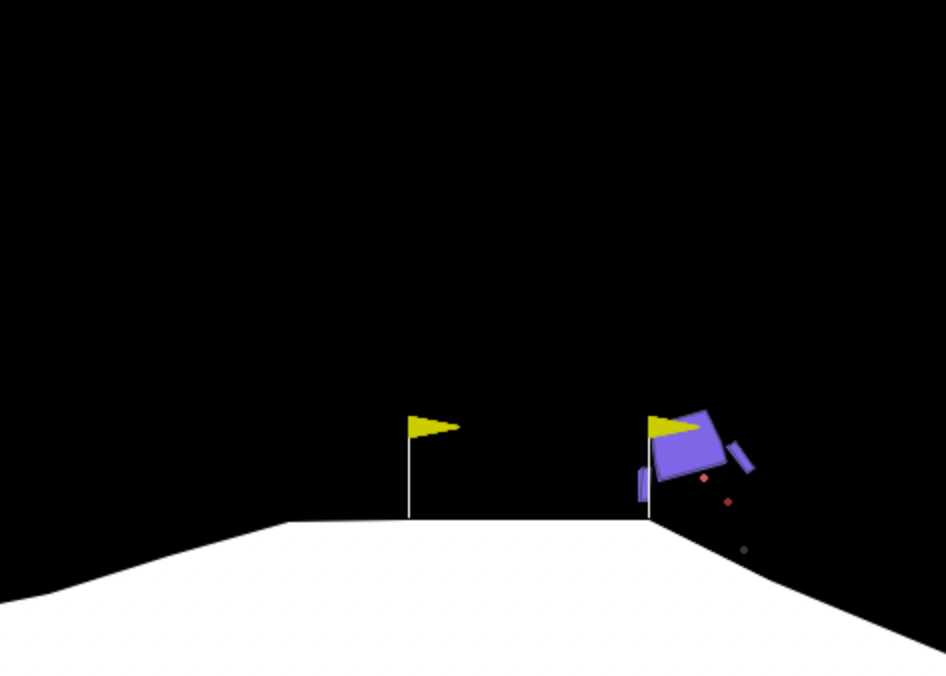
这个是很正常的 如果你等不住，可以设置一个最大长度的step

这个专业术语叫做 “horizon”



@高民权(Minchiuan)那是我代码可能有问题，我需要再重头到位再看下代码。  
我一开始也快的，前面50个episode都是秒结束，后面是平均1.6分钟。  
训练的结果我放到gym环境下render,显示是月球车盘旋，保持稳定，不落地，直到时间耗尽

