

**课 程 实 验 报 告**

**课程名称： 人工智能导论**

**专业班级： CS2003班**

**学 号： U202015375**

**姓 名： 汪宇飞**

**指导教师： 金 燕**

**报告日期： 2021.12.28**

**计算机科学与技术学院**

# Q-Learning算法走迷宫知识背景

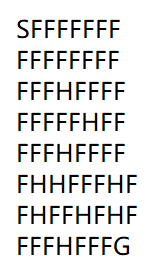
本次课程作业我选取的题目是Q-Learning 实现走迷宫算法。显而易见，本次课程作业的核心内容是Q-Learning算法。Q-Learning是强化学习算法中value-based的算法，算法的主要思想是将State与Action构建成一张Q-table来存储Q值，然后根据Q值来选取能够获得最大的收益的动作。其实现原理在报告第二部分会详细介绍。

Q-Learning算法是由Watkins于1989年在其博士论文中提出，是强化学习发展的里程碑，也是目前应用最为广泛的强化学习算法。Q- Learning算法目前主要应用于动态系统、机器人控制、工厂中学习最优操作工序以及学习棋类对弈等领域。

本次课程作业提供的环境是gym.openai网站所提供的FrozenLake8x8。问题描述如下：

*“冬天来了。你和你的朋友在公园里扔飞盘时，你使劲地投掷飞盘，把飞盘留在了湖中央。湖水大部分是冰冻的，但有几个洞的冰已经融化了。如果你踏入其中一个洞，你会掉进冰冷的水中。此时飞盘短缺，因此必须穿越湖泊并取回飞盘。但是冰很滑，所以你不会总是朝着你想要的方向移动。”*

它给出了一个如图1-1所示的大小为8x8的湖面，也即迷宫。其中“S”代表“start”，表示出发点；“F”代表“frozen surface”，表示安全地点；“H”代表“hole”，表示危险地点；“G”代表“goal”，表示目的地即终点。要求从起点S处出发，路中不经过H而只走F，最终到达终点G。



**图1-1 FrozenLake8x8迷宫**

本次课程作业将通过实现强化学习Q-Learning算法，来达成成功走迷宫到达终点。

# Q-Learning算法原理以及相关步骤

## Q-Learning算法基本思想

Q-Learning算法是一种off-policy的强化学习算法，一种典型的与模型无关的算法。算法通过每一步进行的价值来进行下一步的动作。基于Q-Learning算法智能体可以在不知道整体环境的情况下，仅通过当前状态对下一步做出判断。

Q-Learning是强化学习算法中value-based的算法，Q是指在某一时刻的某一状态下采取某一动作期望获得的收益。环境会根据智能体的动作反馈相 应的回报，所以算法的主要思想就是将状态与动作构建成一张Q值表，然后根据Q值来选取能够获得最大的收益的动作。

## Q-Learning算法原理

### Q表和R表

Q表是由Q(s,a)组成的一个矩阵，记录着状态动作价值函数。其中Q(s,a)是状态动作价值函数，是在状态s时采取动作a之后，可以获得的奖励的期望值。Q(s,a)越大表示在agent在看到状态s是采取动作a比较好。Q表的形式如表2-1所示。

表2-1 Q表形式

|  |  |  |  |
| --- | --- | --- | --- |
| Q表 | A1 | A2 | A3 |
| S1 | Q(A1,S1) | Q(A2,S1) | Q(A3,S1) |
| S2 | Q(A1,S2) | Q(A2,S2) | Q(A3,S2) |
| S3 | Q(A1,S3) | Q(A2,S3) | Q(A3,S3) |

R表与Q表类似，以状态为行、以动作为列，记录着reward值。

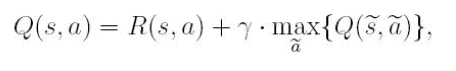
在初始时由于agent对外界一无所知，所以Q表初始化时全部元素应为0。

### 根据Q表进行决策

在某一时刻，根据目前Q表进行决策选择动作。根据Q表的性质可以得出决策方法：首先需要检测当前环境所处于的状态Si，确定状态Si之后在Q表中找到这一状态所处于的行，然后在这一行中对给动作的Q（Si，Aj）进行比较并找到最大值，那么此时决策选取的动作即为这一最大值所对应的Aj。

