Theory Exercises

Problem 1 - Graph Convolution

(a) The adjacent matrix A of the graph is given as :

	0	1	2	3
0	1	1	0	0
1	0	1	1	1
2	0	1	1	0
3	1	0	1	1

(b)

$$X = \begin{bmatrix} 1 & -1 \\ -2 & 0.5 \\ 1 & 3 \\ 0 & -1 \end{bmatrix}$$

$$AX = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & -1 \\ -2 & 0.5 \\ 1 & 3 \\ 0 & -1 \end{bmatrix}$$

$$AX = \begin{bmatrix} -1 & -0.5 \\ -1 & 2.5 \\ -1 & 3.5 \\ 2 & 1 \end{bmatrix}$$

$$x' = \sigma(AX)$$

$$x' = \sigma\left(\begin{bmatrix} -1 & -0.5 \\ -1 & 2.5 \\ -1 & 3.5 \\ 2 & 1 \end{bmatrix}\right) = \begin{bmatrix} 0 & 0 \\ 0 & 2.5 \\ 0 & 3.5 \\ 2 & 1 \end{bmatrix}$$

c)

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{bmatrix} \quad D = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix}$$

$$D^{-1} = \begin{bmatrix} 1/2 & 0 & 0 & 0 \\ 0 & 1/3 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 1/3 \end{bmatrix}$$

$$D^{-1}AX = \begin{bmatrix} 1/2 & 0 & 0 & 0 \\ 0 & 1/3 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 1/3 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & -1 \\ -2 & 0.5 \\ 1 & 3 \\ 0 & -1 \end{bmatrix}$$

$$D^{-1}AX = \begin{bmatrix} -1/2 & -1/4 \\ -1/3 & 5/6 \\ -1/2 & 7/4 \\ 2/3 & 1/3 \end{bmatrix}$$

$$\sigma(D^{-1}AX) = \sigma\left(\begin{bmatrix} -1/2 & -1/4 \\ -1/3 & 5/6 \\ -1/2 & 7/4 \\ 2/3 & 1/3 \end{bmatrix}\right)$$

$$\sigma(D^{-1}AX) = \begin{bmatrix} 0 & 0 \\ 0 & 5/6 \\ 0 & 7/4 \\ 2/3 & 1/3 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0.8333 \\ 0 & 1.75 \\ 0.6666 & 0.3333 \end{bmatrix}$$

Problem 2 – Markov Decision Process.

(a) Action A expected reward $= 0.25 \times 6 + 0.75 \times 2$
$= 3$.

Action B expected reward $= 0.5 \times 3 + 0.5 \times 1$
$= 2$

Reward for action A is greater than action B

Therefore, Action A maximes the expected reward on the following turn starting from state 3.

(b) Discount factor $(\gamma) = 0.8$

Both actions A & B are infinite tracks.

As calculated in (a), Reward for action A from state 3 to state 1 or state 2 is +3 & from state 1 or state 2 to state 3 is 0

Expected discounted reward for action A is,

$$R(A) = \sum_{i=0}^{\infty} (3\gamma^{2i+1} + 0 \times \gamma^{2i})$$

$$= 3(\gamma + \gamma^3 + \gamma^5 + \cdots) + 0$$

$$= 3\gamma(1 + \gamma^2 + \gamma^4 + \cdots)$$

$$= \frac{3\gamma}{1-\gamma^2} = 6.67$$

For action B, expected discounted reward is,

$$R(B) = \sum_{i=1}^{\infty} 0.5(3 \times \gamma^i + 1 \times \gamma^i)$$

$$= 0.5 \times 3(\gamma^1 + \gamma^2 + \cdots) + 0.5(\gamma^1 + \gamma^2 + \cdots)$$

$$= 1.5 \times \frac{\gamma}{1-\gamma} + \frac{0.5\gamma}{1-\gamma} = \frac{2 \times 0.8}{1-0.8}$$

$$= 8$$

$R(B) > R(A)$

(c) r value for $R(A)$ to be greater than $R(B)$ can be achieved as follows,

We want $R(A) > R(B)$,

$$\sum_{i=0}^{\infty} (3r^{2i+1} + 0r^{2i}) > \sum_{i=1}^{\infty} 0.5(3r^i + 1r^i)$$

$$\frac{3}{1-r^2} > \frac{2}{1-r}$$

$$3(1-r) > 2(1-r^2)$$

$$3-3r > 2-2r^2$$

$$2r^2-3r+1 > 0$$

$$(r-1)(2r-1) > 0$$

$$r>1 \quad r<1/2$$

The solution is $\boxed{r < 0.5}$

Problem 3 : Q: Learning

Initially all Q states are initiated to zero (time t = 1)
At every time step we are updating Q value of one state.

