云南大学软件学院期末课程报告

**Final Course Report**

**School of Software, Yunnan University**

**个人成绩**

|  |  |  |  |
| --- | --- | --- | --- |
| **学号** | **姓名** | **专业** | **成绩** |
| **20211120292** | **张万棋** | **人工智能** |  |

学　 期: 2024年春季学期

课程名称: 强化学习

任课教师: 周玉珏

题 目: 迷宫问题

联系电话: 17759630999

电子邮件: 942880223@qq.com

报告完成时间： 2024 年 6 月 22 日

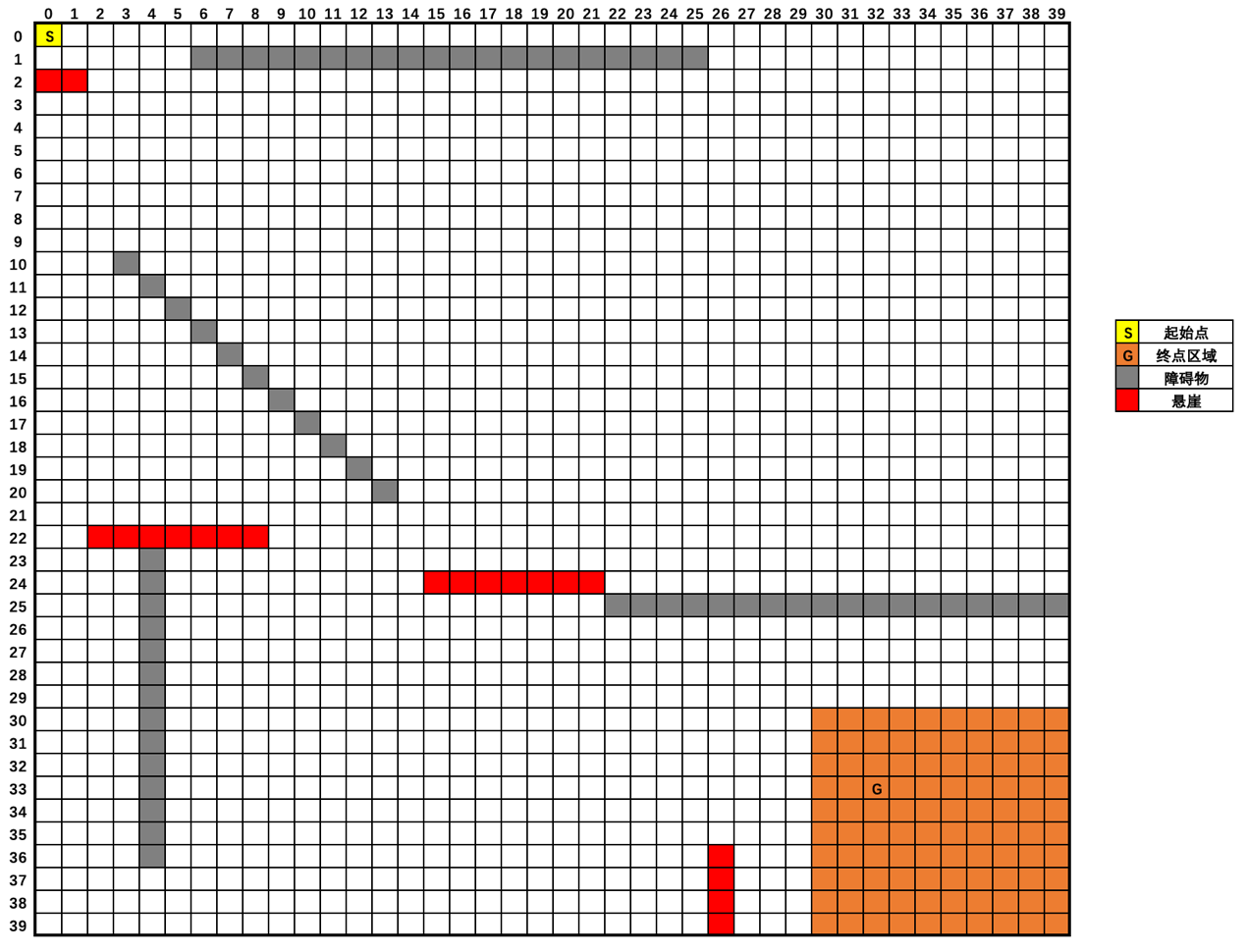
**期末报告成绩考核表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **指标内容** | **分值** | **指标内涵及评估标准** | | | | **得分** |
| 技术路线的可行程度 | 10 | 合理可行，具体且有创新 | 合理可行，具体 | 基本合理可行 | 不够合理或不够具体 |  |
| 项目解决方案的设计思路 | 15 | 思路非常清晰、运行正确 | 思路基本清晰、运行正确 | 思路清晰、环境配置错误无法运行 | 思路不清晰，程序无法运行 |  |
| 小组整体的工作量 | 15 | 高出平均要求工作量的15%以上 | 高出平均要求工作量 | 达到平均要求工作量 | 低于平均要求的工作量 |  |
| 理论知识应用水平 | 15 | 用理论知识对软件项目管理进行详尽、准确地分析和总结 | 用理论知识对软件项目管理进行较准确地分析和总结 | 理论知识应用一般 | 理论知识差 |  |
| 达到预期目标的程度 | 15 | 完全达到 | 基本达到 | 无法预见 | 未能达到 |  |
| 报告撰写质量 | 5 | 报告非常完整 | 报告比较完整 | 完整程度一般 | 报告不完整 |  |
| 5 | 逻辑结构清晰 | 逻辑组织较好 | 逻辑组织一般 | 逻辑不清 |  |
| 5 | 内容非常丰富 | 内容较丰富 | 内容一般 | 内容欠缺 |  |
| 5 | 文字表达非常好 | 文字表达较好 | 文字表达一般 | 文字表达差，意思不明了 |  |
| 5 | 图表制作非常专业化 | 图件制作良好 | 图件制作一般 | 图件制作效果差 |  |
| 5 | 整体效果很好 | 整体效果良好 | 整体效果一般 | 整体效果差 |  |
| 综合得分（满分100分） | |  | | | | |
| 评语 | | 通过学习了本学期的强化学习，该同学能够认识到多种强化学习算法，并通过期末大作业实践了Q-Learing算法，增强了团队合作、实验设计以及实验分析等相关能力，认真完成了期末大作业迷宫问题，认真撰写了实验报告、格式正确，符合本课程考察范围内。  **任课教师签名：**  **日期：** | | | | |

