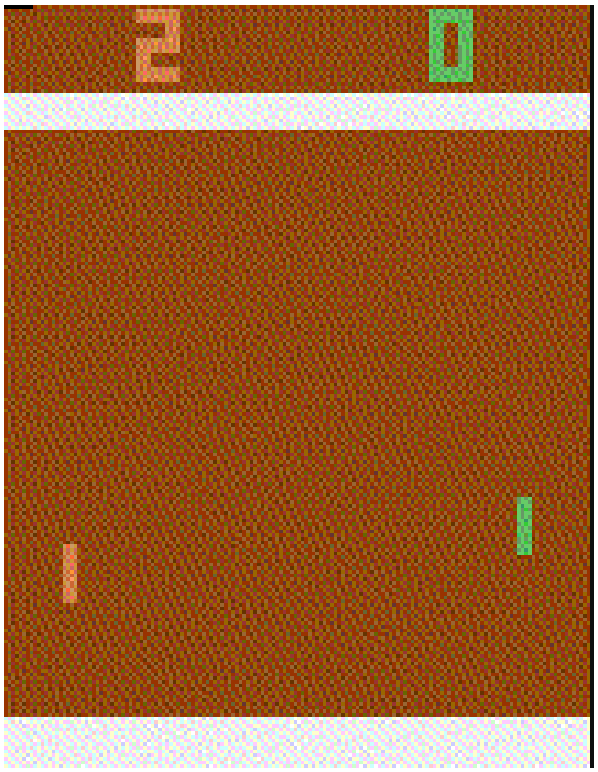
Assignment 3

Reinforcement Learning Programming - CSCN 8020

November 13, 2023

Introduction

You will work with the Pong environment and implement Deep Q-Learning. In this environment, if we got the right and left paddle's positions as the state, we would have continuous states that prevent the use of a tabular solutions, and thus the use of Q-Learning. To solve this problem, you will use a neural network to represent the function with Deep Q Networks and use the game frames as the observation. The way you interact with the environment will be very similar to the Ms. Pacman gym environment used in class. Therefore, most of the code we discussed is directly applicable. You will be using the agent on the right.



Action space

- 0: NOOP (No operation)
- 1: FIRE
- 2: Move right
- 3: Move left
- 4: Right Fire
- 5: Left Fire

Observation Space

The environment has an observation space of size $(210, 160, 3)$, which is an RGB image with values between 0 and 255.

Do not change the difficulty or mode, the default is set to 0 for both. Also, ensure you're using the **PongDeterministic-v4** version of the environment, which has the reduced set of actions.

Helper Utility

To assist you in interacting with the environment, you were provided with a file (*assignment3_utils.py*) that has a few methods to allow pre-processing of the image: cropping, switching to grayscale, and downsizing.

Important Notes

Do not fully count on the existing implementation from class, but you can use it as a starting point. However, **you may need to restructure the code, change the learning loop, or even the CNN architecture for proper learning.**
Change the input size of the CNN and use all 4 images as an input instead of blending them together.
You may need to install atari by running
$pip3 \quad install \quad gym[atari]$

DQN [100]

Problem Statement

Implement the DQN algorithm on the Pong environment from OpenAI Gym. Train an agent to efficiently play the game (And win!). Use the following hyperparameters:

- Mini-batch size: 8
- Update rate of target network: 10 episodes
- Discount Factor γ: 0.95
- Exploration:
  - Exploration Factor Initial Value ϵ_{init}: 1.0
  - Exploration Decay Rate (δ): 0.995
  - Exploration minimum value ϵ_{min}: 0.05
  - Calculate the exploration rate using:

$$\begin{aligned} \epsilon &= \epsilon * \delta \quad if \quad \epsilon \geq \epsilon_{min} \\ &= \epsilon_{min} \quad otherwise \end{aligned}$$

Tasks

- Implement the DQN algorithm and train an agent on the Pong environment.
  - Change the input size of the CNN and use all 4 images as an input instead of blending them together.
  - Note that if you apply the crop function in the utilities, you'll need to change the input state size to 84x80 instead of 105x80.
- Report the following metrics during training with the number of steps:
  1. Score per episode
  2. Average Cumulative reward of the last 5 episodes
- Plot the deliberate change to the following parameters SEPARATELY and plot a figure of the previous metrics for each change as follows:
  - Plot figure with changing the mini-batch size: [8 (default), 16]
  - Plot figure with changing the update rate of the target network to be every [3, 10 (default)] episode

Deliverables

- Python code implementing DQN and running it for the initial hyper-parameters.. [30]
- A report (in PDF) containing (After running the different batch sizes and update rates mentioned):
  - Describe the final network architecture you used. [5]
  - The metrics, observations, and comments on the parameter change. Make sure you include appropriate plots for the previously stated metrics to support your arguments. [45]
- Based on your findings, choose what you think would be the best combination of batch size and update rate for target network. [20]