21' APRIL

03 DAY 093-272 | Wk 14
SATURDAY

MARCH '21

M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

APRIL '21

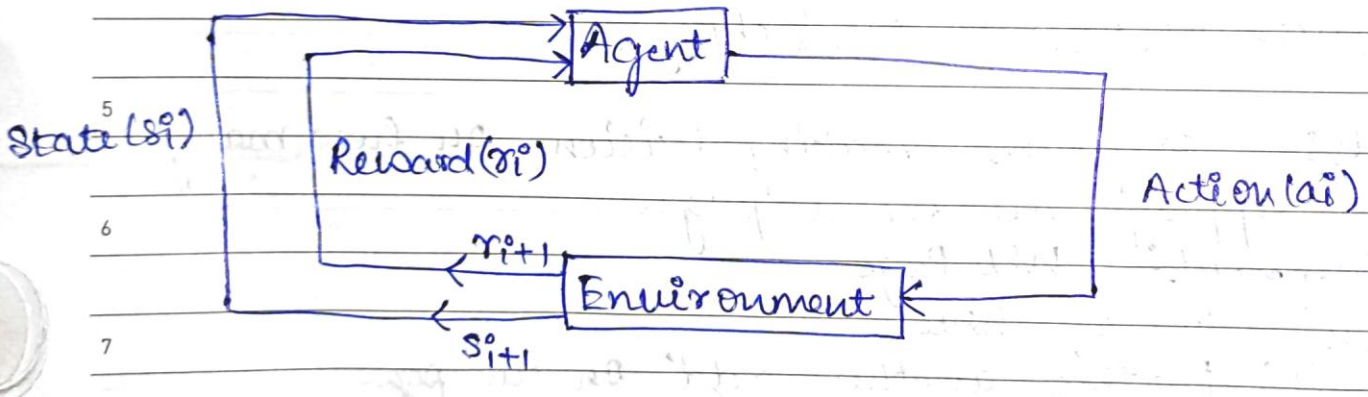
M	T	W	T	F	S	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

Reinforcement learning

A subfield of ML that teaches agent how to choose an action from its action space within particular environment in order to maximize reward over time.

4 Elements:

1) Agent : The program you train, with the aim of doing job.
2) Environment: The world real or virtual in which the agent perform actions.
3) Action: A move made by agent which causes a status change in environment.
4) Rewards: The evaluation of action, positive or negative.



04 SUNDAY

Characteristics of RL

1) No supervisor (Reward signal)
2) Sequential Decision Making
3) Time plays crucial Role
4) Feed back is delayed
5) Agent's actions determine the subsequent data it receives.

Basis	Positive Reinforcement	Negative Reinforcement
1) Def.	Process of introducing a stimulus, to increase the probability of occurence of pattern or behaviour.	Process of removal of unfavourable stimulus for the pupose of encouraging good behaviour.
2) Stimuli	Added	Removed.
3) Consequences	Pleasant	Unpleasant
4) Reinforcer acts as	Reward	Penalty.
5) Result in	Strengthing and Maintaining Responses.	Avoiding and escaping responses.
6) ~~Drawback~~ DSCRT	Maximize performance	Maintain minimum performance.

Applications of Reinforcement learning

1) Robotics for Industrial Automation
2) Business strategy planning
3) Machine learning and Data processing.
4) Aircraft control and Robot Motion control.
5) Manufacturing Optimization problems.
6) Sequential Scheduling Problems

Approaches:

(1) **Value Based:** In this, we try to maximize a value function V(s). In this method agent is expecting a long term return of current states under policy π.

(2) **Policy Based:** In this we try to come up with such policy that the action performed in every state helps you to gain Max reward in future.

(i) Deterministic:
For every state same action is produced by policy π.

(ii) Stochastic:
Every action has a certain probability
$\pi(a|s) = P[A_t = a | S_t = s]$

SUNDAY 28

(3) **Model Based:** In this we need to create a virtual model for each environment. The agent learns to perform in that specific environment.

Terminalogies:

1) **Agent:** It is an assumed entity which performs actions in an envt. to gain some reward.

2) **Environment (e):** A scenario that an agent has to face.

3) **Reward (R):** An immediate return given to an agent when they perform specific task.

4) **State (s):** Current situation returned by environment.

5) **Policy (π):** strategy applied by agent to decide the next action based on current state. $\pi(s) \rightarrow a$

6) **Value (V):** It is expected long-term return with discount as compared to short term reward.

7) **Value function:** It specifies the value of a state that is a total amount of reward. It is an agent which should be expected beginning from the state.

8) **Q-Value / Action Value:** Q is quite similar to value. It takes additional parameters as a current action.

learning → MDP
Model → Ø

Markov Decision Process

Markov property: The future is independent of the past given the present.
A state s_t is Markov iff

$$P[S_{t+1} | S_t] = P[S_{t+1} | S_1, \ldots S_t]$$

MDP consist of

1) Set of state S.
2) Set of Action, A
3) Transition function $T(S, a, S')$
   $T(S, a, S') \sim P(S' | S, a)$
4) Reward, $R(S)$, $R(S, a)$, $R(S, a, S')$

5) **Policy:** $\pi(S) \rightarrow a$
   π^*

↳ Transition Model gives an action's effect in a state.
↳ Policy is a solution to MDP. A policy is the mapping from S to a. It indicates action a to be taken while in state 'S'.

