Reinforcement Learning Theory Learning

my proof of non-linear function approximation with off policy control problem

Yue Wang

November 21, 2016

Abstract

a simple note for non-linear function approximation with off policy control problem

1 Introduction

RL theory is very long history and these days

- Bullet point one
- Bullet point two

1. Numbered list item one
2. Numbered list item two

2 Notation and Formatting

some notations:

$\mathcal{S}$	the state space
$\mathcal{A}$	the action space
s_t	the state at time t, actual
s	the state
r_t	the reward at time t, actual
$r(s, a, s')$	the reward from stat s take action to stat s';
$v_k(s)$	the value of stat s at k iteration
v_k	the value function at time k iteration
$p(s, a, s')$	the probability of transfer to s' when given the current stat s and action a
$p(\pi, s, a, s')$	the probability of transfer to s' when given the current stat s and policy π
π	the policy
$\pi(s, a)$	the probability of take action a given the current stat s under policy π

3 RL algorithms

Q-learning is an off policy control algorithms to solve reinforcement problem. Traditional Q-learning problem can convergence to optimal Q function of each state-action pair. However when the state space is large, use look up table to store Q function of each state-action pair is inefficient. So we employ function approximation to solve this problem. The state-of-art approach is use neural network to encode the Q-value function which is non-linear function approximation and use Q-learning algorithms to update the network parameter.

3.1 Q-learning with look up table

update rule of Q-learning:

- $q_{k+1} = (1 - \alpha)q_k(s_t, a_t) + \alpha \max_b [r_t + \gamma q_k(s_{t+1}, b)]$

3.2 Q-learning with function approximation

update rule of Q-learning with function approximation:

- suppose we use $f_\theta(s,a)$ to denote the approximated Q-value function with parameter θ
- update rule $\theta_{k+1} = \theta_k + \alpha_k \cdot \frac{\partial f_\theta(s_t,a_t)}{\partial \theta} \cdot [r(s,a,s') + \gamma \max_{b \in \mathcal{A}} f_{\theta_t}(s_{t+1},b) - f_{\theta_t}(s_t,a_t)]$

3.3 Q-learning with non-learning function approximation

First of all, we employ some notation to simplify our statement

Let f_{θ_t} denotes $E\left(f_{\theta_t}(s_t,a_t)\right)$,
g_{θ_t} denotes $E\left(\max_{b \in \mathcal{A}} f_{\theta_t}(s_{t+1},b)\right)$,
f'_θ denotes $E\left(\frac{\partial f_\theta(s_t,a_t)}{\partial \theta}\right)$,

M_t denotes $\frac{\partial f_\theta(s_t,a_t)}{\partial \theta} \cdot [r(s,a,s') + \gamma \max_{b \in \mathcal{A}} f_{\theta_t}(s_{t+1},b) - f_{\theta_t}(s_t,a_t)] - f'_\theta \cdot [r + \gamma \cdot g(\theta_t) - f(\theta_t)]$

Then the update rule become $\theta_{t+1} = \theta_t + \alpha \cdot \{f'_\theta \cdot [r + \gamma \cdot g(\theta_t) - f(\theta_t)] + M_t\}$

As in , we need to verify 4 conditions

(1) $\sum_{t=1}^{\infty} \alpha_t = \infty \quad \sum_{t=1}^{\infty} \alpha_t^2 < \infty$

(2) M_t is martingale difference sequence or $E\left(M_t\right) = 0$

(3) $h_{\theta_t} = f'_\theta \cdot [r + \gamma \cdot g(\theta_t) - f(\theta_t)]$ is Lipschitz and there exist $h_\infty(\theta_t)$ such that

$$\lim_{n \to \infty} \frac{h\left(n \cdot \theta_t\right)}{n} = h_\infty(\theta_t) \tag{1}$$

(4) ODE $\dot{\theta} = h(\theta) = f'_\theta \cdot [r + \gamma \cdot g(\theta_t) - f(\theta_t)]$ has global asymptotic stability point

References

Leemon Baird. Residual Algorithms: Reinforcement Learning with Function Approximation. *Icml*, pages 30–37, 1995. ISSN 1098-6596. doi: 10.1017/CBO9781107415324.004. URL `http://kirk.usafa.af.mil/{~}baird`.

Dimitri P Bertsekas. A counterexample to temporal differences learning. *Neural Computation*, 7(2):270–279, 1995.

Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction, 2011.

JN Tsitsiklis and B Van Roy. Feature-based methods for large scale dynamic programming. *Machine Learning*, 1996. URL `http://link.springer.com/article/10.1023/A:1018008221616`.