**2024年春季学期《强化学习》期末课程报告要求**

根据《强化学习》课程理论教学大纲，本门课程的学期要求如下：

 一、选题内容和要求



1. 游戏规则：40\*40迷宫

1）黄色为起点位置；

2）橙色为终点可能出现的区域，终点在智能体每走20步重新刷新一次位置（在橙色区域内随机出现）；

3）灰色为障碍物位置，没法走到或穿过；

4）红色为悬崖位置；

5）智能体走到终点位置或悬崖位置则本轮游戏结束，若步数超过500则本轮游戏结束，智能体的奖励函数可自行定义。

2. 实现要求：

1）所有代码要求在python上实现；

2）涉及到神经网络的部分要求用pytorch实现；

3）算法部分不能直接调用package；

4）尽可能实现多种强化学习算法以解决问题，并分析各种算法的性能表现、收敛速度等。

 二、课程报告撰写规范

1. 总结报告按如下内容顺序用A4纸进行打印（撰写）并装订成册：

1） 统一的封面；

2） 课程期末课程报告个人成绩考核表；

3） 课程期末课程报告要求；

4） 题目、作者、摘要和关键字；

5） 目录；

6） 报告正文。

2. 期末总结报告正文可按章节来撰写，应包含以下内容：

1）研究背景与意义

2）文献综述

3）任务分析

4）方法阐述

5）结果分析

6）总结

7）参考文献

8）附录（如有）

摘要

迷宫问题是一个经典的强化学习应用场景，其中智能体需要在充满障碍物和悬崖的迷宫中找到通往目标的最优路径。作为人工智能领域的一个重要分支，强化学习近年来受到了广泛关注，因为它不需要大量的标记数据，可以直接通过与环境的交互来学习最佳策略。本实验使用了Q-Learning 算法来训练智能体在迷宫环境中导航。 迷宫问题具有挑战性，因为智能体需要在不确定性环境中做出决策。在每个时间步，智能体观察当前状态并选择一个动作。然后，环境根据智能体的动作转移到下一个状态，并给智能体一个奖励信号，表示动作的好坏。智能体的目标是学习一个策略，使得长期累积的奖励最大化。传统迷宫难题在数据结构和算法研究和教育中得到了广泛应用。众所周知的Dijkstra 最短路径算法 仍然是解决此类难题最实用的方法，但由于其熟悉性和直观性，这些难题非常适合演示和测试强化学习技术。Q-Learning 是一种著名的无模型强化学习算法，它通过一个叫做Q表的数据结构来存储每个状态动作对的值。Q表中的每个条目代表了在特定状态下采取特定动作的期望回报。在训练过程中，智能体根据Q表来选择动作，并不断更新Q表以反映新的经验。这种方法的关键优点是它能够适应环境的变化，并逐步改进其策略。 本实验的目的是评估Q-Learning 算法在迷宫问题上的性能，并探讨不同超参数设置对算法性能的影响。我们使用了折扣因子，探索率，奖励机制等方法来调节Q-Learning 算法的行为。我们期望通过适当的超参数设置，智能体能够有效地学习迷宫问题的解决方案，并在训练过程中表现出良好的性能。 在接下来的部分，我们将详细介绍迷宫环境的搭建、Q-Learning 算法的实现以及实验结果和分析。我们还将讨论Q-Learning 算法的优缺点/通过本实验，我们可以更好地理解强化学习算法在解决迷宫问题时的行为和性能，为进一步的研究和应用提供有价值的参考。

**关键词**：强化学习、Q-Learing、迷宫问题

**目录**

[摘要 1](#_Toc5108)

[目录 1](#_Toc22294)

[一、 选题背景及意义 1](#_Toc10371)

[（一） 迷宫问题的背景和挑战： 1](#_Toc28597)

[二、 文献综述 2](#_Toc995)

[三、 任务分析 4](#_Toc30076)

[（一） 功能需求： 4](#_Toc9646)

[1． 迷宫环境搭建： 4](#_Toc5198)

[2． Q-Learning算法实现： 4](#_Toc23414)

[结果分析与可视化： 4](#_Toc4498)

[（二） 性能需求： 4](#_Toc15987)

[四、 迷宫环境搭建 5](#_Toc13409)

[（一） 环境设计与实现 5](#_Toc24148)

[1． 环境概述 5](#_Toc30970)

[（1） 背景与目标 5](#_Toc27771)

[（2） 迷宫环境类的实现 5](#_Toc24777)

[2． 迷宫环境流程 5](#_Toc3032)

[（1） 初始化 5](#_Toc21872)

[（2） 创建迷宫 5](#_Toc12361)

[3． 获取随机终点(\_get\_random\_end 方法) 5](#_Toc28943)

[4． 重置环境（reset方法） 6](#_Toc4672)

[5． 环境步进（step方法） 6](#_Toc15828)

[6． 渲染环境（render方法） 6](#_Toc8336)

[五、 Q-Learning算法搭建 7](#_Toc20260)

[（一） 算法设计与实现 7](#_Toc27008)

[1． 算法概述 7](#_Toc26833)

[（1） 背景与目标 7](#_Toc32154)

[（2） Q-Learning基本原理 7](#_Toc19773)

[2． 算法实现步骤 7](#_Toc8656)

[（1） Q表的初始化 7](#_Toc20824)

[（2） 状态预处理 8](#_Toc5098)

[（3） 动作选择策略 8](#_Toc3552)

[（4） Q-Learning算法训练过程 8](#_Toc31590)

[3． 可视化 8](#_Toc32589)

[（1） 奖励和步骤数的可视化 8](#_Toc12833)

[六、 实验结果分析 9](#_Toc12863)

[（一） 结果分析 9](#_Toc11141)

[七、 总结与展望 27](#_Toc8016)

[八、 参考文献 28](#_Toc15717)

# 选题背景及意义

强化学习是一种机器学习技术，通过反馈系统（奖励和惩罚）解决问题，该反馈系统应用于在环境中运行的代理，需要经过一系列状态才能达到预定义的最终状态。一个典型的例子是老鼠（代理），它试图在迷宫（环境）中找到从起始单元到目标奶酪单元的最短路线。代理正在试验并利用过去的经验（情节）来实现其目标。它可能会一次又一次地失败，但希望在经过大量的反复试验（奖励和惩罚）后，它能够找到问题的答案。如果代理找到累积奖励总和最大的最佳状态序列（简而言之，我们引诱代理累积最大奖励，而这样做时，他实际上解决了我们的问题），则将获得解决方案。请注意，为了达到目标，代理可能不得不在途中承受许多惩罚（负奖励）。例如，迷宫中的老鼠每次合法移动都会受到小惩罚。这样做的原因是我们希望它以尽可能短的路径到达目标单元格。然而，到达目标奶酪单元格的最短路径有时又长又曲折，我们的代理（老鼠）可能不得不忍受许多惩罚，直到它到达“奶酪”（有时称为“延迟奖励”）。

## 迷宫问题的背景和挑战：

迷宫问题是一个经典的强化学习应用场景，其中智能体需要在充满障碍物和悬崖的迷宫中找到通往目标的最优路径。这一问题的挑战在于智能体需要在不确定性环境中做出决策，并在每个时间步观察当前状态并选择一个动作。环境根据智能体的动作转移到下一个状态，并给智能体一个奖励信号，表示动作的好坏。智能体的目标是学习一个策略，使得长期累积的奖励最大化。

# 文献综述

强化学习作为人工智能领域的一个重要分支，近年来受到了广泛关注。它通过智能体与环境的交互，学习最优策略，在机器人导航、游戏AI、自动驾驶等领域展现出巨大的潜力。本文将围绕迷宫问题这一经典场景，综述强化学习算法的研究现状、开发进展和应用案例。

