CDS524 Assignment 1 - Reinforcement Learning Game Design

**Project Overview:**In this project, you will use Q-learning, a reinforcement learning algorithm, to design and implement your own game. The game can be any type of game you choose, from simple grid-based games to more complex simulations. The goal is to apply Q-learning to create a game where an agent (controlled by the algorithm) learns to make optimal decisions based on rewards and penalties.

**Project Requirements:**

1. **Game Design:**
   * You must design a game with a clear objective and set of rules.
   * The game should have a state space (all possible configurations of the game) and an action space (all possible actions the agent can take).
   * The game should have a reward function that provides positive rewards for achieving the objective and negative rewards for undesirable actions or states.
2. **Q-Learning Implementation:**
   * You must implement the Q-learning algorithm in a programming language you are comfortable with (e.g., Python).
   * The algorithm should be able to learn a policy (a mapping from states to actions) that maximizes the cumulative reward over time.
   * You should use techniques such as epsilon-greedy exploration, learning rate, and discount factor to balance exploration and exploitation.
3. **Game Interaction:**
   * You should create a user interface (UI) for your game that allows the agent to interact with the environment.
   * The UI should display the current state of the game, the agent's actions, and any rewards or penalties received.
   * You can use libraries or frameworks such as Pygame, Tkinter to create the UI and game environment.
4. **Documentation and Presentation:**
   * You should document your game design, Q-learning implementation, and evaluation results in a written report.
   * You should also prepare a video presentation to demonstrate your game and explain the Q-learning algorithm, the challenges you faced, and the solutions you implemented.

**Project Suggestions:**

* **Grid-Based Games:** Design simple grid-based games such as Tic-Tac-Toe, the Grid World problem, or a maze-solving game.
* **Board Games:** Implement classic board games such as Chess, Checkers, or Go, and use Q-learning to create an AI opponent.
* **Simulation Games:** You can design simulation games such as a traffic simulation, a resource management game, or a city-building game.
* **Action Games:** For high grade wanted students, you can design action games such as a simple platformer or a shooting game, and use Q-learning to create an AI-controlled enemy or player character.

|  |  |  |
| --- | --- | --- |
| Rubric | Description | Marks |
| **1. Game Design (10 marks)** | Written | 10 |
| 1.1 | Clear objective and set of rules defined | 2 |
| 1.2 | State space and action space well-defined | 2 |
| 1.3 | Reward function correctly implemented with positive and negative rewards | 3 |
| 1.4 | Documentation of game design (including working diagrams if applicable) | 3 |
| **2. Q-Learning Implementation (10 marks)** | Written & Development | 10 |
| 2.1 | Q-learning algorithm implemented correctly in chosen programming language | 3 |
| 2.2 | Policy learned maximizes cumulative reward over time | 3 |
| 2.3 | Effective use of epsilon-greedy exploration, learning rate, and discount factor | 2 |
| 2.4 | Documentation and explanation of Q-learning implementation | 2 |
| **3. Game Interaction (10 marks)** | Development part | 10 |
| 3.1 | User interface (UI) allows agent to interact with environment | 3 |
| 3.2 | UI displays current state, agent's actions, and rewards/penalties | 3 |
| 3.3 | Appropriate use of libraries/frameworks for UI and game environment | 2 |
| 3.4 | User experience and interaction smooth and intuitive | 2 |
| **4. Documentation and Presentation (10 marks)** | Written & Presentation | 10 |
| 4.1 | Written report documenting game design, Q-learning implementation, and evaluation results | 4 |
| 4.2 | Video presentation demonstrating game and explaining Q-learning algorithm | 3 |
| 4.3 | Challenges faced and solutions implemented clearly explained in both report and presentation | 2 |
| 4.4 | Overall quality and clarity of documentation and presentation | 1 |

Deliverables

1. A Google Colab (or equivalent) with your code and in-line explanations.
2. A brief report (1000-1500 words) evaluating the performance of your work.
3. Github repo
4. Youtube Demo Video

Submission Details:

1. You must ensure that all your project files used for this task and the report sit in a directory called “Assignment 1 – Your Name”.
2. All files are required to be uploaded and a link to the “Assignment 1” directory submitted to Moodle.
3. Please make sure that unit Instructor and TA have access to the folder.
4. A **link** to the demo video of your app running **must be submitted**.
5. It would be great if you could submit your **GitHub** link.
6. This is an **individual** assignment, and you should submit it **by 8 pm, Tuesday, Week 6**.

Reference

1. [deep-q-learning-snake-game](https://github.com/happy531/deep-q-learning-snake-game)
2. [基于Q-learning 强化学习的贪吃蛇游戏（源码+论文）](https://developer.aliyun.com/article/1581658)
3. [[Tetris-deep-Q-learning-pytorch](https://gitcode.com/gh_mirrors/te/Tetris-deep-Q-learning-pytorch/?utm_source=artical_gitcode&index=top&type=href&)](https://gitcode.com/gh_mirrors/te/Tetris-deep-Q-learning-pytorch)