Nathan Mohapatra

Ian Davidson

ECS 170 001

18 March 2021

Programming Assignment 3: Report

---

1 Problem Representation

1. For the Q learner we must represent the game as a set of **states**, **actions**, and **rewards**. OpenAI offers two versions of game environments: one which offers the state as the game display (image) and one that offers the state as the hardware ram (array). Which do you think would be easier for the agent to learn from and why?

   I think the game environment which offers the state as the game display (image) would be easier for the agent to learn from, because the hardware ram (array) for an Atari 2600 was 128 bytes, and it is questionable whether or not this is enough information to be able to learn from. Hence, the starter code is designed such that the game environment offers the state as the game display (image).

2. **Use the starter code to answer this question.** Describe the purpose of the neural network in Q-Learning. Neural networks learn a complicated function mapping their inputs to their outputs. What will the inputs and outputs for this neural network be? What do these variables represent? The neural network in the starter code is `class QLearner(nn.Module)`.

The purpose of the neural network in Q-Learning is to incorporate function approximation algorithms (i.e. back-propagation); a neural network substitutes for the lookup table and uses each $Q\hat{}(s, a)$ update as a training example. The neural network is trained with the game display (image) as input and six $Q\hat{}$ values as output. The game display (image) represents the state s and the six $Q\hat{}$ values are for each action a.

3. What is the purpose of lines 48 and 57 of dqn.py (listed below)? Doesn't the Q learner tell us what action to perform?

```
if random.random() > epsilon:
      …
else:
      action = random.randrange(self.env.action_space.n)
```

The purpose of lines 48 and 57 of dqn.py is to choose between two strategies for selecting an action to execute in the current state, exploration versus exploitation: the agent either explores new paths to rewards by randomly selecting an action (with probability *epsilon*), or exploits what it has learned so far by selecting the action corresponding to the largest Q value (with probability 1 - *epsilon*). Thus, how frequently either strategy is used is determined by the value of *epsilon*; ideally, the value of *epsilon* decreases during training, because exploration is preferred at the beginning and exploitation is preferred at the end. The Q learner only tells us what action to perform when we use the exploitation strategy.

2 Making the Q-Learner Learn

1. Explain the objective function of Deep Q Learning Network for one-state lookahead below; what does each variable mean? Why does this loss function help our model learn? This is described in more detail in the Mitchell reinforcement learning text starting at page 383. We've provided the loss function below. This should be summed over the batch to get one value per batch.

$$Loss_i(\Theta_i) = (y_i - Q(s, a; \Theta_i))^2$$

Hint: Θ is convention for "model parameters" and y is convention for "model output." The subscript i refers to "the i-th iteration". If you are stuck, research "mean squared error."

The objective function of Deep Q Learning Network for one-state lookahead is to iteratively reduce the discrepancy between Q value estimates for adjacent states.

Θ_i : neural network weights for iteration i

y_i : target Q value for iteration i

$Q(s, a; \Theta_i)$: predicted Q value for iteration i

→ The Q value is the reward received immediately upon applying action a to state s, plus the value (discounted by γ) of following the optimal policy thereafter.

This loss function helps our model learn because, under certain assumptions, it converges to the optimal Q function (and, unlike supervised learning, the targets are not provided and fixed beforehand).

4 Learning to Play Pong

4. Plot how the loss and reward change during the training process. Include these figures in your report.