研究现状：

传统迷宫难题在数据结构和算法研究和教育中得到了广泛应用。众所周知的Dijkstra 最短路径算法 仍然是解决此类难题最实用的方法，但由于其熟悉性和直观性，这些难题非常适合演示和测试强化学习技术。

MDP （马尔可夫决策过程）的框架由环境 和在该环境中行动的代理组成。在我们的例子中，环境是一个经典的方形迷宫，有三种类型的单元：

已占用的单元格

自由细胞

目标单元格（奶酪所在的单元格）

在经典问题中，我们的代理是一只老鼠，它只能在空闲的单元格上移动，并且它生命中唯一的目标就是找到奶酪。

机器学习中，最经典的算法就是Qlearning了：

Qlearning的主要目标是制定策略π成功穿越迷宫的关键。据推测，在玩了数百次游戏之后，代理（我们例子中的老鼠）应该会获得一个明确的确定性策略，以了解如何在每种可能的情况下采取行动。在每种可能的迷宫状态下采取什么行动？

政策一词应该被简单地理解为一种功能π它以迷宫快照 ( envstate ) 作为输入，并返回代理(本例中为老鼠) 要采取的操作。输入包括完整的 nxn 迷宫单元状态和老鼠位置。

下一个动作 =π(状态)

一开始，我们只是选择一个完全随机的策略。然后我们用它玩了数千场游戏，从中学习如何完善它。当然，在早期的训练阶段，我们的策略 π会产生大量错误并导致我们输掉很多比赛，但我们的奖励策略应该为其提供如何改进自身的反馈。我们的学习引擎将是一个简单的前馈神经网络，它将环境状态（迷宫单元）作为输入并为每个动作向量产生奖励（请参阅后面的详细说明）。

为了增强 Q 学习过程，我们将使用两种类型的动作：

剥削：这些是我们的政策采取的行动π根据以往的经验进行指示。在完成之前，大约 90% 的移动都使用了策略函数。

探索：在大约 10% 的情况下，我们会采取完全随机的行动来获得新的体验（并可能获得更大的回报），而我们的策略功能可能由于其限制性而不允许我们这样做。可以将其想象为偶尔选择一家完全随机的新餐厅，而不是选择您已经熟悉的常规餐厅。

探索因子epsilon是进行探索的频率级别。它通常设置为 0.1，这大致意味着在每 10 次移动中，代理会采取一次完全随机的行动。

总而言之，强化学习算法在迷宫问题等经典场景中取得了显著的研究成果和应用进展。未来，随着算法的不断改进和技术的不断发展，强化学习技术将在更多领域发挥重要作用，并推动人工智能技术的进步和应用。

# 任务分析

## 功能需求：

### 迷宫环境搭建：

创建一个40x40的迷宫环境，包括起始点、目标区域、障碍物和悬崖等。

设计迷宫环境的状态表示，包括智能体位置、目标位置和周围环境信息。

定义智能体的动作空间，包括上、下、左、右四种动作。

设计奖励机制，根据智能体的行为给予相应的奖励或惩罚。

实现迷宫环境的重置和渲染功能。

### Q-Learning算法实现：

实现Q-Learning算法，包括Q表的初始化、状态预处理、动作选择策略和Q值更新等。

调整算法的超参数，例如学习率、折扣因子和探索率等。

实现算法的训练过程，并记录训练结果，例如累积奖励、步骤数等。

## 结果分析与可视化：

可视化迷宫环境和智能体的路径。

绘制累积奖励随训练轮数变化的图像，分析算法的性能和收敛情况。

分析智能体在不同训练阶段的行为和学习策略。

## 性能需求：

算法收敛性：算法能够在有限的训练轮数内收敛，并找到接近最优的路径规划策略。

算法效率：算法能够在较短的时间内完成训练，并具有较高的计算效率。

算法鲁棒性：算法能够适应不同的迷宫环境和超参数设置，并保持稳定的性能。

# 迷宫环境搭建

## 环境设计与实现

### 环境概述

#### 背景与目标

在强化学习的研究中，迷宫环境常被用作智能体（agent）的测试平台。迷宫环境能够模拟各种复杂的路径规划问题，并为智能体的学习过程提供了丰富的状态和行动选择空间。本研究中，我们设计并实现了一个40x40的迷宫环境，旨在测试Q学习算法的有效性和性能。

#### 迷宫环境类的实现

`Maze`类是本研究中核心的环境类。这个类设置并管理迷宫环境，包括障碍物、悬崖、起点和终点位置，奖励机制以及渲染环境。

### 迷宫环境流程

#### 初始化

设置迷宫大小。

创建迷宫并初始化障碍物和悬崖。

设置起点和终点位置。

初始化 Pygame 渲染参数

#### 创建迷宫

初始化一个 40x40 的网格。

将特定区域设置为障碍物或悬崖

### 获取随机终点(\_get\_random\_end 方法)

从定义的目标区域中随机选择一个目标位置

### 重置环境（reset方法）

重置起点位置和步数。

随机设置一个新的终点位置。

### 环境步进（step方法）

根据采取的动作更新代理的位置。

根据新位置和条件（如撞到障碍物、到达终点等）计算奖励。

### 渲染环境（render方法）

使用 Pygame 渲染当前的迷宫状态，包括代理位置、障碍物和终点。

# Q-Learning算法搭建

## 算法设计与实现

### 算法概述

#### 背景与目标

Q-Learning是一种基于值函数的强化学习算法，通过学习状态-动作值函数（Q函数）来实现策略优化。Q-Learning算法在各种离散状态和动作空间的强化学习问题中得到了广泛应用。本研究的目标是通过设计和实现Q-Learning算法，使智能体能够在自定义的迷宫环境中有效地进行路径规划和导航。

#### Q-Learning基本原理

Q-Learning算法的核心是通过经验数据更新Q值，使得在长期回报最大的策略下进行行动。Q值更新的基本公式为：

|  |  |
| --- | --- |
|  | 公式 二-1 |

其中：

· 和 分别表示当前状态和下一个状态；

· 和 分别表示当前动作和下一个动作；

· 是当前动作的即时奖励；

· 是学习率；

· 是折扣因子。

### 算法实现步骤

#### Q表的初始化

Q表用于存储状态-动作对的值。对于每个状态，Q表包含了四个动作（上、下、左、右）的值。Q表的初始值设为零，表示在没有任何经验数据的情况下，所有状态-动作对的值均为零。初始化 epsilon、alpha、gamma。

#### 状态预处理

为了简化状态表示和计算，我们对状态进行了预处理。预处理后的状态包括智能体当前位置、目标位置以及一个3x3的局部网格。

#### 动作选择策略

为了平衡探索和利用，我们采用了ε-贪心策略（epsilon-greedy policy）来选择动作。

在ε-贪心策略中，智能体以概率ε随机选择动作（探索），以概率1-ε选择当前Q值最大的动作（利用）。

#### Q-Learning算法训练过程

在指定的 EPISODES 中训练代理。

重置环境。

根据 epsilon-greedy 策略选择动作。

执行动作并获取新状态和奖励。

更新 Q 表格。

根据衰减策略更新 epsilon。

记录每个回合的奖励。

### 可视化

#### 奖励和步骤数的可视化

为了直观地展示训练过程中的奖励变化，我们使用Matplotlib库绘制了奖励随实验周期变化的图像。

# 实验结果分析

## 结果分析

在本实验中，我们使用了Q-Learning算法对一个40x40的迷宫环境进行了训练，统一进行了3000轮的训练，记录的路径展示了智能体在迷宫中的具体行动轨迹，可以通过可视化工具进一步分析其路径规划策略。

