第三次编程作业报告

——李昭阳 2021013445

一、 问题建模

1. 状态空间：

a) 31 个普通状态二维坐标
b) 4 个陷阱状态二维坐标
c) 1 个奖励状态二维坐标

2. 行动集合：{上、下、左、右}

3. 状态转移概率：

由于$p_{ss'} = P(S_{t+1} = s'|S_t = s)$且$p^a_{ss'} = 1$，因此认为$p_{ss'} = \pi(a|s)$，即 s 状态下选取 a 动作的概率。

4. 回报设计：

a) 行动鼓励：每次转移到不同的位置奖励 1 分
b) 距离惩罚：惩罚动作后位置与奖励状态的曼哈顿距离
c) 结束奖励：若掉入陷阱惩罚-100 分，得到糖果奖励 100 分

二、 DQN 网络设计

1. 算法原理：

Algorithm 1 Deep Q-learning with Experience Replay

Initialize replay memory $\mathcal{D}$ to capacity N
Initialize action-value function Q with random weights
for episode $= 1, M$ **do**
  Initialise sequence $s_1 = \{x_1\}$ and preprocessed sequenced $\phi_1 = \phi(s_1)$
  for $t = 1, T$ **do**
    With probability ϵ select a random action a_t
    otherwise select $a_t = \max_a Q^*(\phi(s_t), a; \theta)$
    Execute action a_t in emulator and observe reward r_t and image x_{t+1}
    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $\mathcal{D}$
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from $\mathcal{D}$
    Set $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma \max_{a'} Q(\phi_{j+1}, a'; \theta) & \text{for non-terminal } \phi_{j+1} \end{cases}$
    Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ according to equation 3
  end for
end for

算法维护了一个策略网络，一个目标网络。目标网络通过神经网络模拟 Q 表，策略网络通过与目标网络贝尔曼方程所得结果进行均方误差 Loss 求解，不断更新自身参数，在一定步长后整体更新目标网络。

2. 算法效果：

算法采用ϵ-贪心策略，选取如下超参数，得到较好训练效果。

```
episodes = 500
max_steps = 50

n_states = 2
n_actions = env.n_actions
n_hidden = 128

capacity = 500  # 经验池容量
lr = 2e-3
gamma = 0.9

initial_epsilon = 0.1
final_epsilon = 0.9
epsilon_increment = 0.005

target_update = 200  # 目标网络的参数的更新频率
batch_size = 32
min_size = 200  # 经验池超过200后再训练
```

训练后智能体采取如下策略进行移动，易知其选取了其中一条最优路径。