Computer Science Internal Assessment

May 22, 2020

Contents

1 Criterion A : Planning

1.1 Defining the Problem

The client, Openai, is a software company that focuses on researching the practical uses of Deep Q learning, a type of reinforcement learning. At the start of 2018 they posted seven problems for the public to solve. Out of the seven problems I chose to do the ‘Slithern” problem. The ‘Slithern” problem asks to “Implement and solve a clone of the classic Snake game (see slither.io for inspiration) as a Gym environment.” In order to solve the problem Openai includes expectations about the environment as well as the agent. The environment must have the following:

- A reasonably large field
- Snakes' dies when colliding with another snake, itself, or the wall
- The game ends when all snakes die

The Agent must have the following:

- Must use a reinforcement learning algorithm

1.2 Rationale for Proposed Solution

In order to solve the problem posed I will create a program. There will be two main parts of the program, the environment and the agent. I will also use a number of libraries in the program in order to handle the data as well as manipulate it. Finally, the program will be written in python. The environment will have classes for objects such as snake, apple, and an environment. The snake and apple objects will be used to store information about each object allowing for modularity. The information these classes will be storing will be a user created class to allow for special manipulation. The environment class will create objects that follow the Gym structure. This means that they will have initializer methods, step methods, render methods, and reset methods. Other than the previously stated method there will be methods to manipulate data. The agent will be structured off of the q learning formula. I am using python to create the program for a multitude of reasons. Python is exceptional at handling data, great for machine learning applications. This makes it the main language used in machine learning applications like reinforcement learning. As a result of the support in this field there is several libraries that are required to create this program, some of which are essential to running machine learning efficiently. Openai also is based on python and to satisfy the Gym requirement the program must be written in python. Finally, I am using python because I have a lot of experience with it.

1.3 Criteria for Success

This program will create both an environment, following the rules of a snake game, and an agent which will be used to solve the environment.

1.3.1 Enviroment

- Train agent to play snake
- Read config file for settings
- Log data during run
- Dump run data

1.3.2 Render

- Read config file for settings
- Render videos from data

1.3.3 Config File

- All settings are configurable

1.3.4 GUI

- Display video of run
- Display graph of performace
- Switch between video and graph
- Display all runs
- Choose run by selecting from menu

2 Criterion B : Record of Tasks

Task #	Planned Action	Planned Outcome	Tim
1	Finding client	Found client with problem	2 we
2	Discuss problem with client	Identify client's requirements so that a solution can be designed	3 da
3	Research about problem	Identify effective ways to addres the problem by researching examples	4 we
4	Define critera for success	Idneitify critera's for success that are based on the clients requirements	1 we
5	Create rapid prototype	Understand how to implement tools needed to make the designed solution	1 we
6	Draw UML diagrams and flow charts to further understanding	Diagrams are created and knowledge to implement solution is increased	2 we
7	Code rough snake game	Rough snake game to act as base for snake enviorment as this allows me to design the enviormentby modifing the snake game instead of from scratch	1 we
8	Adapt snake game to enviorment	Snake enviorment is created in order to start desining the agent for it	2 we
9	Create agent for snake enviorment	Agent for enviorment is created	2 we
10	Optomize agent and enviorment	Agent is successful in playing snake	2 we
11	Implment GUI for playing runs	GUI is created able to render runs	1 we
12	Fixed error where apple can spawn on snake		

3 Criterion B : Design

3.1 Design of the Solution

3.1.1 Internal Structure

1. Classes

   (a) Cell All other classes consist of cells. Cells will be used because it will provide an elegant way to move game pieces around as if they were in a 2D space.

   (b) Snake The snake class extends deque and is a container for all the Cells in the snake. The snake class will handle snake movement and other actions.

   (c) Apple The apple class extends the Cell class and will enable apples to be randomly placed anywhere within the game area.

2. UML Class Diagram

[width=.9]/home/jrobertboos/Documents/Computer Science/internal-assessment-computer-science/diagrams/class

3.1.2 External Structure

1. Enviroment

   (a) UML Activity Diagram

   [width=.9]/home/jrobertboos/Documents/Computer Science/internal-assessment-computer-science/diagrams/proccess

2. Render

3. GUI

3.1.3 Description of Specific Elements

I used many different non-standard algorithims within my program as shown below.

1. Q learning Formula: In order to meet the clients specification I must use a reinforcement learning algorithim. Specificaly, I chose to use the Q learning algorithim as it is one of the most common algorithims used in reinforcement learning. Q learning is an algorithim used to learn a policy and tell an agent what to do. My agent would solve the enviroment through the implementation of the equation into the code. The equation can be found below.

2. Configuration File: As editing constants in the Q learning formula is very important to solving the enviroment and optomizing the agent, a configuration file would be a very useful addition. A configuration file would allow many different variable sets to be stored without having to overwrite them as well as having all variables for all the methods in one location. The configuration file would be a json as python offers a number of powerful features when reading json.

3. Deque Data Type: The program makes use of the deque data type to store the segments of the snake so that the segments would be stored in a double ended queue. Double ended queues have two distinct advatages, being able to append and pop from either end of the queue, and appending and popping much more effecitley. This makes the performace of the snakes movements much more concise and optimized.

3.2 Testing Methodology

Test #	Details
1	Test if A
2	gent is trained on enviroment Test wether the configuation file is parsed successfully by the
3	Test if while training the agent logs are produced in the console
4	Test wether data is being succesfully dumped by the enviroment
5	Test if Render is parsing the configuration file correctly
6	Test if Render correctly produces videos
7	Test wether GUI displays video
9	Test if GUI switches between video and graph views
8	Test wether GUI displays graph
10	Test if GUI can select runs