Developer Guide

1. Introduction

The Reinforcement Learning Model for games is a project focused on demonstrating how reinforcement learning can be used to train artificial intelligence and how an AI taught with RL will behave using risk-avoiding and reward-seeking behavior. This is done in a simple, engaging way by having AI designed to play multiple board games against a human opponent. The AI will play Tic-Tac-Toe, Connect Four, and Dots and Boxes. It will play in a single match format or in a league match with betting to further simulate reward-seeking behavior.

Reinforcement Learning is a form of machine learning in which focuses on the relationship of an AI agent and its environment. The AI is taught through reward, and it builds an understanding of the relationship between taking certain actions under certain conditions and the rewards that will result.

Much of this project is focused on providing a GUI and front-end for the workings of the AI. The AI's methods are contained in Python scripts and AI q-tables. Meanwhile, the GUI and front-end is provided by Kivy files that hook up to Python. Kivy is a framework built for Python that provides simple and straightforward GUI.

The project code repository can be found at https://github.com/mammalwithashell/WJNKCW

Additional information can be found in the readme

Full documentation can be found in the doc folder

2. Tools Required:
   a. Python 3.7 or later
      1. Kivy Python module
      2. Numpy Python module

2. Development tools:
   a. Android Studio to debug Android apks
   b. MacOSX is needed to debug iOS apks
   c. Visual Studio Code

3. Current Issues/Things To Be Aware Of:
   a. Mac users will need to right-click on the package and open, since it is unsigned. It will not open from a double-click

4. Command Line Information to Begin project

Getting Started

- Set up a Python virtual environment
- Activate the environment (venv activation is os specific)
  - `python -m env <env_name>`
- Clone the repository

    - `git clone https://github.com/mammalwithashell/WJNKCW.git`
- Install the requirements
    - `pip install requirements.txt`
- Run the project
    - `python main.py`

<u>Building the Project for Windows</u>

- Install PyInstaller
    - `pip install pyinstaller`
- Install from reinforcement.spec
    - `pyinstaller reinforcement.spec`

5. Code Design

The project is located in the main.py and imports logic and assets from other folders in the project repository.

a. Game_Logic: The logic that makes the games work is contained in these folders. All of the Python files, scripts, and code to run the games and game screens can be found here.
    i. <game_name>AI: Sub-folders containing the Python scripts for the Reinforcement Learning agent and environments for each game.
        1. qtables: Folder containing the q-tables that describe the AI's behavior.
        2. Agent.py: This file reads the qtables and selects the next action for the computer based on the board's current state. For the connect 4 game, this file also controls the computers actions for the league functionality.
        3. BoardEnviornment.py: The BoardEnvironment class in this file represents the state of the game board as a string and houses the necessary logic to play each game.
        4. LeagueEnvironment.py: The LeagueEnvironment class serves to keep track of the logic and state of the league betting functionality.
    ii. cmd&b.py:
        1. Running this file allows you to play the command line dots and boxes game against the qtables
    iii. connect4.py, dotsandboxes.py, tictactoe.py
        1. Each file contains the screen class that houses the kivy logic for each game. Drawing the games, updating the BoardEnvironment based on user behavior, etc is handled in these files.
    iv. Utils.py
        1. Utility functions necessary for compiling MacOS and Windows executables
b. Design: The markup files for Kivy (.kv files) go into this folder. Kivy uses these files in the same way that a browser uses html files. They tweak and change the GUI

used for the game files and scripts. Refer to our Kivy Guide for tips to understanding the kivy library

   i. connect4.kv: The design file for connect 4
   ii. dotsandboxes.kv: The design file for dots and boxes
   iii. gui.kv: The design file for the title screen
   iv. tictactoe.kv: The design file for the tic-tac-toe

c. Main.py
   i. RootWidget
   ii. Load_game passes the game settings to the game scenes and swaps to the right screen based on user input

6. Road Map for the future
   a. Adding sounds to the game
   b. Packaging for android and ios
   c. League Betting Completion
   d. Networking/ Play with others for tic tac toe
   e. Faster loading of qtables (large text files)