MLDS 490 Lab 5

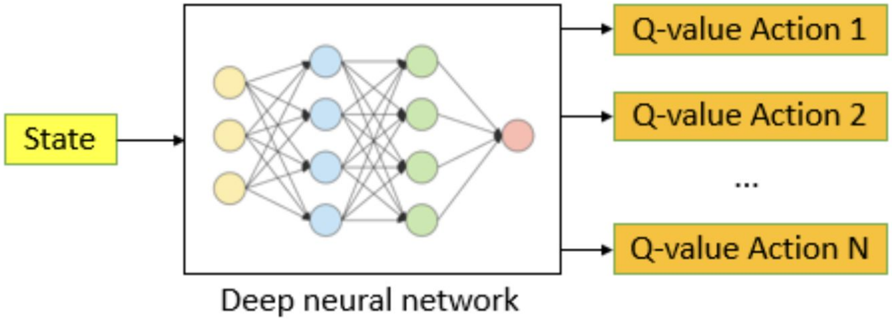
Shuyang Wang Fall 2023

Deep Q-Network

Recall the definition of **Q-factor** (or Q-function, action value function)

$$Q(s,a) = r(s,a) + \gamma \max_{a'} \mathbb{E}_{s' \sim p(s'|s,a)} Q(s',a')$$

Deep Q-Network algorithm models the Q-function using a deep neural network, which outputs a vector of Q-values (one number for each action).



Epsilon-Greedy Policy

No policy *network* is used for DQN. We follow a policy using the Q-function:
At each time step t, take an action

$$a_t = \begin{cases} \text{randomly selected action } a & \text{with probability } \epsilon \\ argmax_a Q(s_t, a) & \text{with probability } 1-\epsilon \end{cases}$$

How to select maxQ(s, a) with a Q-Network?

with probability 1-epsilon:

1. Do a forward pass of the Q-Network with input s_t: $Q_\theta(s_t)$
2. From the output vector $v_t = [Q(s_t, a_1), ..., Q(s_t, a_N)]$
   select the action with the **maximum Q-value**

Deep Q-Network

Algorithm 1 Deep Q-Network

Initialize Q-network weights θ
Initialize the replay buffer $\mathcal{B}$ with capacity D.
Initialize target network $\hat{Q}_{\theta'}$ weights $\theta' = \theta$
for episode=$1, ..., M$ **do**
    for $t = 1, ..., T$ **do**
        Based on the current state s_t, choose action a_t according to ϵ-Greedy policy.
        Take action a_t, collect a tuple (s_t, a_t, r_t, s_{t+1}) and add to the replay buffer $\mathcal{B}$.
        Randomly sample a mini-batch of tuples $\mathcal{S} = \{(s_i, a_i, r_i, s_{i+1})\}$ from the replay buffer $\mathcal{B}$.
        for each tuple (s_i, a_i, r_i, s_{i+1}) in the batch $\mathcal{S}$ **do**
            Compute y_i:
            if episode terminates after step i **then**
                $y_i = r_i$
            else
                $y_i = r_i + \gamma \max_{a'} \hat{Q}_{\theta'}(s_{i+1}, a')$ using the target network with weights θ'
            end if
        end for
        Update θ by taking an optimization step on the loss $L(\theta) = \sum_i (Q_\theta(s_i, a_i) - y_i)^2$
        Update θ' after every N optimization steps.
    end for
end for

Key components

- Q-network
- Replay buffer: deque (`from collections import deque`)
- Target network

About deliverables:

1. Q plot: for each episode, compute the **average** of **max Q values** in this episode, and plot it against the number of episodes.
2. Reward plot: plot the sum of rewards (without discounting) in an episode, overlay with the moving average.