Date
Page 1.01

LECTURE 1:-

* OUTLINE:

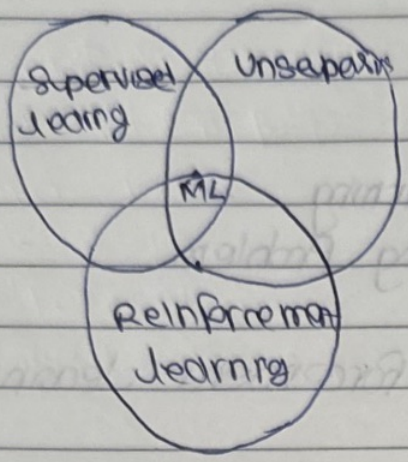
1. Admin
2. About Reinforcement learning
3. Reinforcement learning Problem
4. Inside an RL agent
5. Problems within Reinforcement learning

Book →

1. An introduction to Reinforcement learning, Sutton and Barto, 1998
2. Algorithms for Reinforcement learning, Szepesvari

* ~~ABOU REINFORCEMENT LEARNING~~:-

RL sits at the intersection of many different fields of science



Supervised learning
Unsupervised learning
Reinforcement learning
} Machine learning

Q.] What makes Reinforcement learning different from other machine learning Paradigms?

→ Answer ↓

1.] There is no superviser, only a reward signal
2.] Feedback is delayed, not instantaneous
3.] Time really matters (sequential, non i.i.d data)
4.] Agent's Actions affect the subsequent data it receives

* EXAMPLES OF REINFORCEMENT LEARNING:-

1. Fly stunt manoeuvres in a helicopter
2. Defeat the world champion at backgammon
3. Manage an investment portfolio
4. Control a power station
5. Make a humanoid robot walk
6. Play many different Atari games better than humans

* REINFORCEMENT LEARNING PROBLEM:-

→ 1. A reward R_t is a scalar feedback signal
→ 2. Indicates how well agent is doing at step t
→ The agents job is to maximise cumulative reward
→

Reinforcement learning is based on the reward hypothesis

Definition (Reward Hypothesis)
⇒ All goals can be described by the maximisation of epected cumulative reward

Date
Page

* Examples of Rewards:-

1] Fly stunt manoeuvres in a helicopter
+ve reward for following desired trajectory
-ve reward for crashing

2] Defeat the world champion at Backgammon
+ve/-ve reward for winning/losing a game

3] Manage an investment portfolio
+ve reward for each $ in bank

4] Control a power station
+ve reward for producing power
-ve reward for exceeding safety thresholds

5] Make a humanoid robot walk
+ve reward for forward motion
-ve reward for falling over

6] Play many atari games better than humans
+ve/-ve reward for increasing/decreasing score

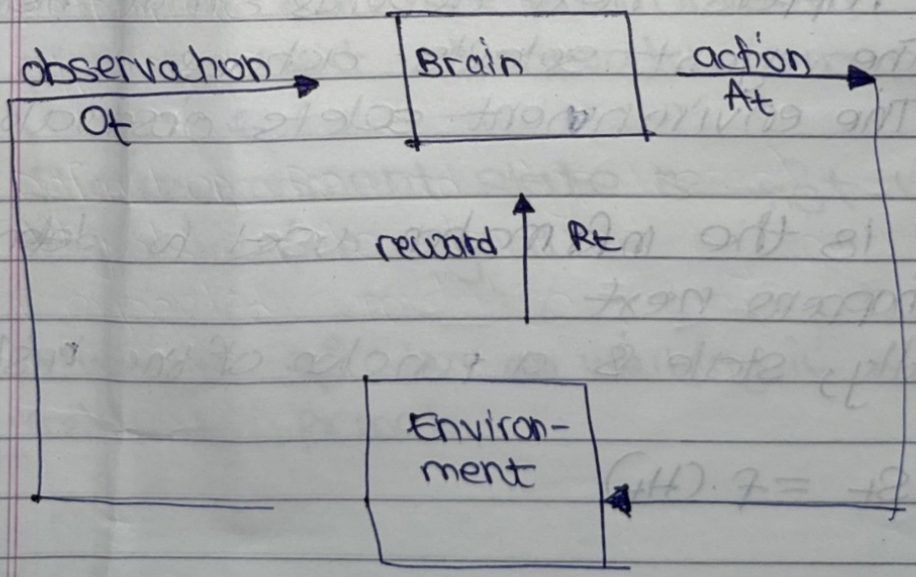
Date .....
Page .....

