ReinforcementLearning Notes

Avinash Reddy

January 2023

1 Introduction

Reinforcement Learning is also known as

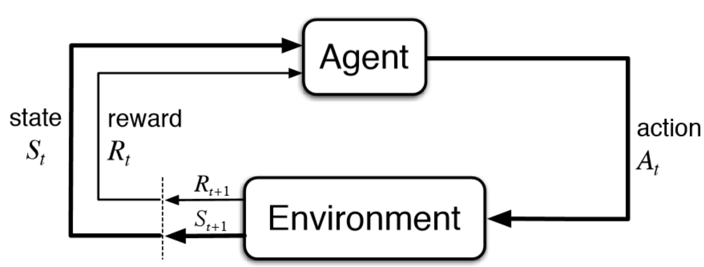
- optimal control
- Approximate Dynamic programming
- Neuro-Dynamic Programming

Reinforcement Learning is an area of machine learning inspired by the behavioural psychology concerned with how software agents ought to take actions in an environment so as to maximise some notion of cumulative reward.
Animal psychology

- negative rewards - pain and hunger
- positive reinforcements - pleasure and food
- reinforcements used to train animals

Applying the similar philosophy to software agents

2 Definition



Reinforcement Leaning Application Areas :

- Game Playing
- Operations Research
- Elevator Scheduling
- controls
- spoken dialouge systems
- data center energy consumption
- self managing networks
- autonomous vehicles
- computational finance

3 Markov Decision Process

Definition

- State $s \in S$
- Action $a \in A$
- reward $r \in \mathbb{R}$
- Transition model $Pr(s_{t+1}|s_t, a_t)$
- Reward model $Pr(r|s_{t+1}, s_t, a_t)$
- Discount Factor $\gamma \in [0, 1]$
- Horizon T

Goal is to find a policy $\pi(a|s)$ that maximises the expectation of discounted return.

How RL differs from MDP solutions

- No Transition Model
- No Reward Model

$$J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{T} \gamma^t r_t \right]$$

However, we still solve the MDP problem using RL by interacting with the environment by learning the transition and reward models or directly learning the policy.

Types of RL algorithms

- Model Based- if we try to learn the transition and reward models
- Model Free - here, we don't learn any model dynamics. No transition and reward models. Below are the types of model free RL algorithms
- Value Based- if we try to learn the value function $V(s)$ of the state or value function of state-action pair $Q(s,a)$.
- Policy Based- if we try to learn the policy $\pi(a|s) - \pi(s,a)$ directly.
- Policy Gradient- if we try to learn the policy $\pi(a|s) - \pi(s,a)$ directly using gradient ascent.
- Actor Critic - contains both policy $\pi(a|s) - \pi(s,a)$ and value function $V(s) - Q(s,a)$.

4 Model Free Evaluation

Monte Carlo Evaluation

$$\begin{aligned}
V^{\pi}(s) &= \mathbb{E}_{\pi}\left[\sum_{t=0}^{T} \gamma^t r_t\right] \\
&\approx \frac{1}{n(s)} \sum_{n(s)}^{k=1} \left[\sum_{t=0}^{T} \gamma^t r_t\right] && \text{(sample approximation)} \\
&= \frac{1}{n(s)} \sum_{n(s)}^{k=1} G_k && \text{discounted sum of reward is defined as } G_k
\end{aligned}$$

Temporal Difference Learning

$$\begin{aligned}
V^{\pi}(s) &= \mathbb{E}_{\pi}[r|s] + \gamma \sum_{s_{t+1}} Pr[s_{t+1}|s_t] V^{\pi}(s_{t+1}) \\
&\approx r + \gamma V^{\pi}(s_{t+1}) && \text{(one sample approximation)}
\end{aligned}$$