我利用控制变量的方法，做了关于不同变量的对比试验

### 其他变量不变，epsilon分别取0.1，0.01，0.005，进行实验对比（学习率Alpha=0.2，折扣因子Gamma=0.95，终点reward+1000，悬崖障碍物reward-1000）

图表, 直方图

描述已自动生成

图 1：epsilon=0.1

图表, 直方图

描述已自动生成

图 2：epsilon=0.01

图表, 直方图

描述已自动生成

图 3 epsilon=0.005

图表, 直方图

描述已自动生成

图 4 epsilon=0.001

表格 1：控制其他不变，Epsilon不同取值所对应的效果

|  |  |  |
| --- | --- | --- |
| Epsilon | 开始收敛轮数（估计） | Reward稀疏程度排名 |
| 0.1 | >3000 | 4 |
| 0.01  0.005  0.001 | 1800  1600  2500 | 3  1  2 |

由对比实验结果可见，在Epsilon=0.005时，该模型的效果最优，在后续实验则选择采取epsilon=0.005

### 在探究epsilon的最优值后，在训练过程中发现，agent在到达终点区域后，会时常在终点区域转圈圈，而不是探索终点位置，所以我考虑在全部区域采取欧氏距离方法，当与终点距离增加时进行惩罚，减少则相反，来探索收敛的最优解，实验结果如下（学习率Alpha=0.2，折扣因子Gamma=0.95，终点reward+1000，悬崖障碍物reward-1000，epsilon=0.005）

图表, 条形图

描述已自动生成

图 5 欧氏距离reward=0.01（全部区域）

添加全局欧氏距离后，发现，直接使模型完全崩塌，则该方法不继续采用

### 在探究全部区域的欧氏距离的训练过程中发现，agent到达不了终点，所以我考虑在终点区域采取欧氏距离方法，当与终点距离增加时进行惩罚，减少则相反，来探索收敛的最优解，实验结果如下（学习率Alpha=0.2，折扣因子Gamma=0.95，终点reward+1000，悬崖障碍物reward-1000，epsilon=0.005）

