COMP9444 Project 3 Report

Group number: g017380

Name: Xuhua Le, student number: z5047516

Name: Xinyu Wang, student number: z5143329

In this project, we are going to implement a Deep Reinforcement Learning algorithm especially Q-Learning to handle a cart-pole control task. The aim is to try to control movement direction and velocity of the cart to try to prevent the pole from falling over the cart. As this is a deep reinforcement learning task, we could get a reward for every cart movement including the terminated step, and we need to implement our processes to maximize the total rewards for each training step.

As we would use Q-Learning algorithm, in order to start, we build a basic network with a hidden layer, and the input is state_in and has action_dim outputs which are the network's estimation of the Q values for those actions and the input state. In other words, this network gets input of state_in and then gets to the hidden_layer and finally goes to the Q value layer which would get the variable q_values ,q_action, loss and optimizer.

We also define a function named “perceive” to store training information and control whether to continue training based on the quantity of training information.

Besides, the “explore” function in the skeleton code has implemented the process that whether to choose random actions or choose actions from neural network outputs based on epsilon value, in other words, epsilon is correlative to the possibility that random explore or learn from network outputs.

As the skeleton code has a main learning loop, we decide to put our training process in the main learning loop, at each training process, we would randomly choose some previous training information with a quantity of batch_size to have a guidance to update our Q value based on Bellman Equation.

Here is the main training process: for every episode, we would refresh the training environment, and we would choose random actions or actions from neural network outputs based on epsilon value, and the epsilon value would be 1 at the beginning as we have no training information before, and it get decreased along with the training times to remain stability of our training process. After every training process, we would store the training information for future usage and update the Q value, and we would calculate the total reward as well.