5 Monte Carlo Evaluation

Monte Carlo Evaluation

$$G_k = \sum_t \gamma^t r_t^{(k)}$$

G_k is a discounted sum of rewards in one episode or trajectory.
Approximate value function

$$
\begin{aligned}
V_n^\pi(s) &\approx \frac{1}{n(s)} \sum_{k=1}^{n(s)} G_k \\
&= \frac{1}{n(s)} \left[G_{n(s)} + \sum_{k=1}^{n(s)-1} G_k \right] \\
&= \frac{1}{n(s)} \left[G_{n(s)} + (n(s)-1) \frac{1}{n(s)-1} \sum_{k=1}^{n(s)-1} G_k \right] \\
&= \frac{1}{n(s)} \left[G_{n(s)} + (n(s)-1) V_{n-1}^\pi(s) \right] \\
&= V_{n-1}^\pi(s) + \frac{1}{n(s)} \left[G_{n(s)} - V_{n-1}^\pi(s) \right] \\
&= V_{n-1}^\pi(s) + \alpha \left[G_{n(s)} - V_{n-1}^\pi(s) \right] && \text{where } \alpha = \tfrac{1}{n(s)}
\end{aligned}
$$

Incremental update step

$$V_n^\pi(s) \leftarrow V_{n-1}^\pi(s) + \alpha_n \left[G_{n(s)} - V_{n-1}^\pi(s) \right]$$

iterate over each sample trajectory and do the incremental update of the value function.

Temporal Difference Learning

Temporal Difference Learning approximate value function

$$V^\pi(s) \approx r + \gamma V^\pi(s_{t+1})$$

Incremental update step

$$V_n^\pi(s) \leftarrow V_{n-1}^\pi(s) + \alpha_n \left[r + V_{n-1}^\pi(s_{t+1}) - V_{n-1}^\pi(s) \right]$$

6 DQN Learning

Algorithm 1 DQN Learning- Gradient Learning

Initialise a Q network with parameters θ
start with state s_t
while True **do**
  take action a_t
  observe next state s_{t+1} and R_{t+1}
  calculate the gradient

$$\theta \leftarrow \theta + \alpha \bigtriangledown_\theta \left[R_{t+1} + \gamma max_a Q_\theta(s_{t+1}, a) - Q_\theta(s_t, a_t)\right]$$

  $s_t \leftarrow s_{t+1}$
end while

Algorithm 2 DQN Learning- Experience Replay Learning

Initialise a Q network with parameters θ
start with state s_t
Initialise a replay buffer D
while True **do**
  take action a_t
  observe next state s_{t+1} and R_{t+1}
  save the transition $(s_t, a_t, R_{t+1}, s_{t+1})$ in D
  while some epochs **do**
    sample a mini-batch N from the replay buffer D
    calculate the gradient

$$\theta \leftarrow \theta + \alpha \bigtriangledown_\theta \frac{1}{N} \sum_{i=1}^{N} \left[R^i_{t+1} + \gamma max_{a^i} Q_\theta(s^i_{t+1}, a^i) - Q_\theta(s^i_t, a^i_t)\right]$$

  end while
  $s_t \leftarrow s_{t+1}$
end while

Algorithm 3 DQN Learning- Experience Replay Learning with Target network

Initialise a Q network with parameters θ and Target Q network with parameters θ'
$\theta' \leftarrow \theta$
start with state s_t
Initialise a replay buffer D
while True **do**
  take action a_t
  observe next state s_{t+1} and R_{t+1}
  save the transition $(s_t, a_t, R_{t+1}, s_{t+1})$ in D
  while some epochs **do**
    sample a mini-batch N from the replay buffer D
    calculate the gradient

$$\theta \leftarrow \theta + \alpha \bigtriangledown_\theta \frac{1}{N} \sum_{i=1}^{N} \left[R^i_{t+1} + \gamma max_{a^i} Q_{\theta'}(s^i_{t+1}, a^i) - Q_\theta(s^i_t, a^i_t)\right]$$

  end while
  $\theta' \leftarrow \theta$
  $s_t \leftarrow s_{t+1}$
end while

7 Policy Gradient Algorithms

Policy Gradient Theorem
you have a stochastic policy $\pi(a|s)$.

$$\bigtriangledown v_\pi(s) = \bigtriangledown \left[\sum_a \pi(a|s) Q_\pi(s,a)\right]$$

$$\bigtriangledown J(\theta) \propto \sum_s \mu(s) \sum_s Q_\pi(s,a) \bigtriangledown (\pi_\theta(a|s)$$

$$\bigtriangledown J(\theta) = \mathbb{E}_\pi \left[G_t \bigtriangledown ln\pi(a|s)\right]$$