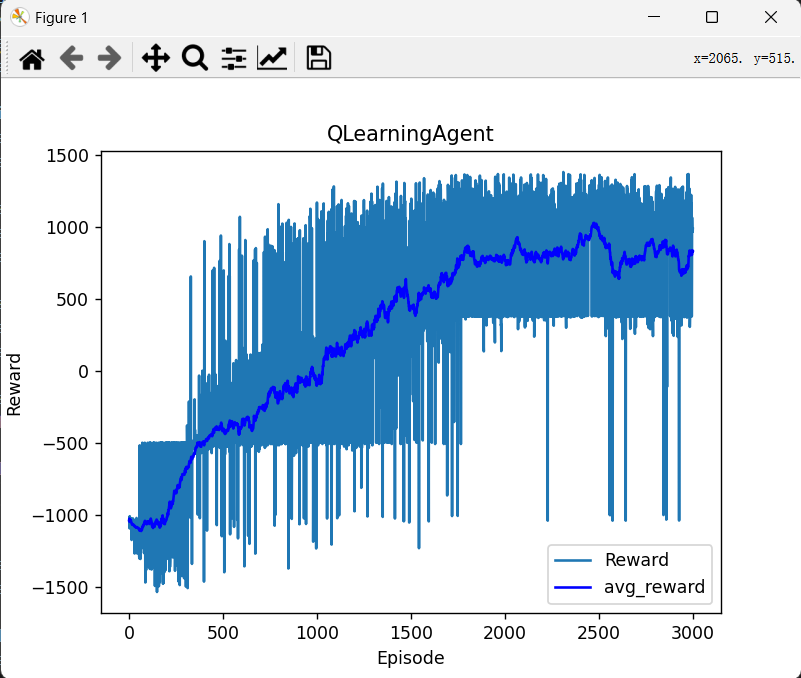


图 6：加入欧氏距离 reward=1（终点区域）

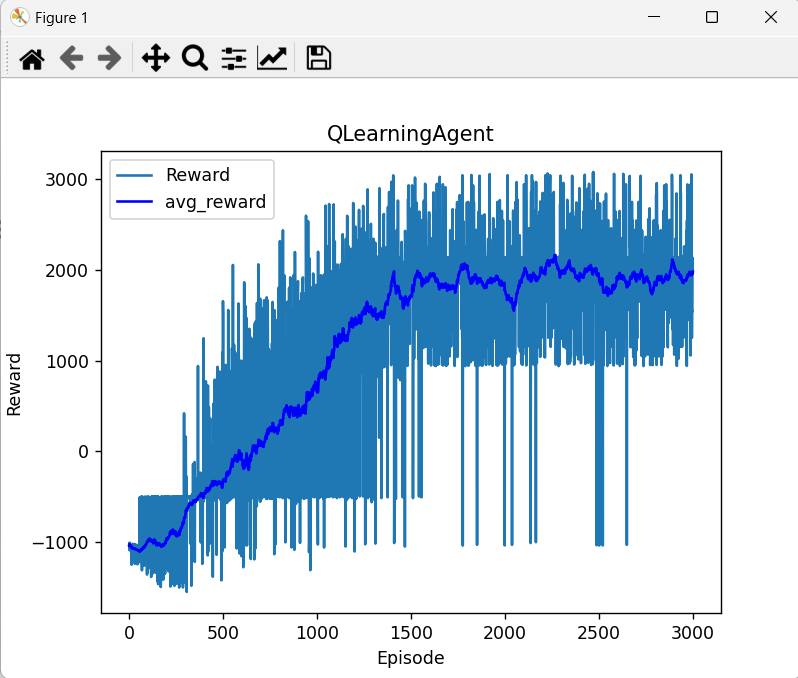


图 7 欧氏距离 reward=5（终点区域）

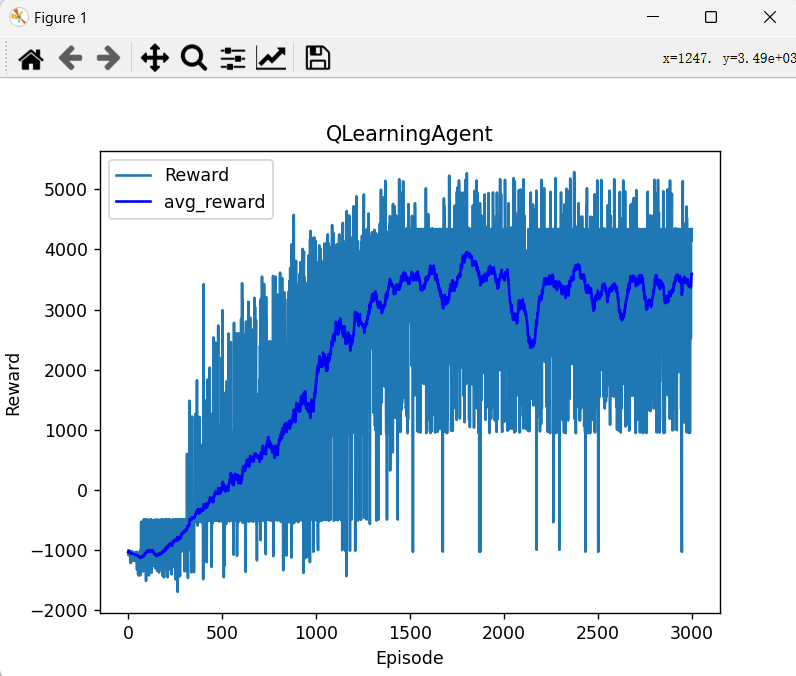


图 8 欧氏距离reward=10（终点区域）

在训练的过程中发现，Agent在终点区域探寻终点的过程中，出现反复横跳，反复转圈的情况，对于寻找终点极为不利，所以我认为可能在欧氏距离的设置中，距离变大的奖励与距离变小的奖励，出现相消的现象，导致出现此现象，所以采取了不对称的奖励机制，进行验证。

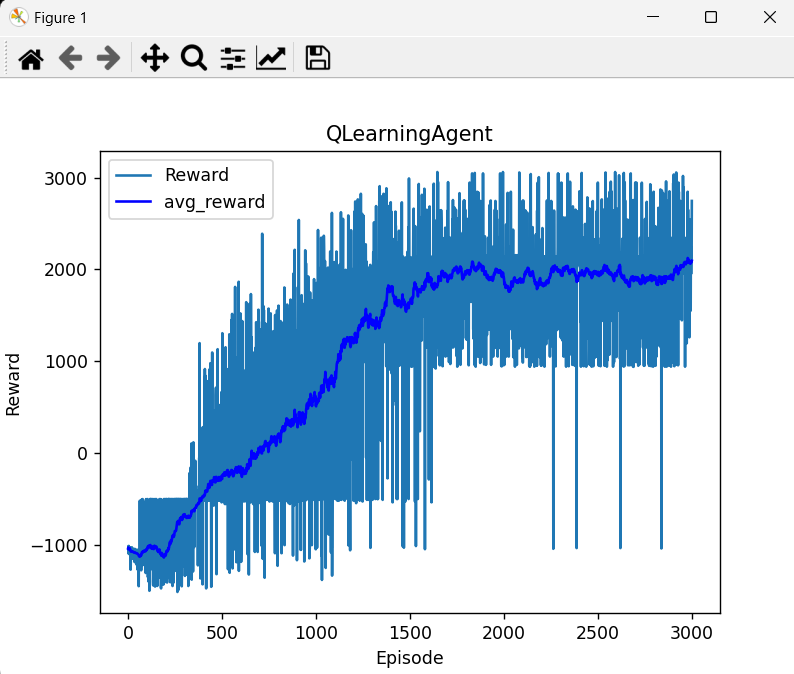


图 9欧氏距离reward=-10 reward=5（终点区域）

表格 2控制其他不变，加入欧氏距离（终点区域）机制后不同奖励取值，所对应的效果

|  |  |  |
| --- | --- | --- |
| 欧氏距离reward | 开始收敛轮数（估计） | Reward稀疏程度排名 |
| 1 | 1750 | 3 |
| 5  10  -10，+5 | 1500  1300  1600 | 2  2  1 |

由对比实验结果可见，在reward=10时，该模型开始收敛的效果最优，在-10，+5的reward中，收敛的稳定性最好，由于主观性，认为-10，+5的reward中，开始收敛的效果大大下降，而稳定性相差不大，则在后续实验则选择采取reward=10欧氏距离（终点区域）的奖励机制

### 在探究欧氏距离（终点区域）的最优值后，在训练过程中还是会发现，agent到达终点后，收敛的还是不够快，因此为了让agent快速到达终点，我在添加欧氏距离（终点区域）的奖励机制基础上，进行添加全局区域的欧氏距离，来探索收敛的最优解（学习率Alpha=0.2，折扣因子Gamma=0.95，欧氏距离（终点区域）reward=10，悬崖障碍物reward-1000，epsilon=0.005）在欧氏距离（全局区域）奖励机制考虑中，出于对前面的经验和本次探究的目的（快速到达终点）经验，以及训练过程中发现智能体在上半区探寻时间长，所以本次选择仅对欧氏距离增加时进行惩罚，以推动智能体向前寻找终点

图表, 直方图

描述已自动生成

图 10 欧氏距离（全局区域）reward=-0.01

图表, 直方图

描述已自动生成

图 11欧氏距离（全局区域）reward=-0.1

图表, 直方图

描述已自动生成

图 12 欧氏距离（全局区域）reward=-1

表格 3：控制其他不变，加入欧氏距离（全局区域）机制后不同奖励取值，所对应的效果

|  |  |  |
| --- | --- | --- |
| 欧氏距离reward | 开始收敛轮数（估计） | Reward稀疏程度排名 |
| 0 | 1300 | 2 |
| -0.01  -0.1  -1 | 1500  1400  1600 | 2  3  1 |

### 在探究欧氏距离的最优值后，在训练过程中还是会发现，agent在到达终点区域后，会时常在终点区域转圈圈，而不是探索终点位置，虽然有所改观但不显著，所以我考虑改变终点的reward，来探索收敛的最优解，实验结果如下（学习率Alpha=0.2，折扣因子Gamma=0.95，欧氏距离reward=10，悬崖障碍物reward-1000，epsilon=0.005）

