强化学习调整收敛

v1.0(~2019.7.5)

数据

tensor: [2, 18]

length = 18 = GeneratorsNum(8) + LoadsNum(10)

channel = 2 = [Pg, Qg]

state: $Pg, Qg \in [0, 10]$

action: 只调整发电机的有功或者无功，调整幅度为[-2, -1, -0.5, -0.1, 0.1, 0.5, 1, 2]， 8 * 2 * 2 * 4 = 32

reward:收敛:1， 越界:-1; 其它:-0.01

DQN网络

(2,32,5) conv1d

(32,64,3) conv1d

(64,64,3) conv1d

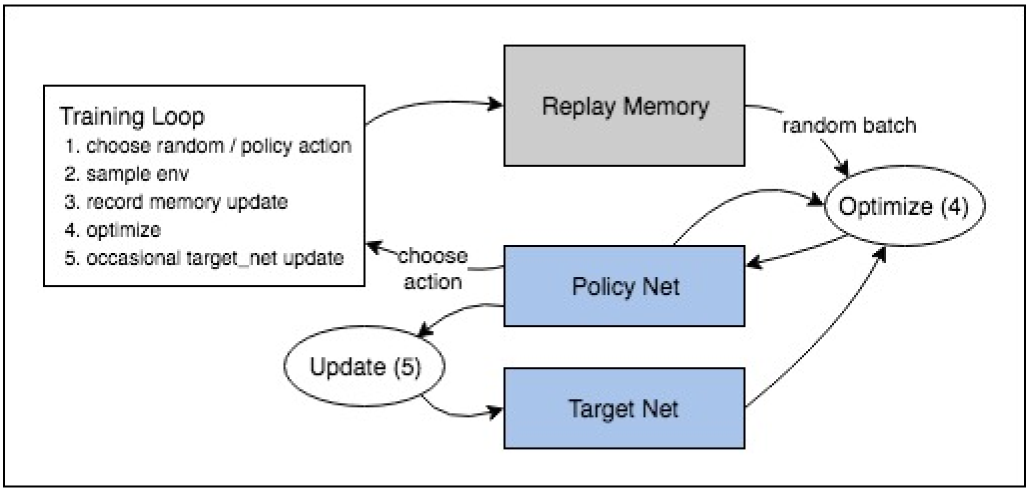
==>output(64,10)

(640, 512) fc

(512, actions_num) fc

输入为*state*，输出为当前*state*下*action*对应的分数

训练



Replay Memory:存放着2个数据池，一个是收敛的样本，一个是不收敛的样本，最开始没有收敛的样本，当开始有收敛样本之后，会在random batch中保证存在收敛的样本，两者数量都是随机的，总和为BATCH_SIZE

Policy Net & Target Net:都是DQN网络，Policy Net输出当前的*(state,action)*得到的*reward*，Target Net得到当前*state*下所有的*action*的*reward*，选取此时*reward*的最大值，两者对比得到loss.

$$loss = Q(s,a) - (r + \gamma max_a Q(s',a))$$

固定步数之后，更新Targe Net = Policy Net. 逐步收敛得到真实的Q表(DQN网络)

参数

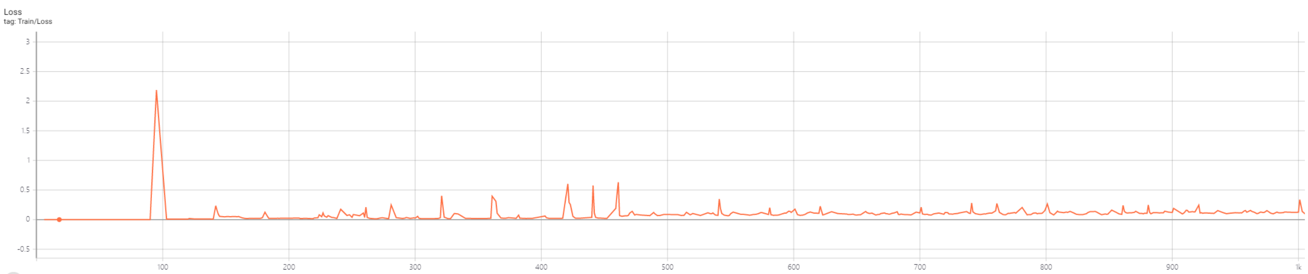
```
- EPOCHS = 10000  //1000次reward = 1 / -1
- BATCH_SIZE = 512
- GAMMA = 0.999
- EPS_START = 0.9 //开始随机探索的概率
- EPS_END = 0.05  //最后随机探索的概率
- EPS_DECAY = 1000 //1000次探索之后，以0.1的概率随机探索
- TARGET_UPDATE = 20 //20个EPOCH后，更新target_net
- optimizer = RMSprop(default)
- lossFuc = smooth_l1_loss
- lr = 1e-2

```

训练结果

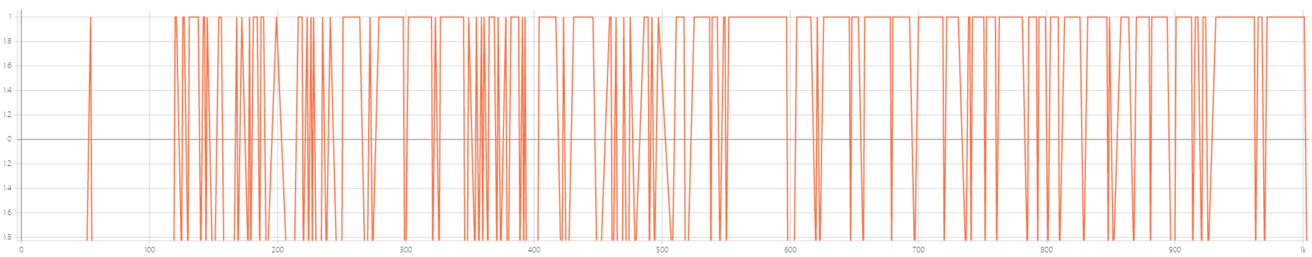
Epoch = 1000

loss:



收敛的很快，说明policy net和target net差距不大

Reward:



120多轮过后收敛，基本上都能潮流收敛，不收敛的可能原因是随机探索导致的

Epoch = 2000

loss: