Project 3: Training A DQN Agent to Play Pong Game

TA: Tianze Chen tianzechen@usf.edu

1. Introduction

Recent advances in deep neural networks have shown great promise for solving high-dimensional problems. Deep reinforcement learning (DRL) has shown great promise for solving complex problems and has been demonstrated to work in the games domain by using deep convolutional neural networks within the reinforcement learning framework to represent action-value functions. This method is called Deep Q-learning, and it learns to solve Atari games from video input with state-of-the-art results. Further developments have led to more state-of-the-art results in various games, from AlphaGo [1] to MuZero [2]. These results have led to increased interest in this field of study. During the past few years, a lot of research has been done on DRL development, including but not limited to areas such as gaming, image and video understanding, robotics, and smart appliances.

Pong is a table tennis-themed arcade video game featuring simple two-dimensional graphics, manufactured by Atari and initially released in 1972. In the pong game, one player scores if the ball passes by the other player. An episode is over when one of the players reaches **21 points**. Figure 1 shows the interface of the pong game.

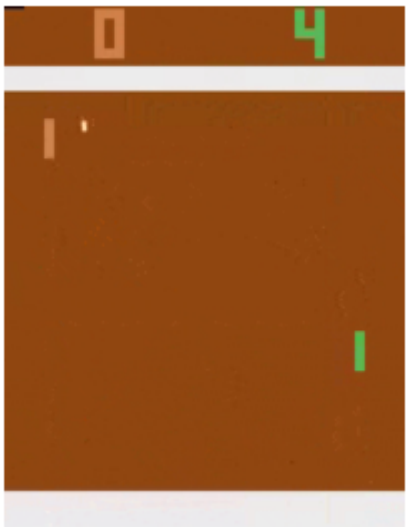


Figure 1. Pong game

2. Project Goal

In this project, students will design a deep Q-Learning agent to play pong. The player is on the right side and can perform three motions: stay, move upwards, and move downwards. The students need to train two agents: one agent is the standard Q-Learning agent with target network, the other agent is with double Q-Learning. The students need to compare the performance between the two agents. Pytorch and OpenAI gym is required for this project.

3. Dataset

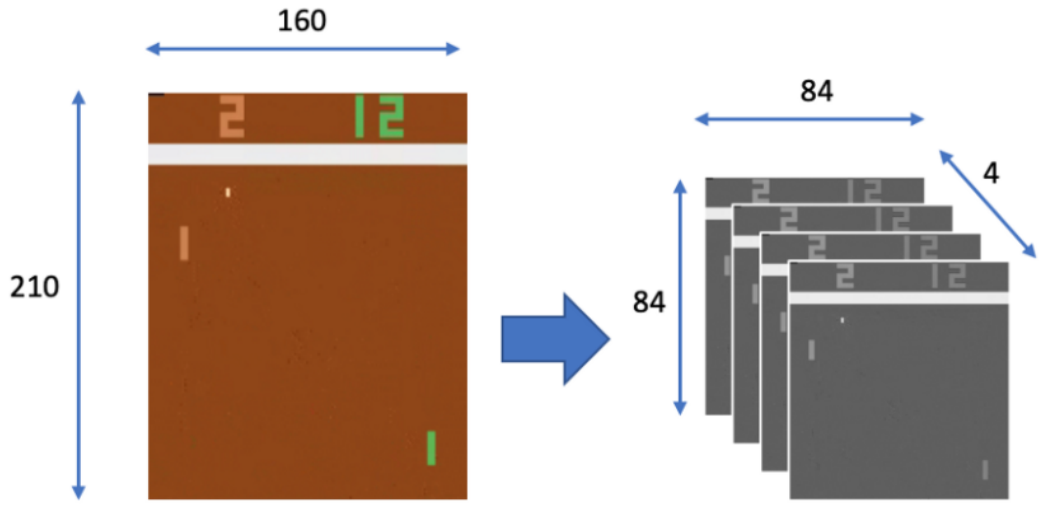
This project will not have a dataset, and the students will build the experience replay buffer during the training.

4. Input and Output

The state, action, and reward are defined as follows:

State: Image of the game interface (RGB image with dimension of (210, 160, 3))
Action: Stationary, move-up, move-down
Reward: The students can define their reward function. E.g., -1 if do not catch the ball and +1 if catches the ball.