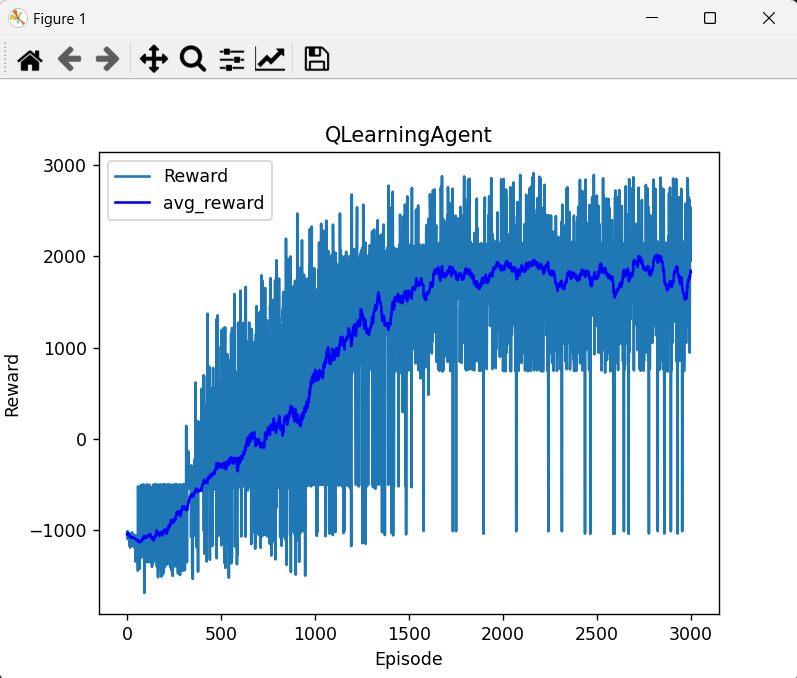


图 13终点Reward=800

图表, 直方图

描述已自动生成

图 14 终点Reward=500

图表, 直方图

描述已自动生成

图 15 终点Reward=900

图表, 直方图

描述已自动生成

图 16终点Reward=1500

表格 4 控制其他不变，加入欧氏距离（全局区域）机制后不同奖励取值，所对应的效果

|  |  |  |
| --- | --- | --- |
| 终点Reward | 开始收敛轮数（估计） | Reward稀疏程度排名 |
| 500 | 1500 | 1 |
| 800  900  1500 | 1500  1700  1600 | 2  3  2 |

实验发现，似乎Reward=1000已经是局部最优，则继续采取终点Reward=1000的奖励数值

### 在探究终点的最优值后，在训练过程中还是会发现，agent在到达终点区域后，会时常在终点区域转圈圈，而不是探索终点位置，所以猜测是不是探索率太低的问题，导致了不选择探索，所以控制了其他变量，添加探索率衰减机制进行实验，实验结果如下（学习率Alpha=0.2，折扣因子Gamma=0.95，欧氏距离reward=10，悬崖障碍物reward-1000）

图表, 直方图

描述已自动生成

图 17 epsilon=0.9 dacay=0.995 下限0.005

图表, 直方图

描述已自动生成

图 18 epsilon=0.1 下限 0.005 dacay0.995

表格 6 控制其他不变，添加探索率衰减机制，所对应的效果

|  |  |  |  |
| --- | --- | --- | --- |
| epsilon | Epsilon\_decay | 开始收敛轮数（估计） | Reward稀疏程度排名 |
| 0.9 | 0.995 | 1900 | 1 |
| 0.1 | 0.995 | 1600 | 2 |

发现添加探索率衰减的机制后，并没有改善，反而恶化了收敛效果

### 对于agent在到达终点区域后，会时常在终点区域转圈圈，而不是探索终点位置这个问题还是未解决，所以猜测是不是欧氏距离设置的问题，导致此现象，所以控制了其他变量，添加动态欧氏距离机制进行实验，实验结果如下（学习率Alpha=0.2，折扣因子Gamma=0.95，悬崖障碍物reward-1000）

图表, 直方图

描述已自动生成

图 19 动态欧氏距离reward=欧氏距离\*3

图表, 直方图

描述已自动生成

图 20动态欧氏距离reward =欧氏距离\*1

表格 7 控制其他不变，添加动态欧式距离机制，所对应的效果

|  |  |  |
| --- | --- | --- |
| 欧氏距离Reward | 开始收敛轮数（估计） | Reward稀疏程度排名 |
| 距离\*1 | 1900 | 1 |
| 距离\*3 | 1600 | 2 |

在添加动态欧式距离机制后，效果并没有多大的改观，所以认为问题并非处在此处，则实验未进行下去

### 继续对于agent在到达终点区域后，会时常在终点区域转圈圈，而不是探索终点位置这个问题机型探究，所以猜测是不是折扣因子设置的问题，导致此现象，所以控制了其他变量，添加动态欧氏距离机制进行实验，实验结果如下（学习率Alpha=0.2，悬崖障碍物reward-1000，欧氏距离reward=10，终点reward+1000）

图表, 直方图

描述已自动生成

图 21 GAMMA=0.99

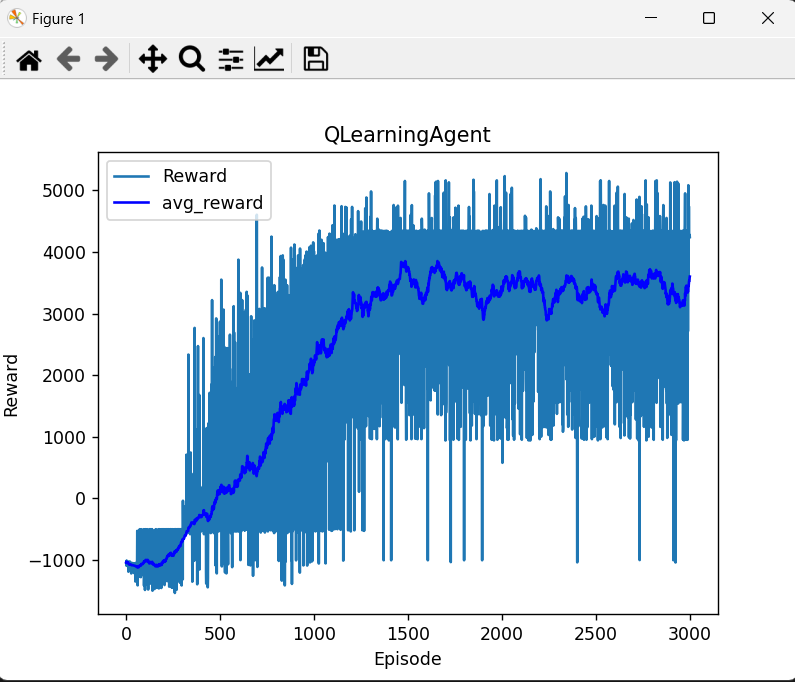


图 22 gamma=0.995

表格 7 控制其他不变，Gamma不同，所对应的效果

|  |  |  |
| --- | --- | --- |
| Gamma（折扣因子） | 开始收敛轮数（估计） | Reward稀疏程度排名 |
| 0.99 | 1200 | 1 |
| 0.995  0.95 | 1200  1300 | 2  3 |

探索发现，Gamma的取值不同增加了模型的效果，所以在后续实验采取Gamma=0.99的参数值进行实验，其实对于Gamma的取值，我还取了0.97，0.98等更细微的取值，但是发现效果并未提升，则没有进行做对比

### 关于在终点转圈圈的问题，还未有明显的提升效果，所以先放下，解决快速收敛的问题，对于快速收敛，控制收敛速度最客观的就是学习率，所以猜测是不是学习率过低，导致无法快速收敛，因此对于学习率变量问题，进行对比试验（gamma=0.99，悬崖障碍物reward-1000，欧氏距离reward=10，终点reward+1000）