After step ①
(a) Since $Q(S_1,A) = Q(S_1,B) = 0$ in the first step. So tie is broken by selecting action A. Action B is not taken. So $Q(S_1,B)$ is not updated

$\therefore \boxed{Q(S_1,B) = 0}$

$$Q(S_1,A) = Q(S_1,A) + \alpha\left(r_1 + \gamma \max_a Q(S_2,a) - Q(S_1,A)\right)$$
$$Q(S_1,A) = 0 + 1\cdot(1 + \gamma \times 0 - 0)$$
$$\boxed{Q(S_1,A) = 1}$$

(b) Since we are updating one state at every step & agent is moving towards right from action A, $Q(S_1,A)$, $Q(S_1,B)$ doesn't get updated until we reach N steps

After 5 steps
$\boxed{Q(S_1,A) = 1}$ $\boxed{Q(S_1,B) = 0}$

(c) After N steps

$$Q(S_N, r) = Q(S_N, r) + \alpha\,(r_N + \gamma \max_{a_1} Q(S_1, a_1) - Q(S_N, r))$$

$Q(S_1, a_1)$ has two actions, action A & B. $Q(S_1, A)$ & $Q(S_1 B)$. The reward of action A is greater than B.

At step N+1

$\therefore\ Q(S_1, A) = 1 + 1(1 + 0.5 \times 1 - 1)$

$= 1.5$

$Q(S_1, B) = 0$.

Step N+2

$Q(S_2, r) = Q(S_2, r) + \alpha\,(r_1 + \gamma \max_{r} Q(S_3, r) - Q(S_2, r))$

$= 1 + 1(1 + 0.5 \times 1 - 1)$

$= 1.5$

⋮

Step N+5

$Q(S_1 A) = 1.5$, $Q(S_1, B) = 0$

(d) (a) At step 2N+1

$$Q(S_1 A) = Q(S_1 A) + \alpha(r_1 + \gamma \max_a Q(S_2, r) - Q(S_1, A))$$

$$= 1.5 + 1(1 + 0.5 \times 1.5 - 1.5)$$

$$= 1.5 + 0.25$$

$$= 1.75$$

So from two update of $Q(S_1, A)$ we can see that

$$x' = x + \alpha(1 + \gamma x - x) = Q^{new}(S_1 A)$$

$$x' = x + (1 + \gamma x - x) \quad \dots \alpha = 1$$

$$x' = 1 + \gamma x. \quad \dots x = Q(S_1, A) = Q(S_2, r)$$

$$x'' = 1 + \gamma x' = 1 + \gamma(1 + \gamma x)$$

$$x'' = 1 + \gamma + \gamma^2 x$$

$$x''' = 1 + \gamma + \gamma^2 + \gamma^3 x$$

In the limiting case

$$x = 1 + \gamma + \gamma^2 + \gamma^3 + \dots$$

$$= \frac{1}{1-\gamma} = \frac{1}{1-0.5} = 2$$

$$\underbrace{Q(S_1 A) = 2, \quad Q(S_1, B) = 0, \quad Q(S_2, \text{right}) = 2}_{\text{Convergence value}}$$

Problem 4

Deep Reinforcement Learning

a)

1) In Reinforcement Learning algorithm must follow some policy in order to decide which actions to perform at each state. Still, the learning procedure of the algorithm doesn't have to take into account that policy while learning. Algorithms which concern about the policy which yielded past state-action decisions are referred to as on-policy algorithms, while those ignoring it are known as off-policy.
2) Off-policy algorithms are Q-learning (last time) R-learning (a variant of Q-learning) and on-policy algorithms are SARSA, TD(λ) and Actor-critic methods

b) Experience replay reduces correlations in the sequence of observations, which otherwise might drive the network into a local minimum while training the DQN model.

c) The difference between Q-learning and Policy Gradient methods can be explained as below:

1. Objective Function
   i. In Q-Learning we learn a Q-function that satisfies the Bellman (Optimality) Equation. This is most often achieved by minimizing the Mean Squared Bellman Error (MSBE) as the loss function. The Q-function is then used to obtain a policy (e.g. by greedily selecting the action with maximum value).
   ii. Policy Gradient methods directly try to maximize the expected return by taking small steps in the direction of the policy gradient. The policy gradient is the derivative of the expected return w.r.t. the policy parameters.
2. On- vs. Off-Policy
   i. The Policy Gradient is derived as an expectation over trajectories (s1,a1,r1,s2,a2,...,rns1,a1,r1,s2,a2,...,rn), which is estimated by a sample mean. To get an unbiased estimate of the gradient, the trajectories have to be sampled from the current policy. Thus, policy gradient methods are on-policy methods.
   ii. Q-Learning only makes sure to satisfy the Bellman-Equation. This equation has to hold true for all transitions. Therefore, Q-learning can also use experiences collected from previous policies and is off-policy.
3. Stability and Sample Efficiency
   i. Directly optimizing the return and thus the actual performance on a given task, Policy Gradient methods tend to more stably converge to a good behavior. Indeed being on-policy, makes them very sample inefficient. Q-learning find a function that is guaranteed to satisfy the Bellman-Equation, but this does not guarantee to result in near-optimal behavior. Several tricks are used to improve convergence and in this case, Q-learning is more sample efficient.

d)

   i. The "Critic" estimates the value function, measures how good the action taken is (value-based). It is useful to assess how things go after each action, the critic computes the new state to determine if there has been any improvement or not.

ii. The critic approximates and updates the value function using samples. The value function is then used to update the actor's policy parameters in the direction of performance improvement. These methods usually preserve the desirable convergence properties of policy gradient methods. Thus, the critic helps to learn optimal policy