MDP is Mathematical framework to describe an environment in Reinforcement Learning.

Q-learning

A value based Method of supplying information to inform which action an agent should take
[value based algorithm update value fun using Bellman eqn]

Q-function: $Q(s,a)$

↳ Its value is the maximum discounted cumulative reward achieved by starting from state s, applying action a as first action.

↳ Q is the reward received immediately upon executing action a from state s, plus the value of following optimal policy thereafter.

$$Q(s,a) \equiv r(s,a) + \gamma V^*(\delta(s,a))$$

$$\pi^*(s) = \arg\max_a Q(s,a)$$

$\pi^*(s)$, optimal action in state s.

$r(s,a)$, reward

V^* value of immediate successor state

γ discount.

Q-learning Algorithm:

1) For each s,a initialize table entry $\hat{Q}(s,a)$ to zero
2) Observe current state
3) Do forever:
   - ↳ Select an action and execute it
   - ↳ Receive Immediate reward r
   - ↳ Observe new state s'
   - ↳ Update table entry for $\hat{Q}(s,a)$ as:

$$\hat{Q}(s,a) \leftarrow r + \gamma \max_{a} \hat{Q}(s',a')$$

   - ↳ $s \leftarrow s'$

1) for each (s,a) initialize table entry $\hat{Q}(s,a)$ to zero
2) Observe current state
3) Do forever:
   - (i) Select an action and execute it
   - (ii) Receive Immediate reward r
   - (iii) Observe new state s'
   - (iv) Update table entry for $\hat{Q}(s,a)$ as:

$$\hat{Q}(s,a) \leftarrow r + \gamma \max_{a} \hat{Q}(s',a')$$

   - (v) $s \leftarrow s'$

Examples

1) controlling a walking Robot:

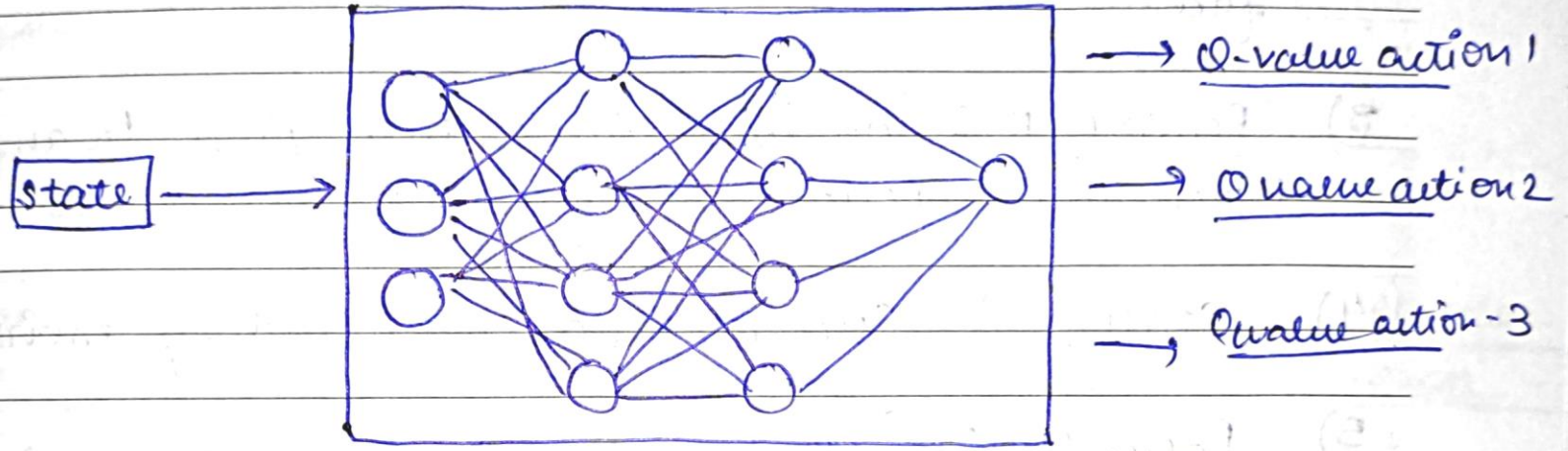
- Agent: Program controlling a walking Robot.
- Environment: Real World
- Action: One out of 4 moves.
  (1) Forward
  (2) Backward
  (3) left
  (4) Right
- Reward: Positive when it approaches the target destination direction; negative when it wastes time and goes in wrong direction.

2) Placement of Ads on Webpage:

- Agents: Program making decision on how many ads are appropriate for a page.
- Environment: Web Page
- Action: one of three
  (I) Putting another add on the pg.
  (II) Drop add from the pg.
  (III) Neither add nor remove
- Reward: Positive when revenue increases
  Negative when Revenue drops.

Deep Q-learning

In deep learning we use Neural Network to approximate the Q-value function.
The state is given as I/P and Q values of all possible actions is generated as the O/P



Step:1 Store all the past experience in memory.
Step:2 Next action is determined by Max O/P of Q-network
Step:3 Loss function = Mean sq. error of predicted Q-value and target Q-value - Q*

Bellman Equation:

$$Q(S_t A_t) \leftarrow Q(S_t A_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right]$$