Since the input image size is large, it is recommended to convert the RGB image to a greyscale image with only one channel to increase the training speed. Also, you can stack several continuous steps to the channels to represent the state. E.g., convert the image from RGB (210, 160, 3) to greyscale (210, 160, 1) and stack four continuous steps together (210, 160, 4). The input into the deep reinforcement learning agent is a state. The output of the agent is the action that gives the highest accumulated Q value. Code example of the process the images from pong into greyscale images are provided.



5. Network Architecture and Training

The students can freely design their network structure and choose a mini-batch size and replay buffer size. Since we define the states as images, convolutional layers may be helpful.

6. Additional Material of Setup the Environment

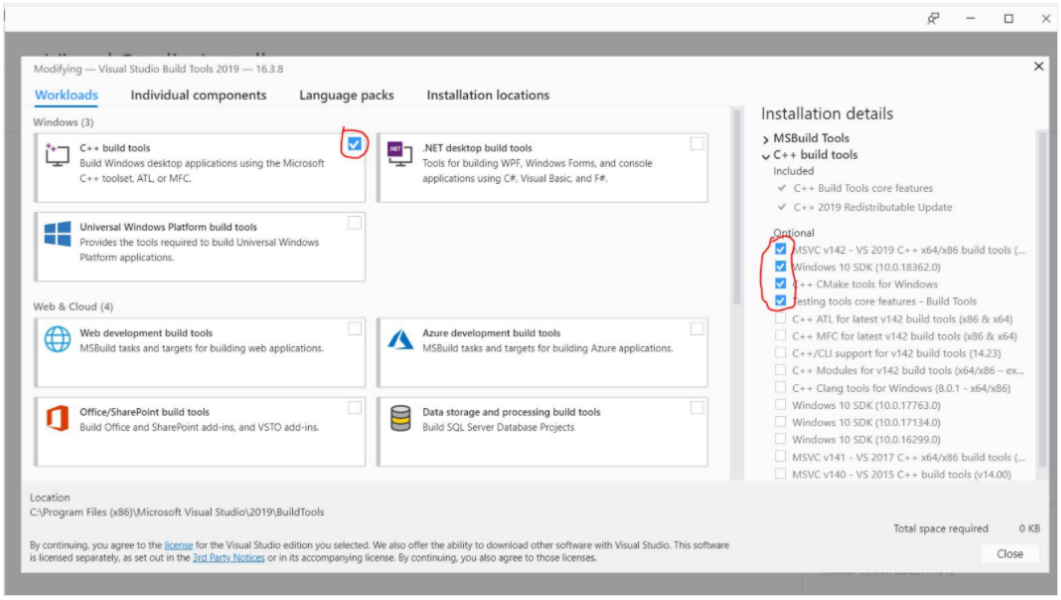
Install gym:
https://gym.openai.com/docs/
pip install gym
pip install atari-py
pip install gym[atari]

OpenAI gym wrapper for Atari games:
https://github.com/openai/baselines/blob/master/baselines/common/atari_wrappers.py

For Windows users, before installing gym, a few more steps need to be done:

a. Download VS build tools:
https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=BuildTools&rel=16

b. Run VS build tools and install the following tools:



c. Restart the computer and install the gym packages

7. **Deadline**
   May 3, 11:59 pm

8. **Grading Rubric:**

Points	Section	Description
70 pts	Deep Q-learning Network Design & Training	Design deep neural network for Q-learning. Design proper loss functions based on the requirements. Use the OpenAI Gym to get training data. Train the deep Q-learning networks.
30 pts	Performance	You should use your trained Q-learning networks to compete in the Pong game with the machine. The grade will depend on the performance of your trained DQN agent in the Pong game.

9. Submission
   - Your training code (including the training code for two different Q learning approaches)
   - Your trained Q-learning network
   - The final replay buffer used for training
   - A readme on how to use your code for training and your network in OpenAI Gym.

References

[1] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al., "Mastering the game of go with deep neural networks and tree search," nature, vol. 529, no. 7587, p. 484, 2016.

[2] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, et al., "Mastering atari, go, chess and shogi by planning with a learned model," arXiv preprint arXiv:1911.08265, 2019.