**机器人自动走迷宫**

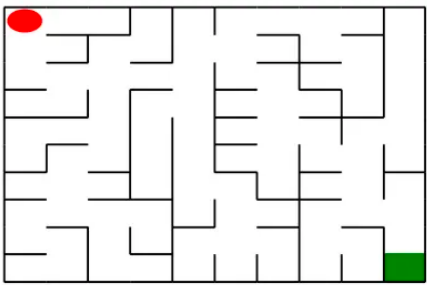
学号：3210102037 姓名：徐铭

1. **问题重述**

（简单描述对问题的理解，从问题中抓住主干，必填）

====================================================================

使用python语言，分别基于基础搜索算法（DFS/BFS/A\*）和Deep QLearning算法实现能够自动从初始位置（红色位置）走到迷宫（迷宫为随机生成）出口（绿色位置）的机器人：



1. **设计思想**

（所采用的方法，有无对方法加以改进，该方法有哪些优化方向（参数调整，框架调整，或者指出方法的局限性和常见问题），伪代码，理论结果验证等… **思考题，非必填**）

====================================================================

**方法概述**

本实验主要采用基础搜索算法DFS和强化学习中的 DQN 算法，通过深度神经网络来近似值函数，从而指导机器人在迷宫中找到路径。

**优化方向**

1. 调整学习率、折扣因子 、探索率等模型参数，如设置衰减率为0.5时能有比较好的效果
2. 尝试不同的神经网络结构（增加层数和神经元数量）。

**方法局限性**

1. DQN算法在高维状态空间中可能效果不佳。
2. 需要大量的训练数据和计算资源。
3. 容易出现过拟合问题。

**伪代码**

initialize\_replay\_memory()

initialize\_Q\_network()

initialize\_target\_network()

for episode in range(max\_episodes):

state = initialize\_state()

for t in range(max\_timesteps):

if random() < epsilon:

action = random\_action()

else:

action = argmax(Q\_network.predict(state))

next\_state, reward, done = step(action)

store\_experience(state, action, reward, next\_state, done)

if done:

break

state = next\_state

if len(replay\_memory) > batch\_size:

batch = sample\_experiences(batch\_size)

train(Q\_network, target\_network, batch)

if t % target\_update\_frequency == 0:

update\_target\_network()

1. **代码内容**

（能体现解题思路的主要代码，有多个文件或模块可用多个"===="隔开，必填）

====================================================================

class Robot(QRobot):

def \_\_init\_\_(self, maze):

"""

初始化 Robot 类

:param maze:迷宫对象

"""

super(Robot, self).\_\_init\_\_(maze)

self.maze = maze

def select\_action(self, state):

"""

根据当前状态选择动作

:param state: 当前状态

:return: 选择的动作

"""

if state not in self.q\_table:

self.q\_table[state] = {a: 0.0 for a in self.valid\_action}

if random.random() < self.epsilon:

return random.choice(self.valid\_action)

else:

return max(self.q\_table[state], key=self.q\_table[state].get)

def update\_q\_table(self, state, action, reward, next\_state):

"""

更新Q表

:param state: 当前状态

:param action: 执行动作

:param reward: 收到的奖励

:param next\_state: 下一个状态

"""

if next\_state not in self.q\_table:

self.q\_table[next\_state] = {a: 0.0 for a in self.valid\_action}

current\_q = self.q\_table[state][action]

max\_next\_q = max(self.q\_table[next\_state].values())

new\_q = reward + self.gamma \* max\_next\_q

self.q\_table[state][action] = self.alpha \* current\_q + (1 - self.alpha) \* (new\_q - current\_q)

def train\_update(self):

"""

以训练状态选择动作并更新Deep Q network的相关参数

:return :action, reward 如："u", -1

"""

state = self.maze.sense\_robot()

action = self.select\_action(state)

reward = self.maze.move\_robot(action)

next\_state = self.maze.sense\_robot()

self.update\_q\_table(state, action, reward, next\_state)

self.epsilon \*= 0.5

return action, reward

def test\_update(self):

"""

以测试状态选择动作并更新Deep Q network的相关参数

:return : action, reward 如："u", -1

"""

state = self.maze.sense\_robot()

action = self.select\_action(state)

reward = self.maze.move\_robot(action)

self.epsilon \*= 0.5

return action, reward

===============================================================================

def my\_search(maze):

"""

任选深度优先搜索算法、最佳优先搜索（A\*)算法实现其中一种

:param maze: 迷宫对象

:return :到达目标点的路径 如：["u","u","r",...]

"""

path = []

# -----------------请实现你的算法代码--------------------------------------

# 使用深度优先（DFS）

start = maze.sense\_robot()

root = SearchTree(loc=start)

stack = [root] # 迷宫节点栈

h, w, \_ = maze.maze\_data.shape

is\_visit\_m = np.zeros((h, w), dtype=np.int) # 标记迷宫的各个位置是否被访问过

while True:

current\_node = stack[-1] # 当前节点

if current\_node.loc == maze.destination: # 到达目标点

path = back\_propagation(current\_node)

break

if current\_node.is\_leaf() and is\_visit\_m[current\_node.loc] == 0: #叶子节点,有路可走

is\_visit\_m[current\_node.loc] = 1

expand(maze, is\_visit\_m, current\_node)

for child in current\_node.children:

stack.append(child)

else:

