Value Function Approximation I

Alina Vereshchaka

CSE4/510 Reinforcement Learning
Fall 2019

avereshc@buffalo.edu

September 24, 2019

*Slides are based on David Silver Course. Lecture 6

Overview

1. Value Function Approximation

Before

- Last time: how to learn a good policy from experience
- So far, have been assuming we can represent the value function or state-action value function as a vector/ matrix (Tabular representation)
- Many real world problems have enormous state and/or action spaces
- Tabular representation is insufficient

Large-Scale Reinforcement Learning

Reinforcement learning can be used to solve *large* problems, e.g.

- Backgrammon: 10^{20} states
- Go: 10^{170} states
- Helicopter: ?

Large-Scale Reinforcement Learning

Reinforcement learning can be used to solve *large* problems, e.g.

- Backgrammon: 10^{20} states
- Go: 10^{170} states
- Helicopter: ? continuous state space

How can we scale up the model-free methods for prediction and control?

Table of Contents

Large-Scale Reinforcement Learning

- So far we have represented value function by a *lookup table*
  - Every state s has an entry $V(s)$
  - Or every state-action pair s, a has an entry $Q(s, a)$
- Problem with large MDPs:

Large-Scale Reinforcement Learning

- So far we have represented value function by a *lookup table*
  - Every state s has an entry $V(s)$
  - Or every state-action pair s, a has an entry $Q(s, a)$
- Problem with large MDPs:
  - There are too many states and/or actions to store in memory
  - It is too slow to learn the value of each state individually

Large-Scale Reinforcement Learning

- Solution for large MDPs:

Large-Scale Reinforcement Learning

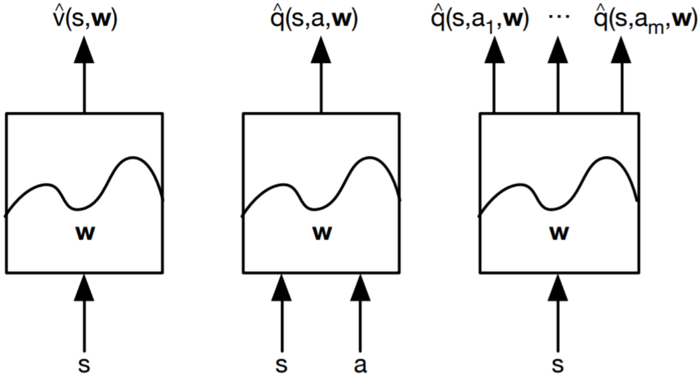
- Solution for large MDPs:
  - Estimate value function with *function approximation*

$$\hat{v}(s, \mathbf{w}) \approx v_\pi(s)$$
$$\hat{q}(s, a, \mathbf{w}) \approx q_\pi(s, a)$$

  - *Generalise* from seen states to unseen states
  - Update parameter **w** using MC or TD learning

Types of Value Function Approximation (VFA)

Represent a (state-action/state) value function with a parameterized function instead of a table



Motivation for VFA

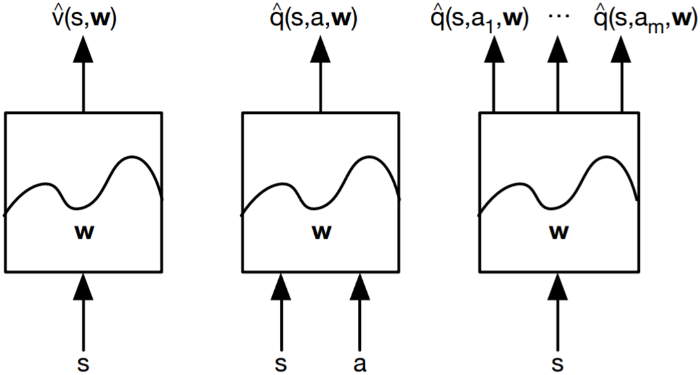
- Don't want to have to explicitly store or learn for every single state a
  - Dynamics or reward model
  - Value
  - State-action value
  - Policy
- Want more compact representation that generalizes across state or states and actions

Benefits of Generalization

- Reduce memory needed to store (P, R)/V/Q/π
- Reduce computation needed to compute (P, R)/V/Q/π
- Reduce experience needed to find a good (P, R)/V/Q/π

Value Function Approximation (VFA)

Represent a (state-action/state) value function with a parameterized function instead of a table



Which function approximator?

Function Approximators

- Many possible function approximators including
  - Linear combinations of features
  - Neural networks
  - Decision trees
  - Nearest neighbors
- For the next few classes we will focus on function approximators that are differentiable (Why?)
- Two very popular classes of differentiable function approximators
  - Linear feature representations
  - Neural networks
- Furthermore, we require a training method that is suitable for *non-stationary, non-iid* data