* SEQUENTIAL DECISION MAKING :-

→ Goal :- select actions to maximise total future reward
→ Actions may have long term consequences
→ Reward maybe delayed
→ It may be better to sacrifice immediate reward to gain more long-term reward

Examples :-
→ A financial investment (may take months to mature)
→ Refuelling a helicopter (might prevent a crash in several hours)
→ Blocking opponent moves (might help winning chances many moves from now)

* AGENT AND ENVIRONMENT :-



⇒ At each step t the agent:-
- Executes action A_t
- Receives observation O_t
- Receives scalar reward R_t

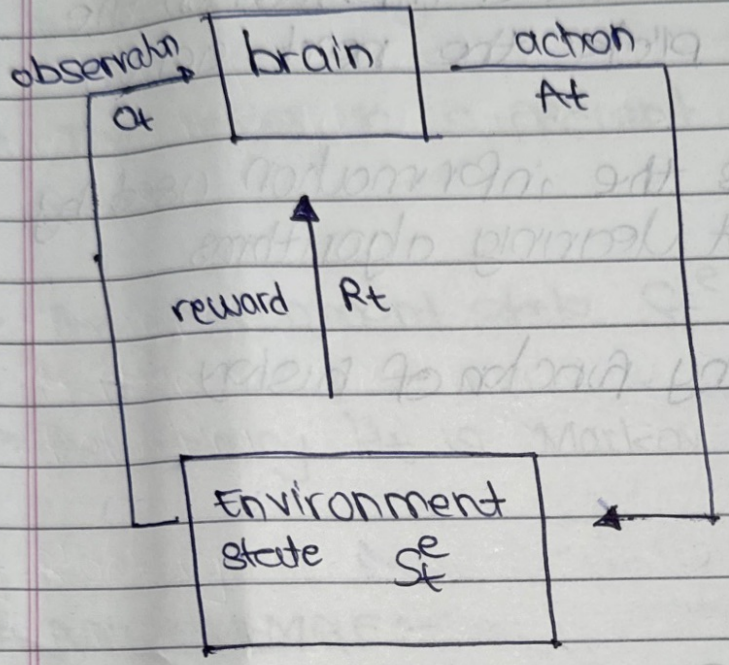
⇒ The environment:-
- receives action A_t
- Emits observation O_t
- Emits scalar reward R_t

* HISTORY AND STATE:-

- The history is the sequence of observations, actions, rewards
  $H_t = A_1, O_1, R_1, \ldots, A_t, O_t, R_t$
- that is all observable variables up to time t
- that is the sensorimotor stream of a robot or embodied agent
- What happens next depends on the history:-
  - The agent selects actions
  - The environment selects observations/rewards
- State is the information used to determine what happens next
- Formally, state is a function of the history

$$S_t = f(H_t)$$

* ENVIRONMENT STATE:



→ The environment state S^e_t is the environment's private representation

→ whatever data the environment uses to pick the next observation/reward

→ The environment state is not usually visible to the agent

→ Even if S^e_t is visible, it may contain irrelevant information

* AGENT STATE:-

→ The agent state S_t^a is the agents internal representation

→ that is used whatever information the agent uses to pick the next action

→ that is it is the information used by reinforcement learning algorithms

→ It can be any function of history
$S_t^a = f(H_t)$

* INFORMATION STATE:-

→ An information state (a.k.a Markov state) contains all useful information from the history

#Definition

A state S_t is Markov is and only if

$$P[S_{t+1} \mid S_t] = P[S_{t+1} \mid S_1, S_2 \dots S_t]$$

↑ probability

→ "The future is independent of the past given the present"

$H_{1:t} \rightarrow S_t \rightarrow H_{t+1:\infty}$

→ Once the state is known, the history may be thrown away

→ The state is sufficient statistic of the future

→ The environment state S_t^e is Markov

→ The history H_t is Markov

RAT EXAMPLE:-

light light
~~bell~~ ~~bell~~ lever bell electricity
bell light lever lever cheese
lever light. lever bell ?

→ what if agent state = last 3 items in sequence
→ what if agent state = counts for lights, bells, levers
→ what if agent state = complete sequence?

Date
Page

* FULLY OBSERVABLE ENVIRONMENTS:-

⇒ Full observability:-
agent directly observes environment state
$o_t = s_t^a = s_t^e$

→ Agent state = environment state = information state
→ Formally, this is a Markov decision process (MDP)

* PARTIALLY OBSERVABLE ENVIRONMENTS:-

⇒ Partial observability:-
agent indirectly observes environment

→ A robot with camera vision isn't told its absolute location
→ A trading agent only observes current prices
→ A poker playing agent only observes public cards

⟹ Now agent state ≠ environment state

⟹ Formally, this is a partially observable Markov decision process (POMDP)

⇒ Agent must construct its own state representation s_t^a, eg

→ Complete history: $s_t^a = H_t$
→ Beliefs of environment state $s_t^a = (p[s_t^{e=s^1}], p[s_t^{e=s^2}], \ldots)$
→ Recurrent neural network: $s_t^a = \sigma(s_{t-1}^a W_s + o_t W_o)$

Recurrent nueral network - $s_t^a = \sigma(s_{t-1}^a W_s + a_t W ...$

* INSIDE AN RL AGENT:-

→ An RL agent may include one or more of these components:-

1] Policy:- agent's behaviour function
2] Value function:- how good is each state and/or action
3] Model:- agent's representation of the environment

