强化学习：作业二

丁云翔 191250026

December 3, 2021

1 作业内容

在gridworld环境中实现Q-learning算法。

2 实现过程

在main.py和algo.py中补全了Q-Learning的相关代码，其中算法主体位于algo.py中，具体代码如下

```
class MyQAgent(QAgent):
    def __init__(self):
        super().__init__()
        self.learningRate = 0.01
        self.discountFactor = 1
        self.qTable = defaultdict(lambda: [0.0, 0.0, 0.0, 0.0])

    def select_action(self, ob):
        stateQ = self.qTable[str(ob)]
        max = []
        maxValue = stateQ[0]
        max.append(0)
        for i in range(1, 4):
            if stateQ[i] > maxValue:
                max.clear()
                maxValue = stateQ[i]
                max.append(i)
            elif stateQ[i] == maxValue:
                max.append(i)
        return random.choice(max)

    def learn(self, ob, action, reward, nextOb):
        oldQ = self.qTable[str(ob)][action]
        newQ = reward + self.discountFactor * max(self.qTable[str(nextOb)])
        self.qTable[str(ob)][action] += self.learningRate * (newQ - oldQ)
```

MyQAgent类即为我实现的算法，其中__init__函数中初始化了算法的参数，包括学习率，折扣因子和Q值表格；select_action函数则是根据传入的状态

返回根据当前Q值表格得到的当前状态下的最优动作，如有多个最优动作则在最优动作中随机选取一个；learn函数则是根据刚刚运行的结果依照Q(s,a)的更新公式对Q值表格进行更新。
在main.py中，修改的地方主要有三处，第一处是用MyQAgent替换QAgent

```
    # agent initial
    # you should finish your agent with QAgent
    # e.g. agent = myQAgent()
    # agent = QAgent()
    agent = MyQAgent()
```

第二处是修改epsilon，使用epsilon随轮数增加递减的策略，使得前期注重探索（偏向于选择envs.action_sample()），后期注重利用（偏向于选择agent.select_action(obs)），希望在有限地轮数内最优化算法表现。第三处是在调用envs.step()函数后，根据返回的结果调用agent.learn()函数，以更新Q值矩阵。

```
# start to train your agent
for i in range(num_updates):
    # an example of interacting with the environment
    obs = envs.reset()
    for step in range(args.num_steps):
        # Sample actions with epsilon greedy policy
        epsilon = 1 - i / 100.0
        if np.random.rand() < epsilon:
            action = envs.action_sample()
        else:
            action = agent.select_action(obs)

        # interact with the environment
        obs_next, reward, done, info = envs.step(action)
        agent.learn(obs, action, reward, obs_next)
        obs = obs_next
        if done:
            envs.reset()
```

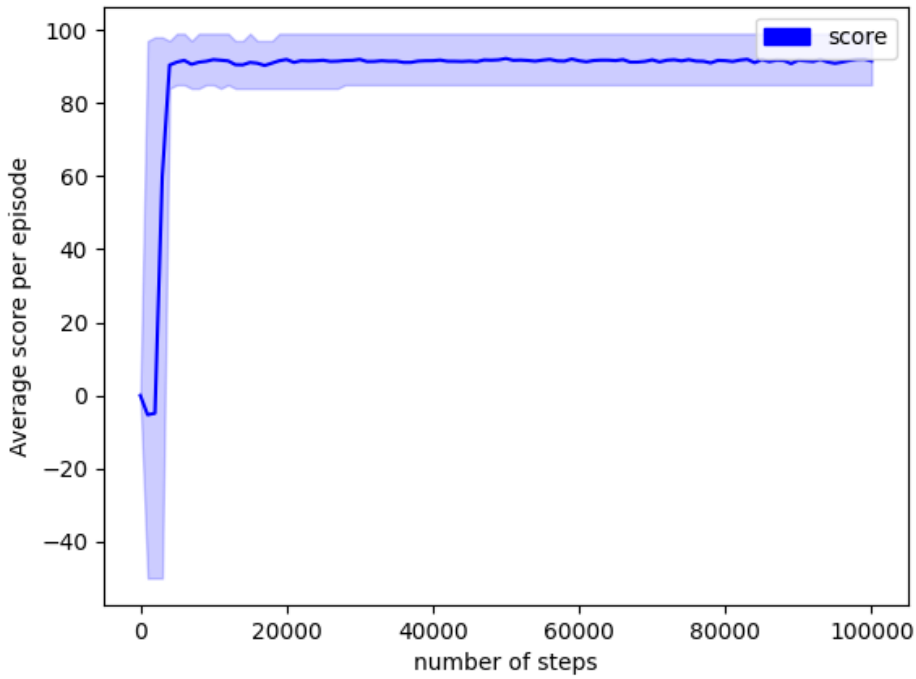
另外绘图时未用到query这个变量，将与其有关的内容删去，此处不赘述。

3 复现方式

在主文件夹（code）下运行 `python main.py`.

4 实验效果

通过调整epsilon、学习率和折扣因子，进行了多次实验，找到了一个相对较优的参数组合，最后找到的最优结果如下：
采用随轮数增加递减的epsilon，以在前期偏向探索、后期偏向利用，比固定epsilon值效果更佳。又尝试了几组学习率和折扣因子的参数组合，发现学习率设为0.5，折扣因子设为0.8（可以使步数更少的通关轨迹更优）时效果较好。性能图如下：



得分均值（即累计奖励）总体随训练轮数（即样本量）增加而增大，以上述参数组合的结果为例，大约在50轮左右的更新后达到最优，得分均值约为91.5，之后在上下很小的范围浮动。

5 小结

在这次实验中，我发现Q-Learning实现起来并不复杂，尤其是这次的地图相对而言比较简单，状态数不算多，算法的效果也很好，收敛比较快，但是参数的调整确实有一些技巧在里面，想选取到合适的参数组合并不容易。