### Q表的更新

在Q-Learning算法中，对于Q表的更新是重要部分。更新公式为



其中γ为学习参数，是大于等于0且小于1的常数。γ趋向于0表示主要考虑immediate reward，γ趋向于1表示主要考虑future reward。Q表在s达到目标状态时停止更新，此时也即算法结束。

## Q-Learning算法步骤

根据2中的Q-Learning算法原理。可得出算法步骤如图2-1所示。



图2-1 Q-Learning算法步骤

首先初始化Q表，创建一个8x8的矩阵并将其数据均初始化为0，然后在当前状态s的所有可能行为中选取一个行为a，由此得到下一格状态，并计算回报更新Q表，此时判断s是否已经到达最终状态，若已到达则结束，否则再次选择行为，如此循环下去。

## Q-Learning算法实现走迷宫

根据上述算法描述并结合gym给出的库可以得出使用Q-Learning算法走迷宫的步骤，如图2-2所示。



图2-2 使用Q-Learning算法走迷宫步骤

# 实验结果及分析

## 实验结果

按照上述的算法步骤，将Q-Learning训练次数设置为1500次，得出结果如图3-1所示。

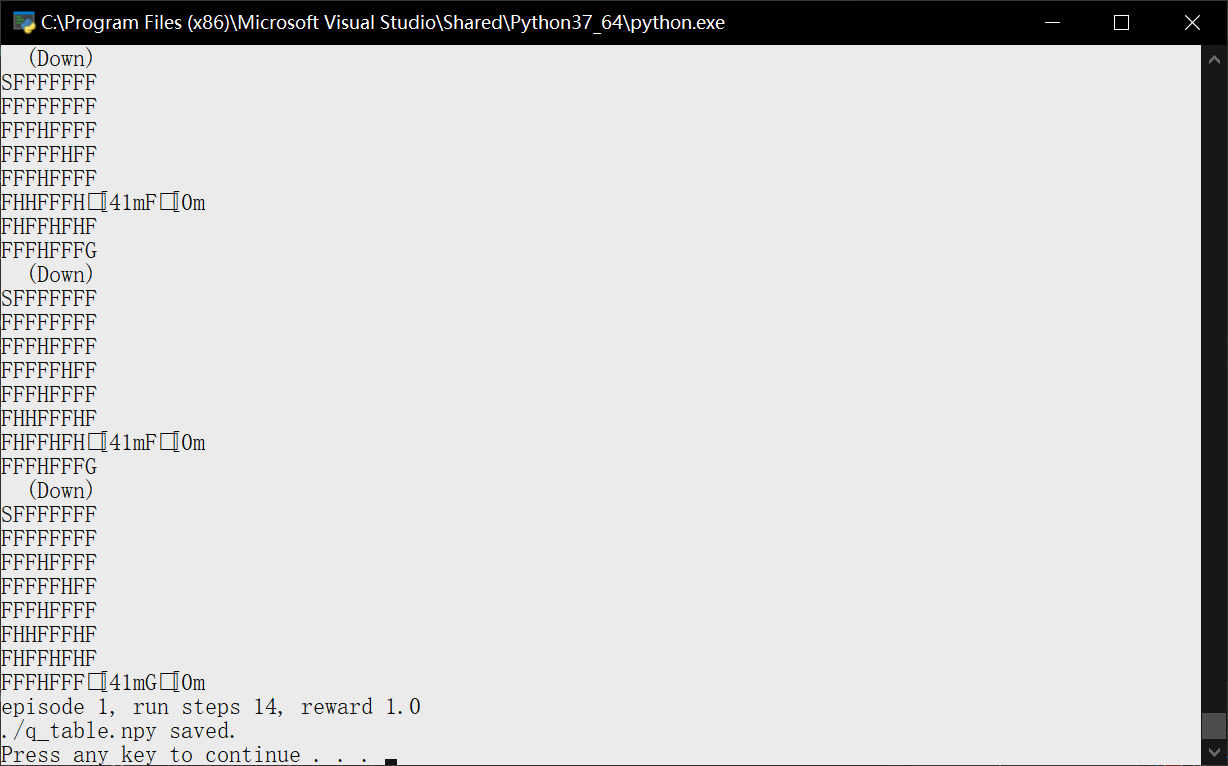


图3-1 训练1500次后测试结果

可见总共走了14步到达终点，reward为1.0，成功达成了Q-Leaning算法实现FrozenLake8x8的目标。

## 问题分析

1. 冰面打滑问题

根据gym所述，冰面是光滑的，因此每次移动并不一定会按agent所选取的动作来进行，即发生“打滑”现象，这会严重影响Q表的更新，对Q-Learning造成较大影响。因此为避免这一情况的发生，将冰面设置为不打滑的，从而使Q表的更新按照设想的动作进行，减少影响，提高算法学习效率。

1. 局部最优问题

Q-Leaning算法与贪心算法有共通之处，对于一个较大规模的问题，如果每次都是仅仅按当前状态的最优解来行动则有可能会陷入在某一部分中的死循环中，从而导致Q-Learning算法陷入死循环而导致错误。由于Q表的决策与Q（S，A）有关，而Q（S，A）的值时与Q表的更新有关，因此对于Q表更新时学习参数γ的选取很重要。如果γ过于大，则极有可能造成所述的局部最优的问题，因此此次实验中对于学习参数γ选取为0.01，以放缓学习速度为代价来避免局部最优问题的出现。

1. 训练次数问题

由于FrozenLake8x8是属于8x8的较大迷宫，在较少训练次数的情况下Q表可能尚未更新成熟。在每次均未调用已保存的Q表文件而是重新开始的情况下，训练次数不同的测试结果如下。