1.] Policy:-

→ It is the agents behaviour
→ It is a map from state to action, eg
→ Deterministic policy : $a = \pi(s)$
→ Stochastic policy : $\pi(a|s) = P[A = a | S = s]$

2.] Value function:-

→ Value function is a prediction of future reward
→ Used to evaluate the goodness / badness of states
→ And therefore to select between actions, eg

$$V_\pi(s) = E_\pi [R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \ldots | S_t = s]$$

3] * model :-

→ Model predicts what the environment will do next

→ Transitions: P predicts the next state (i.e. dynamics)

→ Rewards: R predicts the next (immediate) reward

$$P^{a}_{ss'} = P[S' = s' \mid S = s, A = a]$$

$$R^{a}_{s} = E[R \mid S = s, A = a]$$

→ Maze example ✓

* CATEGORIZED RL AGENTS :-

→ Value Based
- No Policy (Implicit)
- Value function

→ Policy Based
- Policy
- No value function

→ Actor Critic
- Policy
- Value function

→ Model free
- Policy and /or Value function
- No model

→ Model Based
- Policy and /or Value function
- Model

⇒ RL agent Taxanomy (diag)

Date
Page

* PROBLEMS WITHIN R.L

Two fundamental problems in sequential decision making.

→ Reinforcement learning:-
- The environment is initially unknown
- The agent interacts with the environment
- The agent improves its policy

→ Planning:-
- The model of the environment is known
- The agent performs computations with its model (without any external interaction)
- The agent improves its policy
- aka deliberation, reasoning, introspection, pondering, thought, search

→ Atari Example
LEARNING
- Rules of game are unknown
- learn directly from interactive game play
- Pick actions on joystick, see pixels and scores

PLANNING:-
- Rules of the game are known
- Can query emulator
  - perfect model inside agent's brain
- If I take action a from state s
  - what would the next state be?
  - what would the score be?
- Plan ahead to find optimal

→ Reinforcement learning is like trial and error learning
→ The agent should discover a good policy
→ from its experiences of the environment
→ without losing too much reward along the way

* Exploration & Exploitation

→ Exploration finds more information about the environment
→ Exploitation exploits known information to maximise the reward
→ It is important to explore & exploit

Examples:-

1] Restraunt Selection:-
Exploitation: go to fav restraunt
Exploration: try new restraunt

2] online banner Advertisements:-
Exploitation Show the most successful advert
Exploration Show a different advert

3] Oil drilling
Exploitation: Drill at the best known location
Exploration Drill at new location

4] Game playing:-
Exploit:- play the move you believe is best
Explore:- Play a new move

Date
Page

* Prediction & Control

- Prediction: evaluate the future
  - Given a policy
- Control: optimise the future
  - Find the best policy

LECTURE:-2 MARKOV DECISION PROCESS

1] Markov Processes
2] Markov Reward Processes
3] Markov Decision Processes
4] Extensions to MDPs

1] MARKOV PROCESS

⇒ Introduction to MDPs

- Markov decision processes formally describe an environment for Reinforcement learning
- where the environment is fully observable
- that is the current state completely characterises the process
- Almost all R.L problems can be formalised as MDPs eg
  - Optimal control primarily deals with continuous MDPs
  - Partially observable problems can be converted into MDPs
  - Bandits are MDPs with one state.

MARKOV PROPERTY:-

"The future is independent of the past given the present"

Definition

A state S_t is Markov if & only if

$$P[S_{t+1} | S_t] = P[S_{t+1} | S_1, S_2 \ldots S_t]$$

- The state captures all relevant information from the history
- Once the state is known, the history may be thrown away
- That is state is a sufficient statistic of future

* Why do we have Discount factor?
→ To indicate that our model is not perfect

Value Function

eg if dropped in class '2' from there till pass how many rewards would you get

$$V(S) = E[G_t \mid S_t = S]$$

Expected return

eg 2

~~1.3 = (0.4)(0.8) + (1.5)(0.8) - 2~~

~~= 0.32 + 0.37 + 1.2 - 2~~

Date .....
Page .....

$$V_\pi = R^\pi + \gamma P^\pi V_\pi$$

avg reward function

Avg transition dynamics

Date
Page

LECTURE 3:- Planning By Dynamic programming

policy evaluation → if someone gives you a policy how good is that

policy iteration → using policy evaluation,

π' → new policy

⇒ LECTURE! - 4 MODEL FREE PREDICTION

2.] Monte-Carlo → methods that go to the end of trajectory & estimate the value just by looking at sample

3.] Temporal-diff
look 1 step ahead & estimate return

MC / TD better!

TD is more efficient?
because markov property

⇒ LECTURE 5:- Model free control

When you throw your robot in an unknown environment

TD learning → or off policy