stack.pop() #无路可走

# -----------------------------------------------------------------------

return path

1. **实验结果**

（实验结果，必填）

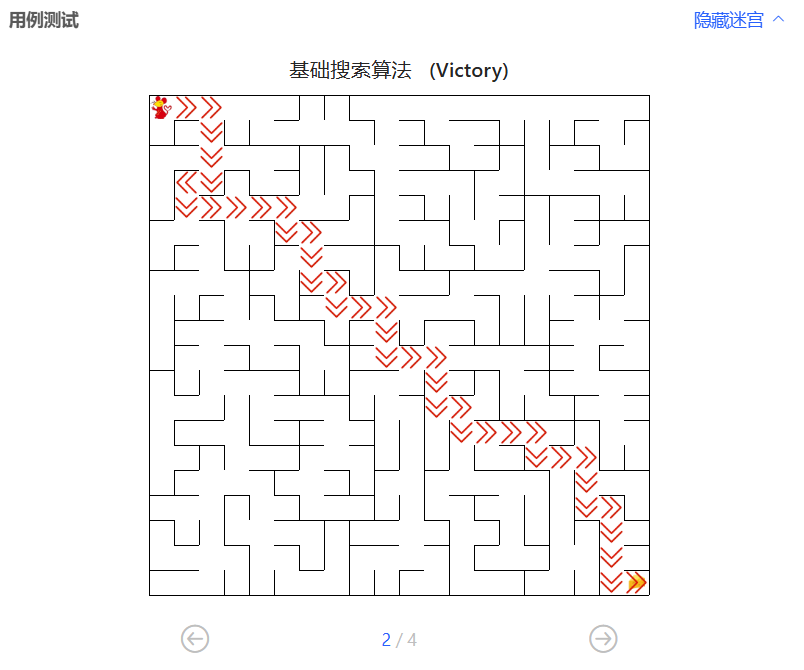
====================================================================

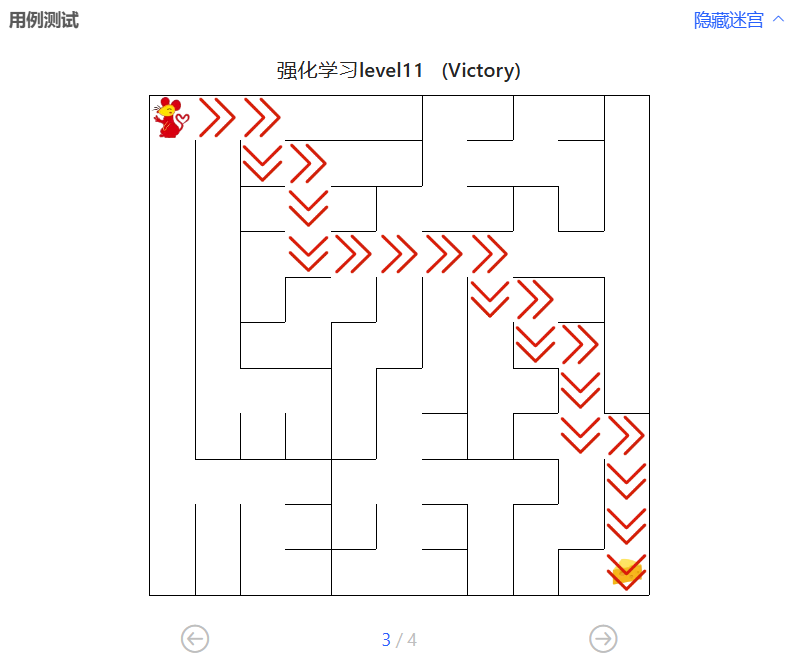
**测试结果：**

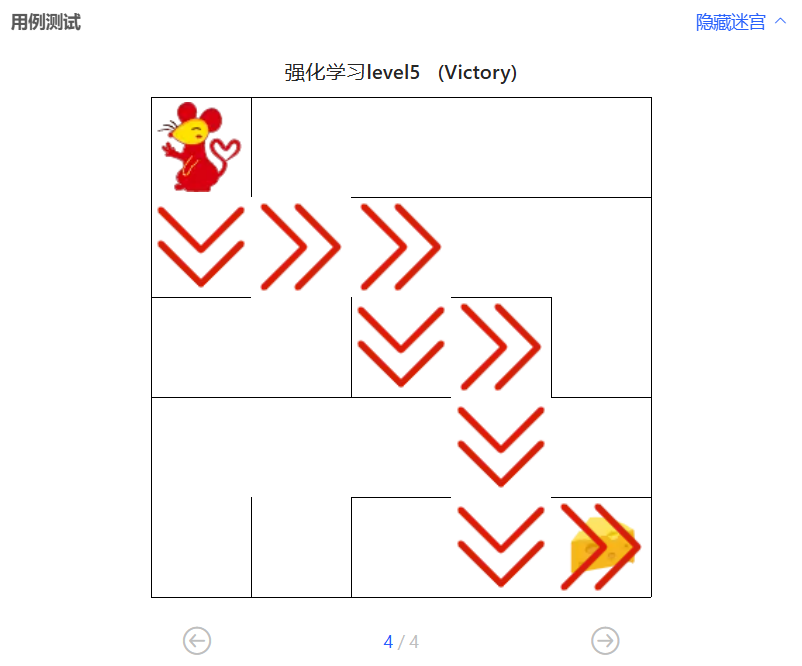


**该测试中的四张迷宫：**









1. **总结**

（自评分析（是否达到目标预期，可能改进的方向，实现过程中遇到的困难，从哪些方面可以提升性能，模型的超参数和框架搜索是否合理等），**思考题，非必填**）

====================================================================

**改进方向：**

1. 尝试引入双DQN、Dueling DQN等改进版本，提升算法性能。
2. 尝试不同的状态表示方法，如增加机器人感知的环境信息。