@丁涛(D.T)应该是这样的 最终它会落地, 但是落地的瞬间就截止了, 所以给人造成了一种不落地的感觉

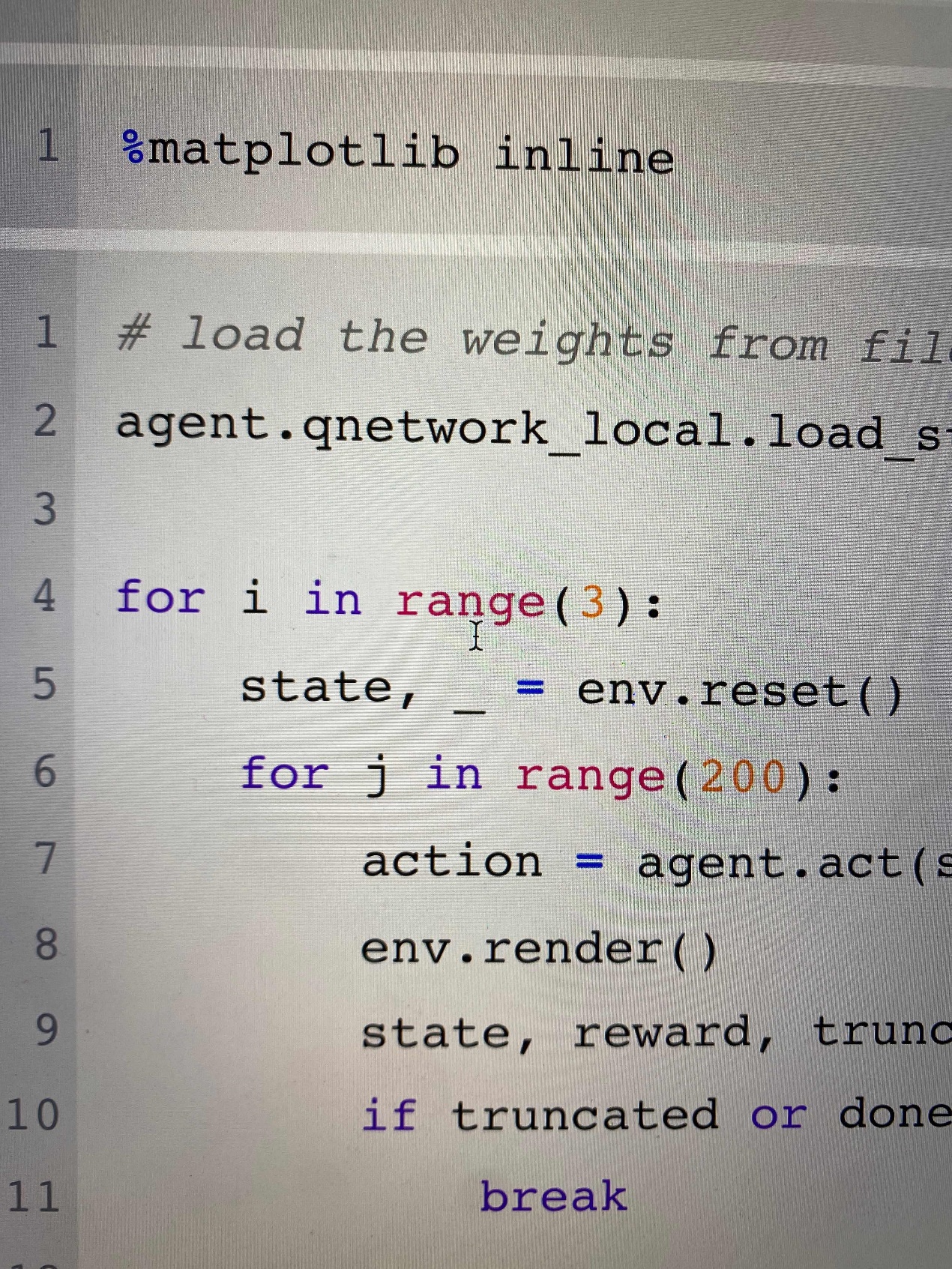


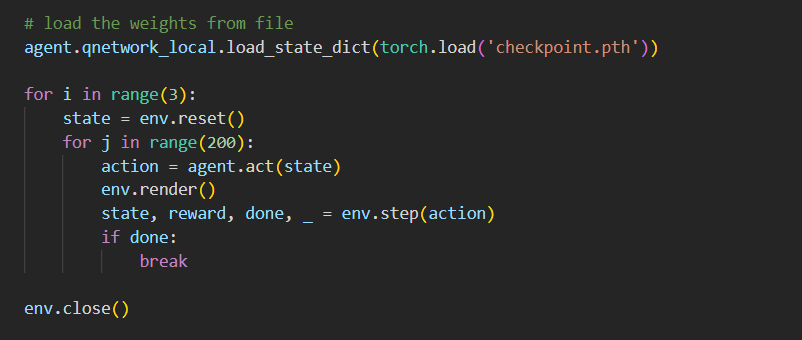
这两张图,似乎都失败了? (登陆在两旗帜中间,并双腿着陆?)

有成功的，多跑几次，把3（后面验证的次数）改成10，并不是每次都成功，

 平均分200分,肯定有成功的了; 我的平均分达不到这水平

我的代码估计有重大问题,里面





act不就是决策动作吗

step就是把动作放到env里面执行啊

**课程录播：https://shanji.dingtalk.com/app/transcribes/763275696438303139323931355f3136323835343832315f30/80192915/0?from=4**

**课程代码：**

<https://github.com/computing-intelligence/RL-2023-Spring/tree/main/RL-2023-Course-Code/lesson-05-deep-q-learning>

**课后练习：**

我们在课程上已经讲解了DQN的原理，以及我们为大家提供了一份包含了neural network， DQN-Agent的代码，该代码没有CNN模块，解决的是月球车登录问题，请**修改源代码，**加入CNN等DQN-Paper中提到的方法，解决Atari问题。

**课程重点记录：**

Deep-Q-Learning 是一个具有历史意义的算法，该算法让我们从原始的tabular（表格方式）进化到了使用deep learning深度学习来进行RL问题求解。

**使用监督学习解决DeepQLearning方法一直有两个问题：**

1. RL问题中，随着Q的更新，agent观察到的episode变化太大，导致学习难以收敛，或者不稳定；
2. RL在转化为监督学习的时候，我们要拟合的函数是Q(s, a)，我们使用的target是： r + max\_{a'}Q(s', a') ，target中的Q同样也在不断更新，这就导致我们学习的时候，目标是在持续变化。所以也导致训练的不稳定性；

**DeepQLearning如何解决：**

1. Experience Replay 通过将 (s, a, s', r)的tuple对进行存储，并且在训练的时候随机sampling，打乱了之前的correlation；
2. Fixed Target，通过固定Target的参数特定的次数，可以使得训练过程更加稳定。
   1. 在实际代码中，有人提出了soft-update这种方法，每一次都更新target参数，但是只更新“一点点”，同样也实现了类似的功能。

**DeepQLearning的Loss函数：**

L\_i\left(\theta\_i\right)=\mathbb{E}\_{\left(s, a, r, s^{\prime}\right) \sim U(D)}\left[\left(r+\gamma \max \_{a^{\prime}} Q\left(s^{\prime}, a^{\prime} ; \theta\_i^{-}\right)-Q\left(s, a ; \theta\_i\right)\right)^2\right]

其中：

* \left(s, a, r, s^{\prime}\right) \sim U(D) 表示训练experience的随机sampling；
* \max \_{a^{\prime}} Q\left(s^{\prime}, a^{\prime} ; \theta\_i^{-}\right)值得是Target函数，该函数的参数theta只在特定的C steps后依此时训练得出的Q-network的参数值进行更新；

**一些实现细节：**

1. 图像进行了预处理，将大图片压缩为小图片，并且进行了黑白处理，降低输入大小；
2. 输入为最近的m个时刻，例如m=4，则输入为连续的4个时刻的“连续段时刻”数据；
3. 代码中 deep-q-learning，输出为一个 |action| 长度的向量，每个值是此action的向量，所以这个生成方法可以既得出Q-value，又得出我们该使用哪个动作；

