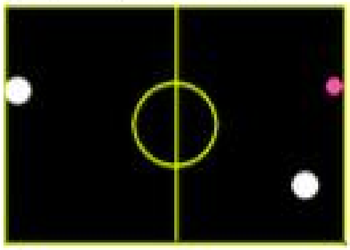
HOCKEY GAME

王竑睿 B05502087　　吳由由 B06902104　　吳采耘 B06902041



INTRODUCTION

Hockey Game is a classical game in which two players play against each other by trying to make the ball touch the opponent's edges with a disk.

CNN

1. Convolution: Line up the feature and the image. Multiply each image pixel by corresponding feature pixel. Add the values and find the sum. Divide the sum by the total number of pixels in the feature.
2. Rectified Linear Unit (ReLU) transform function only activates a node if the input is above a certain quantity.
3. Pooling Layer shrinks the image stack into a smaller size.
4. Fully connected layer is the classification layer. When certain values are arranged the way they are, they can be mapped to an actual object which we require.

EXPERIMENTAL METHODS

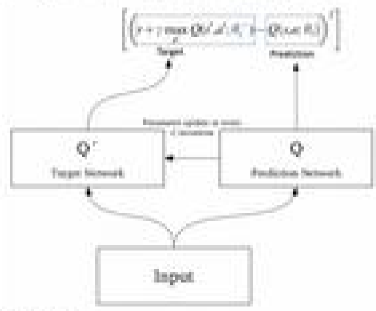
- We preprocessed the images by converting to grayscale, resizing them to 80x80, and then stacked together the last four frames to produce an 80x80x4 input array.
- The output layer, obtained with a simple matrix multiplication, has the same dimensionality as the number of valid actions which can be performed in the game. The values at this output layer represent the Q function. At each time step, the network performs whichever action corresponds to the highest Q value using a ϵ greedy policy.
- At startup, we initialize all weight matrices randomly using a normal distribution with a standard deviation of 0.01. Bias variables are all initialized at 0.01. We then initialize the replay memory with a max size of 500,000 observations.
- We start training by choosing actions uniformly at random for 50,000 time steps, without updating the network weights. This allows us to populate the replay memory before training begins.
- After that, we linearly anneal ϵ from 1 to 0.05 over the course of the next 500,000 frames. During this time, at each time step, the network samples minibatches of size 100 from the replay memory to train on, and performs a gradient step on the loss function with a learning rate of 0.000001. After annealing finishes, the network continues to train indefinitely, with ϵ fixed at 0.1.

DEEP Q-NETWORK

Q-value:

- From being at state s and performing action a is the reward r(s,a) plus the highest Q-value from the next state s'. Gamma is the discount factor which controls the contribution of rewards. Alpha is the learning rate or step size.

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)\right]$$



Deep Q-Network:

- All the past experience is stored by the user in memory
- The next action is determined by the maximum output of the Q-network
- The loss function here is mean squared error of the predicted Q-value and the target Q-value – Q*.

REFERENCES

1. Anirudh Rao. (May 29,2019). Convolutional Neural Network Tutorial (CNN) – Developing An Image Classifier In Python Using TensorFlow. https://www.edureka.co/blog/convolutional-neural-network/
2. Akshay Srivatsan, Ivan Kuznetsov, Willis Wang. (21 Jun 2016). Using Deep Q Networks to Learn Video Game Strategies. https://github.com/asrivat1/DeepLearningVideoGames
3. ANKIT CHOUDHARY. (APRIL 18, 2019). A Hands-On Introduction to Deep Q-Learning using OpenAI Gym in Python. https://www.analyticsvidhya.com/blog/2019/04/introduction-deep-q-learning-python/