Reinforcement Learning (RL)

Chapter 7:

Value Function Approximation (VFA)

Saeed Saeedvand, Ph.D.

Contents

In this Chapter:

- ✓ Parameterization meaning
- ✓ Value Function Approximation for Q-Learning
- ✓ Value Function Approximation for SARSA
- ✓ Value Function Approximation for E-SARSA
- ✓ Advantages of Value Function Approximation

Aim of this chapter:

- ✓ Understand the general concepts of the Value Function Approximation and loss function to find proper weighs for reinforcement learning networks.

Value Function Approximation

Idea - Challenge

- ✓ In all previous approaches we considered tabular representation for value function V or state-action function (Q-value)
  - In form of array, vector, matrix, or lookup table

- ✓ In many problems the state space and/or action spaces is very large:
  - Very slow to learn the value of each state individually
  - All possible state-values or Q-values may not fit in the memory

Value Function Approximation

Solution if MDP is big

✓Find a way to **parameterize the value function** and remove tables as storage.

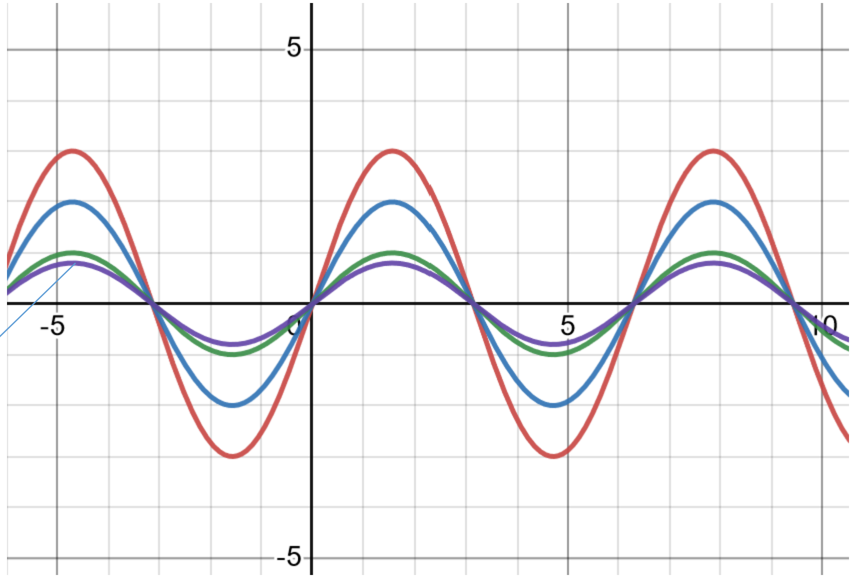
What does parameterization mean?

w sin x

1sin x
2sin x
3sin x

w

target



Value Function Approximation

- ✓ We can Estimating **value** with **value function approximation**
- ✓ We usually represents the state-value or Q-value function with a parameterized function ($\hat{Q}$ or $\hat{V}$).

$$V_\pi(s) \approx \hat{V}(s, w)$$

$$Q_\pi(s, a) \approx \hat{Q}(s, a, w)$$

Value Function Approximation

✓ Lets consider state s is indicated by set of features $\mathrm{s} = (x_1, x_2, \dots, x_n)^T$

If we write a simple linear approximation:

For state-value function:

$$\hat{V}(s, w) = \sum_{i=1}^{n} w_i x_i(s)$$

For state-action function:

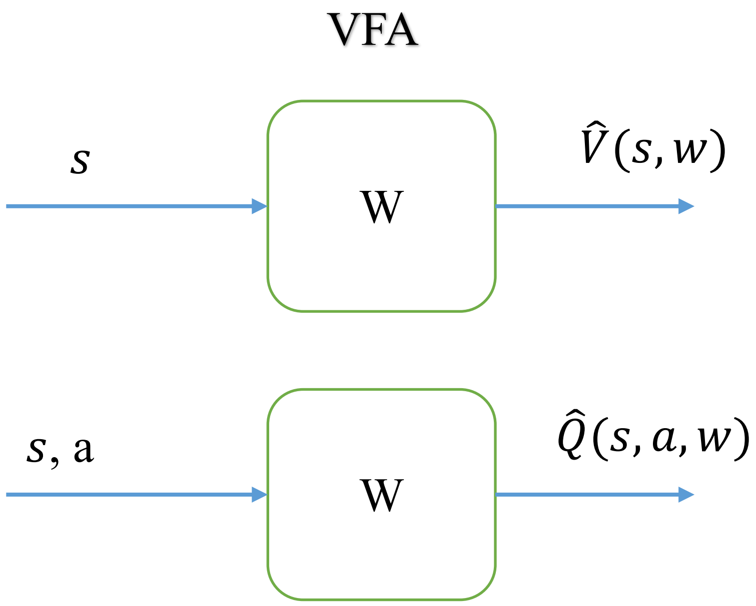
$$\hat{Q}(s, a, w) = \sum_{i=1}^{n} w_{ai} x_i$$