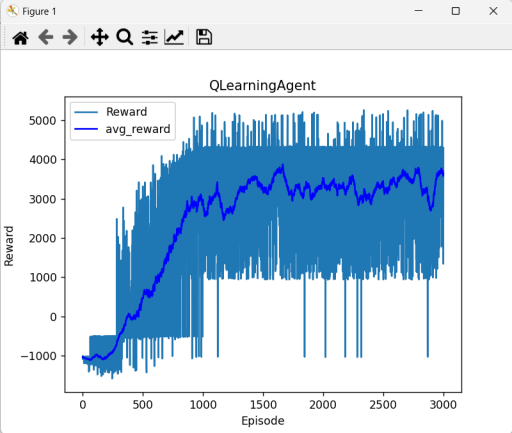


图 23alpha=0.3

图表, 直方图

描述已自动生成

图 24 alpha=0.4

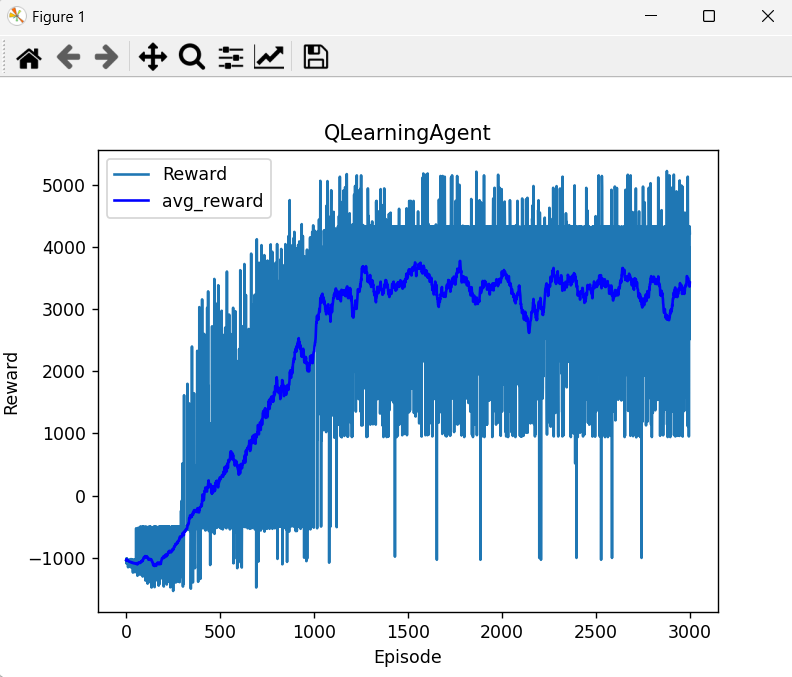


图 25 alpha=0.25

表格 9 控制其他不变，Apaha不同，所对应的效果

|  |  |  |
| --- | --- | --- |
| Alpha（学习率） | 开始收敛轮数（估计） | Reward稀疏程度排名 |
| 0.3 | 1000 | 1 |
| 0.4  0.25  0.2 | 1000  1100  1300 | 2  1  3 |

果然，如猜测所料，对于学习率，我们实验采取的过小，导致前期的学习过慢，从而无法快速收敛。

在实验过程中，其实也采取了学习率为0.5，0.9，0.99，甚至1的实验，试图加快收敛速度，但是发现并没有效果，则猜测可能是确实加速了探索前往重点路线的速度，但是由于在终点区域的探索终点的速度过慢，导致无法快速收敛

### 关于快速收敛的问题，由于先前的测试，alpha探索率设置超过0.9时，效果并不好，所以考虑加入alpha衰减机制，进行测试（gamma=0.99，悬崖障碍物reward-1000，欧氏距离reward=10，终点reward+1000）

