Q-Function Learning Methods

February 15, 2017

Value Functions

- Definitions (review):

$$Q^\pi(s, a) = \mathbb{E}_\pi \left[r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \mid s_0 = s, a_0 = a \right]$$

Called Q-function or state-action-value function

$$V^\pi(s) = \mathbb{E}_\pi \left[r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \mid s_0 = s \right]$$
$$= \mathbb{E}_{a \sim \pi} \left[Q^\pi(s, a) \right]$$

Called state-value function

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$$

Called advantage function

Bellman Equations for Q^π

- Bellman equation for Q^π

$$
\begin{aligned}
Q^\pi(s_0, a_0) &= \mathbb{E}_{s_1 \sim P(s_1 \mid s_0, a_0)} \left[r_0 + \gamma V^\pi(s_1) \right] \\
&= \mathbb{E}_{s_1 \sim P(s_1 \mid s_0, a_0)} \left[r_0 + \gamma \mathbb{E}_{a_1 \sim \pi} \left[Q^\pi(s_1, a_1) \right] \right]
\end{aligned}
$$

- We can write out Q^π with k-step empirical returns

$$
\begin{aligned}
Q^\pi(s_0, a_0) &= \mathbb{E}_{s_1, a_1 \mid s_0, a_0} \left[r_0 + \gamma V^\pi(s_1, a_1) \right] \\
&= \mathbb{E}_{s_1, a_1, s_2, a_2 \mid s_0, a_0} \left[r_0 + \gamma r_1 + \gamma^2 Q^\pi(s_2, a_2) \right] \\
&= \mathbb{E}_{s_1, a_1 \ldots, s_k, a_k \mid s_0, a_0} \left[r_0 + \gamma r_1 + \cdots + \gamma^{k-1} r_{k-1} + \gamma^k Q^\pi(s_k, a_k) \right]
\end{aligned}
$$

Bellman Backups

- From previous slide:

$$Q^\pi(s_0, a_0) = \mathbb{E}_{s_1 \sim P(s_1 \mid s_0, a_0)}\left[r_0 + \gamma \mathbb{E}_{a_1 \sim \pi}\left[Q^\pi(s_1, a_1)\right]\right]$$

- Define the Bellman backup operator (operating on Q-functions) as follows

$$[\mathcal{T}^\pi Q](s_0, a_0) = \mathbb{E}_{s_1 \sim P(s_1 \mid s_0, a_0)}\left[r_0 + \gamma \mathbb{E}_{a_1 \sim \pi}\left[Q(s_1, a_1)\right]\right]$$

- Then Q^π is a *fixed point* of this operator

$$\mathcal{T}^\pi Q^\pi = Q^\pi$$

- Furthermore, if we apply $\mathcal{T}^\pi$ repeatedly to any initial Q, the series converges to Q^π

$$Q,\ \mathcal{T}^\pi Q,\ (\mathcal{T}^\pi)^2 Q,\ (\mathcal{T}^\pi)^3 Q,\ \cdots \to Q^\pi$$

Introducing Q^*

- Let π^* denote an optimal policy
- Define $Q^* = Q^{\pi^*}$, which satisfies $Q^*(s, a) = \max_\pi Q^\pi(s, a)$
- π^* satisfies $\pi^*(s) = \arg\max_a Q^*(s, a)$
- Thus, Bellman equation

$$Q^\pi(s_0, a_0) = \mathbb{E}_{s_1 \sim P(s_1 \mid s_0, a_0)} \left[r_0 + \gamma \mathbb{E}_{a_1 \sim \pi} \left[Q^\pi(s_1, a_1) \right] \right]$$

  becomes

$$Q^*(s_0, a_0) = \mathbb{E}_{s_1 \sim P(s_1 \mid s_0, a_0)} \left[r_0 + \gamma \max_{a_1} Q^*(s_1, a_1) \right]$$

- Another definition

$$Q^*(s_0, a_0) = \mathbb{E}_{s_1} \left[r + \gamma V^*(s_1) \right]$$

Bellman Operator for Q^*

- Define a corresponding Bellman backup operator

$$[\mathcal{T}Q](s_0, a_0) = \mathbb{E}_{s_1 \sim P(s_1 \mid s_0, a_0)} \left[r_0 + \gamma \max_{a_1} Q(s_1, a_1) \right]$$

