Lecture I : Introduction to Reinforcement Learning

29/07/19

RL is at intersection of various fields
- ↳ study of the science of decision making
- ↳ optimal way to make decisions
- ↳ optimal control in engineering
- ↳ neuroscience : decision making in human brain
  - ↳ dopamine system (reward system)
- ↳ psychology : classical, optimal conditioning
- ↳ math : operations research
- ↳ GT : utility theory, bounded rationality (economics)

RL v/s Supervised learning :

- no supervisor, just trial & error
- feedback is not instantaneous (delayed feedback)
- sequential decision making matters in RL
  - ↳ not an IID dataset (independant and identically dist.)
- agent takes actions and influences the environment ie. it influences the data it sees

Example RL problems :

- flying stunt manoeuvres in a helicopter
- play board games like Backgammon
- manage an investment portfolio
- control a power station

- make a humanoid robot walk
- single program to play a suit of games, eg :- atari

"games are microcosms of real things happening in the real world"
~ D. Silver

The Reinforcement Learning Problem :

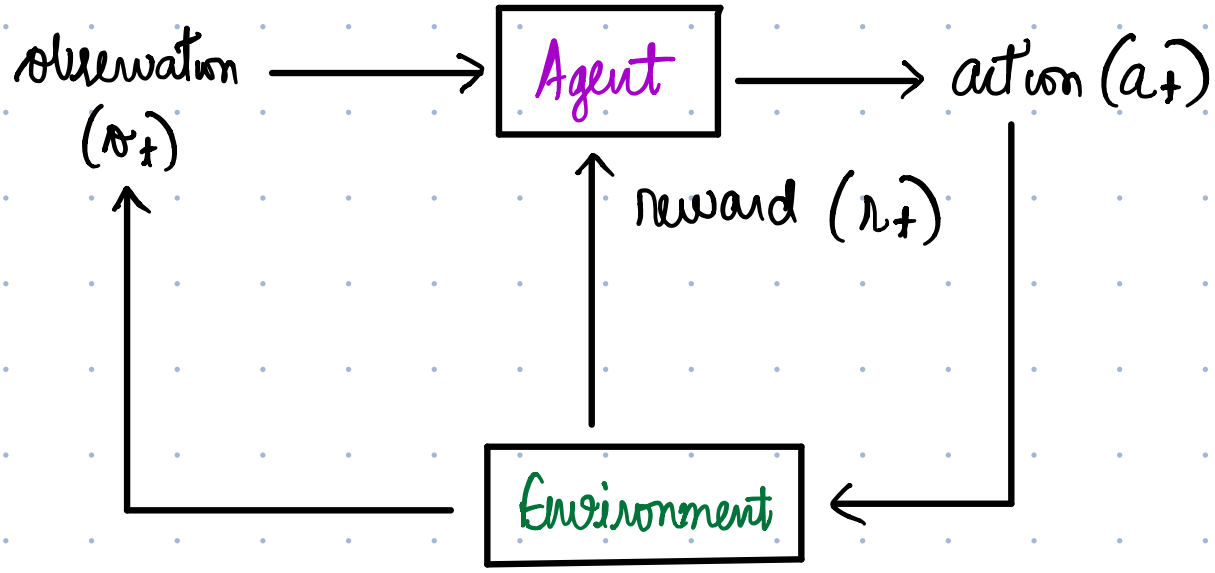
Rewards :

- R_t, scalar feedback signal
- indicates how well the agent is doing at step "t"
- goal is to maximize cumulative reward
- reward hypothesis : all goals can be described by the maximisation of expected cumulative reward
- to minimize time taken, a negative reward per time step till end of episode is used

Sequential Decision Making :

Goal :

- select actions to maximise total future reward
- actions can have long term consequences
- rewards may be delayed
- sacrifing a present reward may lead to a large future reward — non greedy action choices



History and State:

- history (H_t)
  - ↳ $A_1 O_1 R_1 A_2 O_2 R_2 \ldots A_t O_t R_t$
- goal is to build a mapping from H_t to plausible actions
- H_t is not very useful as it is enormous
- We use state instead to determine what happens next
  - ↳ $S_t = f(H_t)$

Environment State:

- information used within the env. to determine what happens next
- S_t^e is the environment's private representation
- it is usually not visible to the agent
- more like a formalism that helps us understand what an environment is
- it may contain irrelevant information to the task

Agent State :

- set of numbers inside the agent
- S_t^a is the agent's internal representation
- This is used by the RL algorithm
- It can be any function of the history : $S_t = f(H_t)$

Information State : (aka Markov State)

- It contains all useful information from history
- A state S_t is Markov iff :

$$\Pr[S_{t+1} | S_t] = \Pr[S_{t+1} | S_t, \dots S_1]$$

- "The future is independant of the past given the present"
- The state is a sufficient statistic of the future (and future rewards)
- The environment state (S_t^a) is Markov
- The history (H_t) is also Markov

Fully Observable Environment :

- Agent directly observes the environment state

$$O_t = S_t^a = S_t^e$$

- This is a MDP

Partially Observable Environment :

- agent indirectly observes environment
- eg : poker player, trading agent
- POMDP (partially observable MDP)
- Agent must construct its own S_t^a
  - Complete history : $S_t^a = H_t$
  - Beliefs of env state : $S_t^a = (Pr[S_t^e = s^1], \ldots, Pr[S_t^e = s^n])$
  - RNN : $S_t^a = \sigma(S_{t-1}^a W_s + O_t W_o)$

Guide an RL Agent :

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

Policy :

- mapping from state to action
- deterministic policy : $a = \pi(s)$
- stochastic policy : $\pi(a|s) = Pr[A = a | S = s]$

Value function :

- prediction of expected future reward
- formally written as :

$$\hookrightarrow v_\pi(s) = E_\pi[R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \dots \mid S_t = s]$$

- helps evaluate the goodness badness of states

Model :

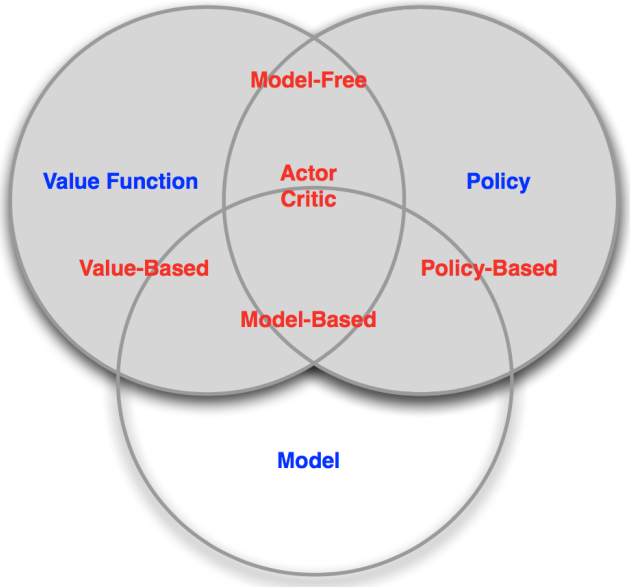
- predicts what the environment will do next
- Transitions : predicts next state (dynamics) [P]
- Rewards : predicts next (immediate) reward [R]

$$P^a_{ss'} = Pr[S' = s' \mid S = s, A = a]$$
$$R^a_s = E[R \mid S = s, A = a]$$

Categorizing 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 Taxonomy

Learning and Planning:

- RL:
  - env initially unknown
  - interacts with env with trial and error learning
  - agent improves its policy
- Planning: (lookahead search, tree search etc)
  - model of env is known
  - performs computation with its model
  - agent improves its policy

Exploration and Exploitation:

- RL is trial and error
- agent should discover a good policy
    - from interaction w. the env.
    - w/o losing too much reward along the way
- exploration
    - sacrifice reward to know more about env.
- exploitation
    - exploit learned information

Prediction and Control:

- Prediction: evaluate the future
    - given a policy
- Control: optimise the future
    - find the best policy

RL: solve the prediction problem to solve the control problem

Course Outline:

Part I: Elementary RL:

- Intro to RL
- MDP
- Planning by DP
- Model free prediction
- Model free control

Part II: RL in practice

- Value function approximation
- Policy gradient methods
- Integrating learning & planning
- Exploration and exploitation
- Case study – RL in games