* 1. 训练次数500次

可见训练500次Q表不够，未能成功到达终点。

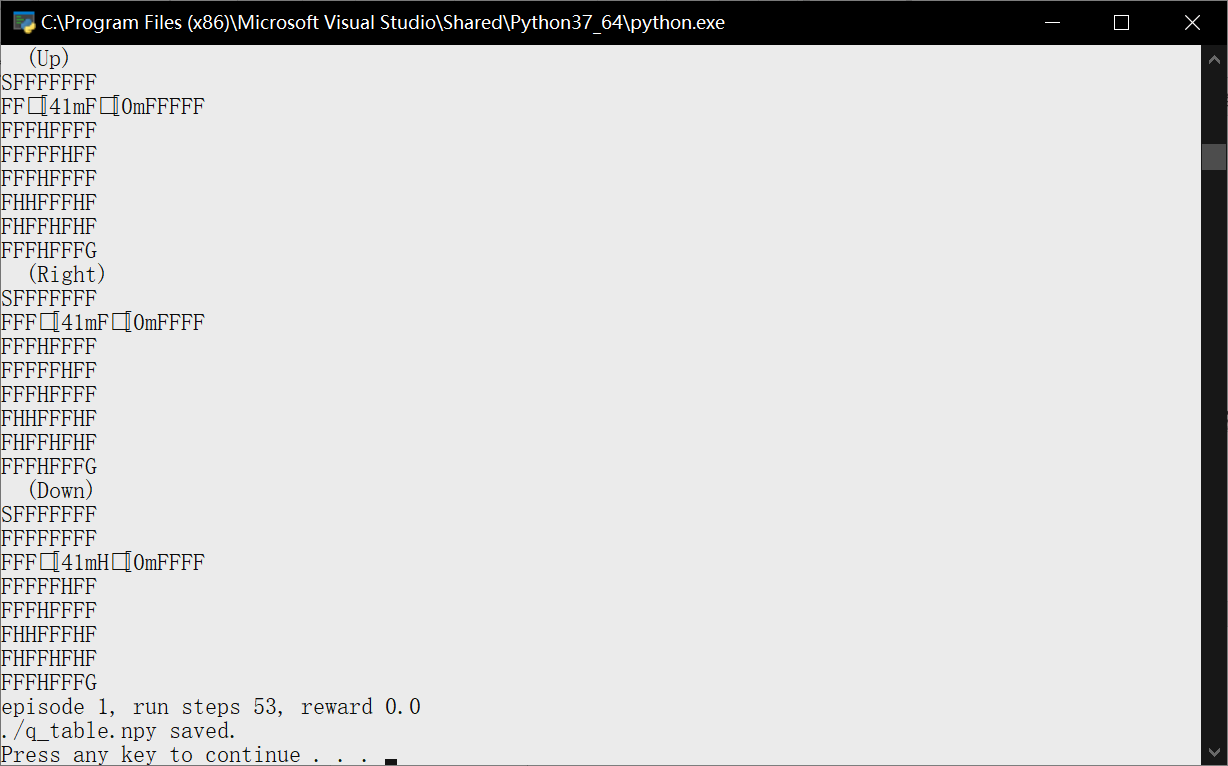


图3-3 训练次数500次的测试结果

* 1. 训练次数800次

可见训练800次Q表仍然不够，还是未能成功到达终点。

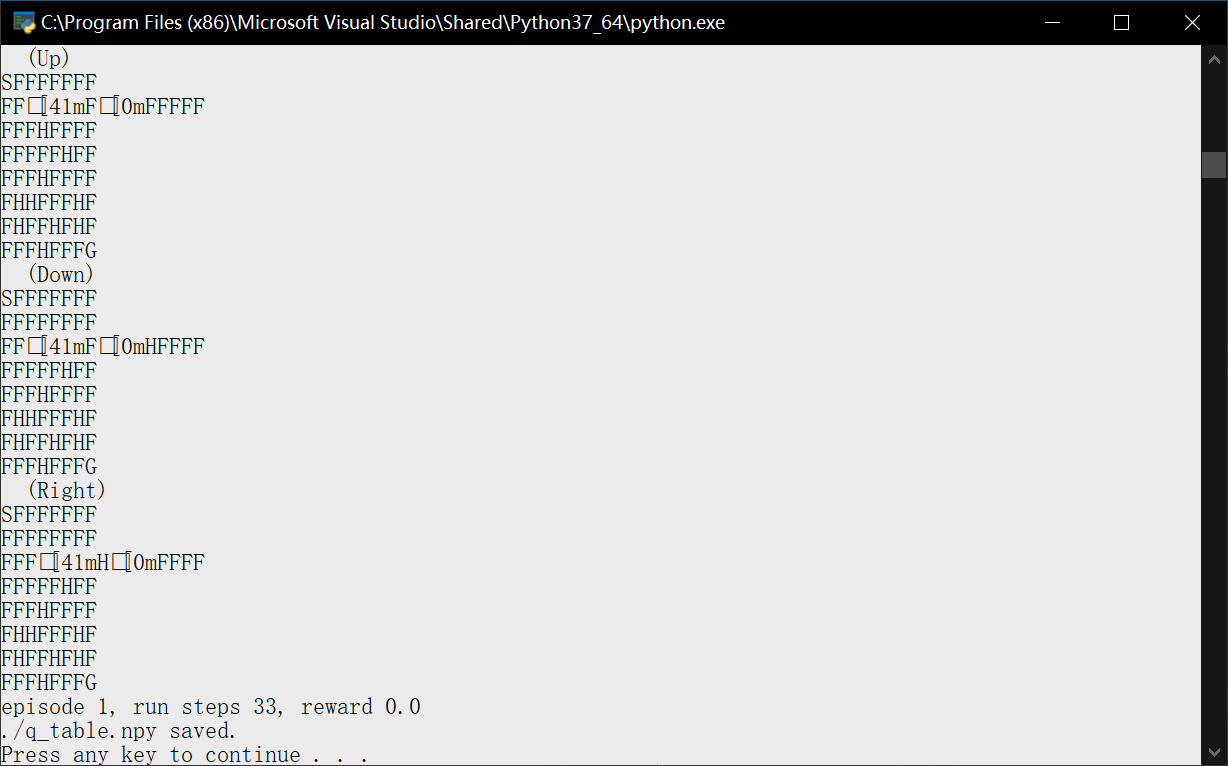


图3-4训练次数800次的测试结果