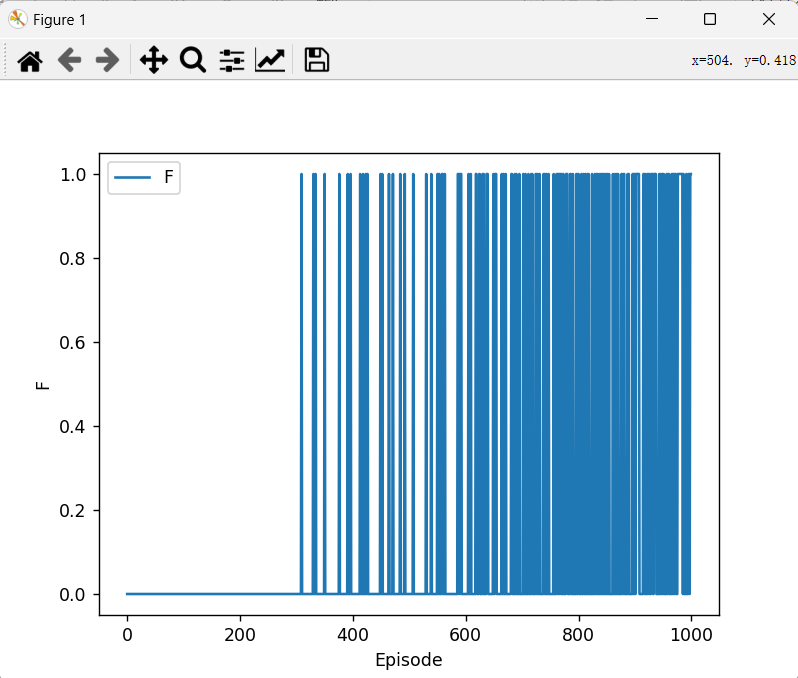


图 26 alpha=0.4 alpha\_decay=0.995 下限0.3

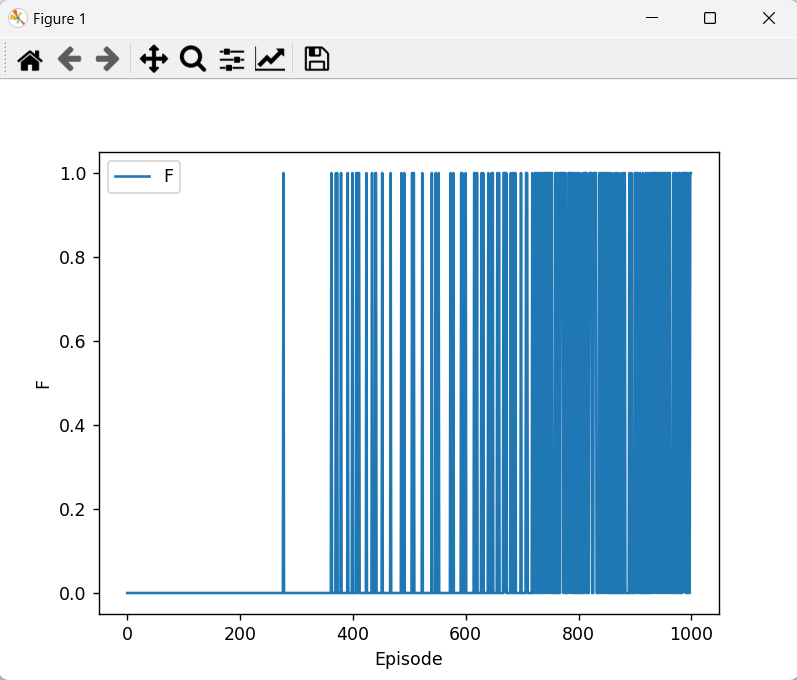


图 27：alpha=0.5 alpha\_decay=0.995 下限0.3

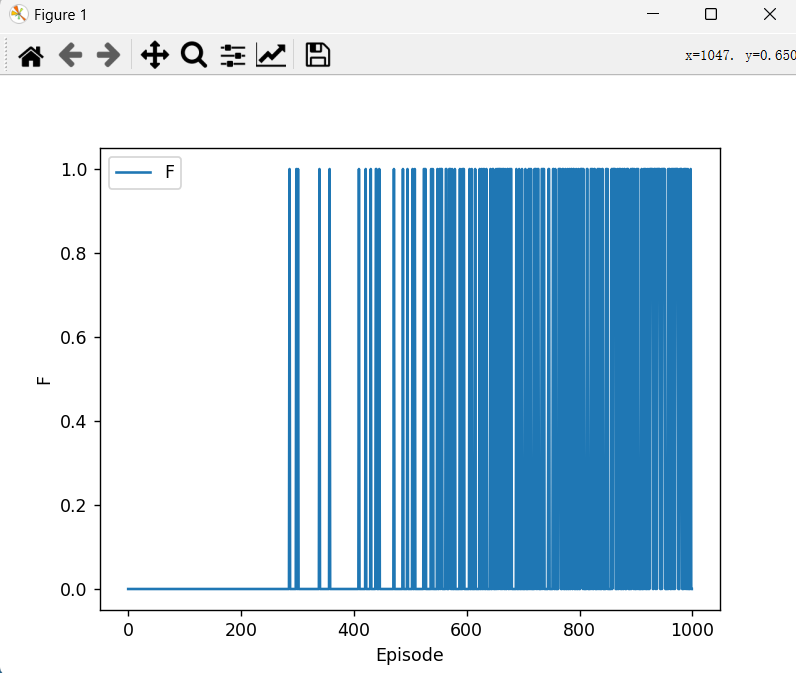


图 28 alpha=0.6 alpha\_decay=0.995 下限0.3

表格 9 控制其他不变，Alpha衰减，所对应的效果

|  |  |
| --- | --- |
| Alpha（学习率） | 开始收敛轮数（估计） |
| 0.5  0.4  0.6 | 700  600  550 |

实验发现Alpha衰减，对于加快收敛存在着明显效果。则后续实验采取该方法

### 20步重复机制，本机制对在转圈的问题，能进行更好的解决。（学习率：0.6，学习率衰减：0.995，gamma：0.99，epsilon：0.005，）

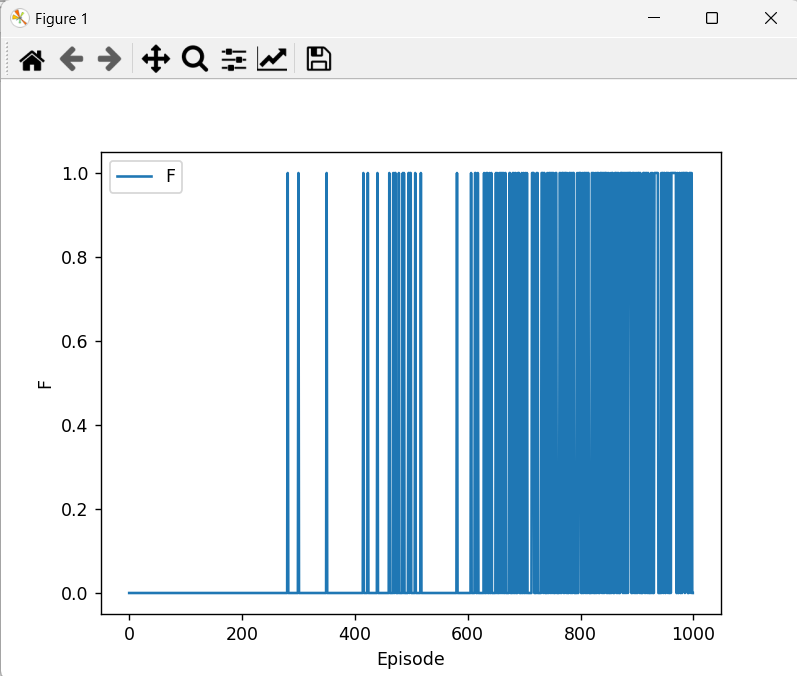
图 29 添加20步重复奖励机制reward=-5

图 30：添加20步重复奖励机制reward=-10

表格 9 控制其他不变，不同20步重复奖励机制，所对应的效果

|  |  |
| --- | --- |
| Reward | 开始收敛轮数（估计） |
| 0  -5  -10 | 550  400  600 |

# 总结与展望

在本次研究中，我们采用了Q-Learning算法对一个40x40的迷宫环境进行了智能体训练，并分别进行了不同参数的训练实验，记录了每个参数的寻找最优的详细历程，最终取得良好的效果。

智能体的学习能力由epsilon和学习率决定，在epsilon高的时候出现不收敛的情况，而学习率偏高时，能快速收敛，并且效果显著。我害在实验中采取了不同的奖励机制，并对奖励机制设置的不同奖励值进行探索,欧氏距离在全局和局部的效果，在探究全部区域的欧氏距离的训练过程中发现，agent到达不了终点，所以我考虑在终点区域采取欧氏距离方法，当与终点距离增加时进行惩罚，减少则相反，来探索收敛的最优解，结果发现，确实有对模型进行很好的提升。之后在训练的过程中发现，Agent在终点区域探寻终点的过程中，出现反复横跳，反复转圈的情况，对于寻找终点极为不利，所以我认为可能在欧氏距离的设置中，距离变大的奖励与距离变小的奖励，出现相消的现象，导致出现此现象，所以采取了不对称的奖励机制，进行验证。在探究欧氏距离（终点区域）的最优值后，在训练过程中还是会发现，agent到达终点后，收敛的还是不够快，因此为了让agent快速到达终点，我在添加欧氏距离（终点区域）的奖励机制基础上，进行添加全局区域的欧氏距离，发现效果也不显著。最后思考会不会是参数设置导致的问题。所以在alpha，gamma上进行参数探索，结果发现，确实有很大的影响，但在试验中一直贯穿着一个问题，智能体在终点区域寻找终点时，会一直出现转圈的现象，所以采取了20步重复奖励机制，最终解决完善了问题。

在探究的过程中就是如此反复，繁琐，但是探究出现效果时，就会带来快乐，从而增加探究的热情。

本研究通过Q-Learning算法在迷宫环境中的应用，验证了该算法在强化学习领域的有效性和潜力。通过对不同训练轮数下的实验结果分析，我们发现智能体在不断探索和学习过程中逐步优化其行为策略，最终能够高效地找到目标区域并避开障碍。未来，结合更多的优化方法和先进技术，Q-Learning算法有望在更复杂的环境和实际应用中发挥更大的作用。通过持续的理论研究和实践探索，强化学习技术将进一步推动智能体和人工智能领域的发展，带来更多创新和突破

# 参考文献

1. Jianqing Fan, Zhaoran Wang, Yuchen Xie, Zhuoran Yang, A Theoretical Analysis of Deep Q-Learning，Machine Learning (cs.LG); Optimization and Control (math.OC); Machine Learning (stat.ML)，arXiv:1901.00137 [cs.LG]
2. Watkins C J C H .Learning From Delayed Rewards[J].Robotics & Autonomous Systems, 1989.DOI:10.1016/0921-8890(95)00026-C.