Value Function Approximation

- ✓ Also can be represented as:

$$\hat{V}(s, w) \approx V_\pi(s)$$

$$\hat{Q}(s, a, w) \approx Q_\pi(s, a)$$



Usually **Neural Network** is the most proper option is among different function approximators.

Value Function Approximation

If we write a simple **Non-linear approximation:**

For state-value function:

$$\hat{V}(s, w) = g(x; w)$$

For state-action function:

e.g. Neural Network

$$\hat{Q}(s, a, w) = g(x; w)$$

Value Function Approximation

- ✓ Therefore we can define an **objective function** $E(w)$ (also known as $J(w)$) as **minimizing loss** between target value and predicted value.

$$E(w) = \mathbb{E}_{\pi}[(V_{\pi}(s) - \hat{V}(s,w))^2]$$

- ✓ This is an optimization problem (find parameter vector w to minimize MSE).
- ✓ For instance **Stochastic gradient descent (SGD)** algorithm can be used to solve it.

Value Function Approximation

How to calculate target and predicted values for VFA?

- ✓ There can be multiple solutions
- ✓ The most common way is to use TD
- ✓ For instance we can use Q-learning

$$E(w) = \mathbb{E}_{\pi}[(V_{\pi}(s) - \hat{V}(s,w))^2]$$

VFA as Q-learning

$$E(w) = \mathbb{E}[(R(s_t, a_t) + \gamma \operatorname{Max}_a[\hat{Q}(s_{t+1}, a_{t+1}; w)] - \hat{Q}(s_t, a_t; w))^2]$$

$$\Delta w = \alpha[R(s_t, a_t) + \gamma \operatorname{Max}_a[\hat{Q}(s_{t+1}, a_{t+1}; w)] - \hat{Q}(s_t, a_t; w)\nabla_w \hat{Q}(s_t, a_t; w)]$$

objective function

Gradient (Nambla)

Note: The real target value $Q(s_{t+1}, a_{t+1})$ in practice is also prediction from model as $\hat{Q}(s_{t+1}, a_{t+1}; w)$.

Value Function Approximation

How to calculate target and predicted values for VFA?

VFA as SARSA

$$E(w) = \mathbb{E}_{\pi}[(V_{\pi}(s) - \hat{V}(s,w))^2]$$

$$E(w) = \mathbb{E}[((R(s_t, a_t) + \gamma\, \hat{Q}(s_{t+1}, a_{t+1}; w) - \hat{Q}(s_t, a_t; w))^2]$$

$$\Delta w = \alpha[R(s_t, a_t) + \gamma \hat{Q}(s_{t+1}, a_{t+1}; w) - \hat{Q}(s_t, a_t; w) \nabla_w \hat{Q}(s_t, a_t; w)]$$

Value Function Approximation

How to calculate target and predicted values for VFA?

VFA as ESARSA

$$E(w) = \mathbb{E}_{\pi}[(V_{\pi}(s) - \hat{V}(s,w))^2]$$

$$E(w) = \mathbb{E}[((R(s_t, a_t) + \gamma\, \hat{Q}(s_{t+1}, a_{t+1}; w) - \hat{Q}(s_t, a_t; w))^2]$$

$$\Delta w = \alpha[R(s_t, a_t) + \gamma \sum \hat{\pi}(a|s_{t+1})\hat{Q}(s_{t+1}, a_{t+1}; w) - \hat{Q}(s_t, a_t; w) \nabla_w \hat{Q}(s_t, a_t; w)]$$

Value Function Approximation

Advantage

Solving memory problem

- ✓ State action pair table is very big

Generalization

- ✓ For non-seen states there can be approximation of some actions

Continues

- ✓ More applicable for continues environments