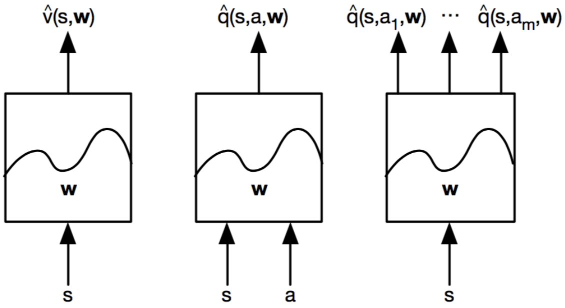
香港中文大學
The Chinese University of Hong Kong

Reinforcement Learning: function approximation

Zihao DENG
Wenqian ZHAO

Department of Computer Science and Engineering
The Chinese University of Hong Kong

Value Function Approximation



- Tablular methods: impossible to record all states for real word problems
- Function approximation: generalize from seen states to unseen states

Value Function Approximation

- Goal: find parameter vector w minimising mean-squared error between approximate value function $\hat{v}(S, w)$ and true value function $v_\pi(S)$

$$J(w) = ||v_\pi(S) - \hat{v}(S, w)||_2^2 \tag{1}$$

- Stochastic gradient descent samples the gradient

$$\Delta w = \alpha(v_\pi(s) - \hat{v}(S, w))\nabla_w \hat{v}(S, w) \tag{2}$$

- In reality we don't have the true value function $v_\pi(S)$
  - For Monte-Carlo, use discounted return G_t
  - For TD, use $R_{t+1} + \lambda \hat{v}(S_{t+1}, w)$

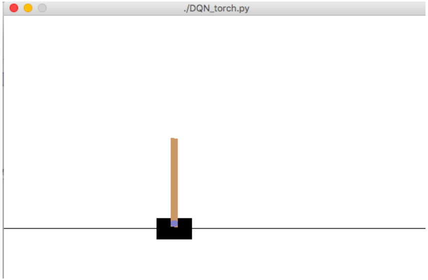
Deep Q-Networks (DQN)

- Take action a_t according to ϵ-greedy policy
- Store transition $(s_t, a_t, r_{t+1}, s_{t+1})$ in memory D
- Sample random mini-batch of transitions (s, a, r, s') from D
- Compute Q-learning targets w.r.t. old, fixed parameters w^-
- Optimise MSE between Q-network and Q-learning targets

$$L(w) = \mathrm{E}_{s,a,s,r' \sim D_i}[(r + \gamma \max_{a'} Q(s', a'; w^-) - Q(s, a; w))^2] \tag{3}$$

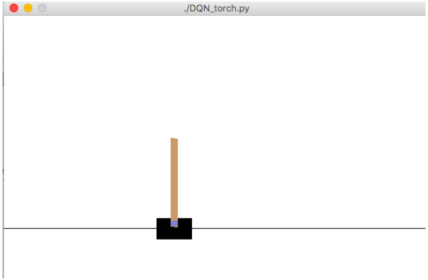
- Two important tricks in ensuring convergence: experience replay and fixed target

Deep Q-Networks(DQN): play games in OpenAI gym



- States are represented by 4-element tuples (position,cart velocity,angle,tip velocity)
- Actions can be either moving left or right
- Function approximator is a feed foward neural network
- 1 hidden layer with 10 neurons, 2 output neurons representing value estimation for two actions
- Implemented using torch and tensorflow, can stay alive for 1 minute

Deep Q-Networks(DQN): play games in OpenAI gym



- States are represented by 4-element tuples (position,cart velocity,angle,tip velocity)
- Actions can be either moving left or right
- Function approximator is a feed foward neural network
- 1 hidden layer with 10 neurons, 2 output neurons representing value estimation for two actions
- Implemented using torch and tensorflow, can stay alive for 1 minute