Project 1 Navigation Report

Alvin (Cheuk Hin) Li

April 2020

1 Introduction

In this environment, the agent would be responsible for exploring the world and collecting yellow bananas, while avoiding blue bananas. A +1 score would be assigned if the agent collects a yellow banana, while a -1 score would be assigned if the agent collects a blue banana. The score if accumulated over the maximum 1000 timestamp. The assignment is considered solved if the agent has an average score of +13 over 100 consecutive episodes.

The state is defined by 37 variables with the agent's and its surrounding's information. The agent can choose to move forward, backward, left or right at any given timestamp.

2 Learning Algorithm

The chosen algorithm is Double Deep Q-Learning with Dueling Network Architecture. In the common Deep Q-Network, the error is calculated by [1]:

$$error = r_{t+1} + \gamma max_a Q(S_{t+1}, a; \theta_t^-)$$

However, by using the max value in the equation, the model is more vulnerable to overestimated values [2]. In a double deep Q-learning algorithm, the error is calculated by:

$$error = r_{t+1} + \gamma Q(S_{t+1}, argmax_a Q(S_{t+1}, a; \theta_t); \theta'_t)$$

How is this different? The action is evaluated by two parameters: θ and θ'. The two parameters must both speak highly of the action in order for the return to be high, preventing the overestimation problem. In the dueling network architecture, instead of the traditional fully connected layers in the end, the fully connected layers are split into two distinct branches. One branch calculates the state values while the other calculate the advantage values [3]. The split estimators can learn whether all the states are equally as valuable as others. The dueling architecture is consists of two dense layers performing on the x variable. On one side, the second layer returns just one state value, which is the value function. On the other side, the second layer returns a vector the size of the number of actions possible (ie. 4). They are then returned with the following operation:

$$Q = V + A - mean(A)$$

Epsilon starts at 0.1 and ends at 0.01, with a decay of 0.996. Batch size is set at 64, gamma value at 0.99, tau at 1e-3, learning rate at 5e-4 and the network is updated every 5 episodes. The epsilon decay has been changed to 0.99 and 0.999, both producing suboptimal results. Update per episodes has been increased to 5, which saw a significant decrease of time needed to reach the goal.

3 Plot of Rewards

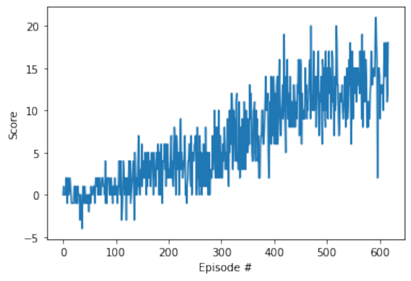

Figure 1: Rewards per Episode

The environment was solved within 517 episodes, with the final average score of 13.05.

4 Ideas for Future Work

In the future, prioritized experience replay can be used in conjunction with other methods as used in this project [4]. This would result in a similar model to the rainbow model, which has a significantly better outcome than the standard deep-Q model. Also, in the paper by Wang et. al., a different way of adding the value and advantage functions is proposed. The max of the advantage function is used instead of the mean. It would be interesting to see how that would have impacted the learning of the agent.

5 References

[1] https://www.nature.com/articles/nature14236
[2] https://arxiv.org/pdf/1509.06461.pdf
[3] https://arxiv.org/pdf/1511.06581.pdf
[4] https://arxiv.org/abs/1511.05952