Reinforcement Learning

Introduction

- Uses a mix of supervised and unsupervised learning
- Differentiator between other ML paradigms:
-
    - there is no supervisor, only a reward signal (good or bad signals, not correct or wrong signal)
    - delayed feedback rather than instantaneous
    - time is really important(non i.i.d.)
    - agent affects the environment and subsequently the data that it receives

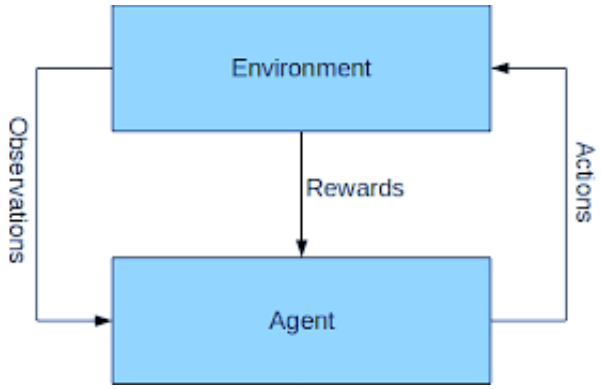
Reinforcement Learning Problem

- R^t : a scalar feedback signal. An indication of how well the agent is doing at step t.
- Agent's goal is to maximize the the cumulative reward
- RL is based on the following premise: **All goals can be described by the maximization of expected cumulative reward**

Sequential Decision Making

- Goal: **Select actions that maximize total future reward**
- Actions may have long term consequences
- Reward may be delayed
- Sacrifice immediate reward to gain a greater long-term reward
- 2 problems in sequential decision making:
    - RL: env is initially unknown, agent interacts and builds its policy to maximize reward
    - Planning: env model is fully known, agent performs computations with its model and then builds the policy
- Exploitation/Exploration tradeoff

Agent and Environment



- At each step t, the agent:
- - Executes action A_t
  - Receives observation O_t
  - Receives scalar reward R_t
- At each step 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 actions, observations and rewards: $H_t = A_1, O_1, R_1, \cdots, A_t, O_t, R_t$
- These are all the observable variables up to time t.
- What happens at time t+1 depends on the history H_t. Specifically:
  - From the agent's perspective: It picks an action A_{t+1} depending on history H_t(We are trying to build a mapping from the history H_t to the next action A_{t+1})
  - From the environment's perspective: It picks a observation O_{t+1} and reward R_{t+1} depending on the history H_t
- But the history is too vast and has too much information
- **State** is the more concise version of the history which we can use to determine what happens next
- Formally, state is the function of history: $S_t = f(H_t)$
- - **Environment State:** S_t^e is the environment's private information. It is the inner mechanics of the environment that is not usually visible and/or useful to the agent.
  - **Agent State:** S_t^a is the agent's internal representation. This is used by the agent to pick its next action as a function of the history: $S_t^a = f(H_t)$. It is a part of the RL algorithm
  - **Information State/Markov State:** contains all useful information about the history. A state is Markov iff $P[S_{t+1}|S_t] = P[S_{t+1}|S_1, \cdots, S_t]$, i.e. the future is independent of the past given the present. $H_{1:t} \rightarrow S_t \rightarrow H_{t+1:\infty}$ (from the full history, we only need the current state representation S_t which will give us enough information about all future states $H_{t+1:\infty}$) See MDP
- Full Observability

    - Agent directly observes environment state. $O_t = S_t^a = S_t^e$
    - This is a Markov Decision Process(MDP)
- Partial Observability
    - Agent indirectly/incompletely observes the environment
    - This is a Partially Observable Markov Decision Process(POMDP)
    - Agent must construct its own state representation:
        - Complete history $S_t^a = H_t$
        - Belief of Env State $S_t^a = (P(S_t^a = s_1), \cdots, P(S_t^a = s_n))$
        - Recurrent Neural Network $S_t^a = \sigma(S_{t-1}^a W_s + O_t W_0)$

RL Agent

Components of the Agent:

- **Policy:** agent's behavior function
- **Value Function:** how good each state/action is
- **Model:** agent's representation of the environment

- **Behavior of the agent:** Map from state to action
- **Deterministic policy:** $a = \pi(s)$
- $\pi(a|s) = P[A = a|S = s]$

Value Function:

- prediction of future reward(expected total reward)
- used by the agent to evaluate states and choose between them
- $V_\pi(s) = E_\pi(R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \ldots | S_t = s)$

- The model predicts what the environment will do next
- **Transition Model** P**:** predicts the next state. $P_{ss'}^a = P[S' = s'|S = s, A = a]$
- **Reward Model** R**:** predicts the next immediate reward. $R_s^a = E[R|S = s, A = a]$

Types of Agents:

- **(Only)Value based agent:** Maximize value at each step
- **(Only)Policy based agent:** Maximize policy at each step
- **Actor Critic (Both policy and value):** Combination of value and policy
- **Model Free:** Don't try to explicitly understand how the environment is working. Rather just use the policy and/or value
- **Model Based**

Miscellaneous

A well defined reward structure should specify the desired outcomes and not specific behaviors

If we reward a specific behavior in the hope that it will result in the desired outcome, then we are making an assumption that it is the optimal behavior

Rather, we must allow the agent to learn that optimal behavior

This results in another difficult but manageable problem of reward sparsity.

Building an optimal reward structure which has well defined subgoals is quite important to get over this