* 1. 训练次数1000次

此时已经能成功地到达终点了，Q表较为成熟。

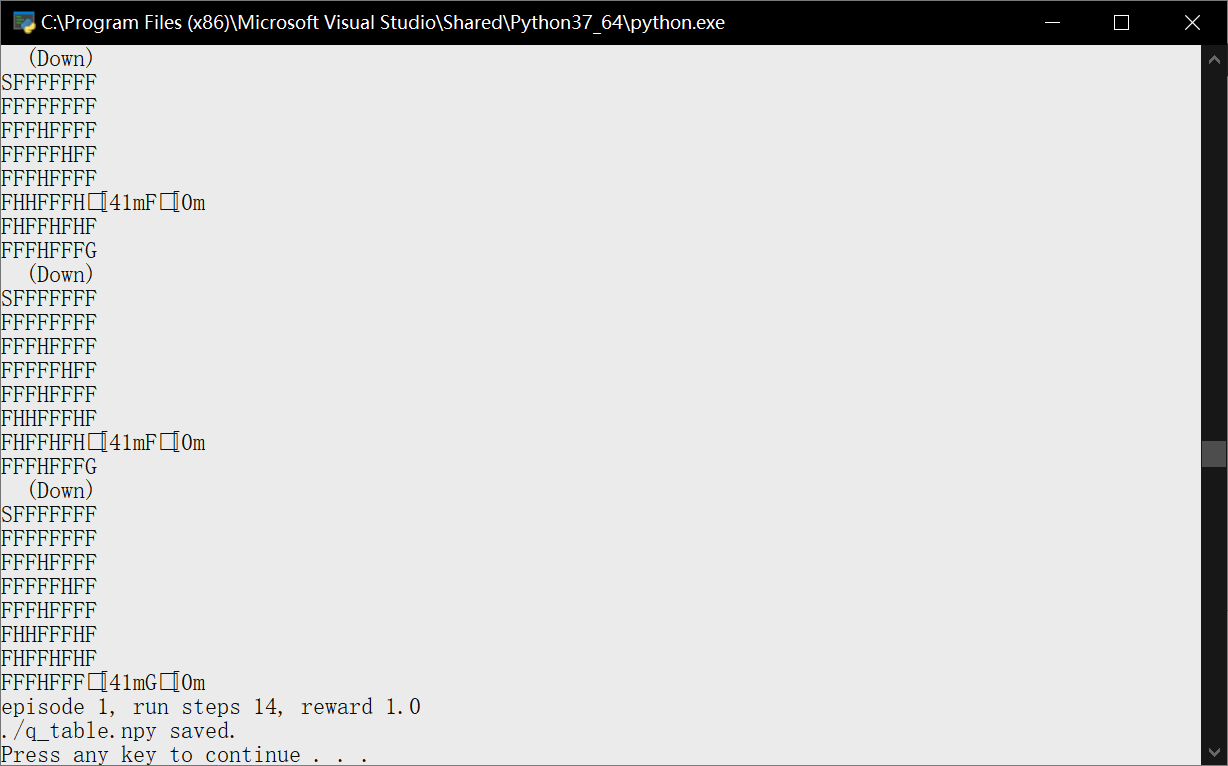


图3-5训练次数1000次的测试结果

由上可见，对于FrozenLake8x8迷宫，训练次数应该为1000次以上，为提高准确性，将训练次数设置为1500次。

# 实验心得与体会

本次课程作业是要求理解实现Q-Learning算法实现走迷宫，是我初次亲自接触这类强化学习算法。通过这次实验进一步理解了Q-Learning算法的思想，对于Q-Learning算法的实现步骤有了更清晰的认知，也对其缺点与不足之处有了了解。除此之外，对于Python语言的使用也得到了联系与巩固。

在刚刚接触强化学习等概念时我一度以为这些都是十分高深艰难的概念，但是在实际深入了解并自己上手过后才发现实际上也并没有那么高不可攀。本次实验让我有了强化学习初体验，对于人工智能等领域有了进一步接触。