- Q^* is a fixed point of $\mathcal{T}$:

$$\mathcal{T}Q^* = Q^*$$

- If we apply $\mathcal{T}$ repeatedly to any initial Q, the series converges to Q^*

$$Q, \ \mathcal{T}Q, \ \mathcal{T}^2Q, \ \cdots \to Q^*$$

Q-Value Iteration

Algorithm 1 Q-Value Iteration

Initialize $Q^{(0)}$

for $n = 0, 1, 2, \ldots$ until termination condition **do**

  $Q^{(n+1)} = \mathcal{T} Q^{(n)}$

end for

Q-Policy Iteration

Algorithm 2 Q-Policy Iteration

Initialize $Q^{(0)}$

for $n = 0, 1, 2, \ldots$ until termination condition **do**

$\pi^{(n+1)} = \mathcal{G} Q^{(n)}$

$Q^{(n+1)} = Q^{\pi^{(n+1)}}$

end for

Q-Modified Policy Iteration

Algorithm 3 Q-Modified Policy Iteration

Initialize $Q^{(0)}$

for $n = 0, 1, 2, \ldots$ until termination condition **do**

$\pi^{(n+1)} = \mathcal{G} Q^{(n)}$

$Q^{(n+1)} = (\mathcal{T}^{\pi^{(n+1)}})^k Q^{(n)}$

end for

Sample-Based Estimates

- Recall backup formulas for Q^π and Q^*

$$[\mathcal{T}Q](s_0, a_0) = \mathbb{E}_{s_1 \sim P(s_1 \mid s_0, a_0)} \left[r_0 + \gamma \max_{a_1} Q(s_1, a_1) \right]$$
$$[\mathcal{T}^\pi Q](s_0, a_0) = \mathbb{E}_{s_1 \sim P(s_1 \mid s_0, a_0)} \left[r_0 + \gamma \mathbb{E}_{a_1 \sim \pi} \left[Q(s_1, a_1) \right] \right]$$

- We can compute unbiased estimator of RHS of both equations using a single sample. Does not matter what policy was used to select actions!

$$[\widehat{\mathcal{T}Q}](s_0, a_0) = r_0 + \gamma \max_{a_1} Q(s_1, a_1)$$
$$[\widehat{\mathcal{T}^\pi Q}](s_0, a_0) = r_0 + \gamma \mathbb{E}_{a_1 \sim \pi} \left[Q(s_1, a_1) \right]$$

- Backups still converge to Q^π, Q^* with this noise[1]

[1] T. Jaakkola, M. I. Jordan, and S. P. Singh. "On the convergence of stochastic iterative dynamic programming algorithms". *Neural computation* (1994); D. P. Bertsekas. *Dynamic programming and optimal control*. Athena Scientific, 2012.

Multi-Step Sample-Based Estimates

- Expanding out backup formula

$$
\begin{aligned}
[\mathcal{T}^\pi Q](s_0, a_0) &= \mathbb{E}_{a_0 \sim \pi}\left[r_0 + \gamma \mathbb{E}_{a_1 \sim \pi}\left[Q(s_1, a_1)\right]\right] \\
[(\mathcal{T}^\pi)^2 Q](s_0, a_0) &= \mathbb{E}_{a_0 \sim \pi}\left[r_0 + \gamma \mathbb{E}_{a_1 \sim \pi}\left[r_1 + \gamma \mathbb{E}_{a_2 \sim \pi}\left[Q(s_2, a_2)\right]\right]\right] \\
&= \mathbb{E}_\pi\left[r_0 + \gamma r_1 + \gamma^2 Q(s_2, a_2)\right] \\
&\dots \\
[(\mathcal{T}^\pi)^k Q](s_0, a_0) &= \mathbb{E}_\pi\left[r_0 + \gamma r_1 + \dots + \gamma^{k-1} r_{k-1} + \gamma^k Q(s_k, a_k)\right]
\end{aligned}
$$

- $\Rightarrow$ can get unbiased estimator of $[(\mathcal{T}^\pi)^k Q](s_0, a_0)$ using trajectory segment $(s_0, a_0, r_0, s_1, a_1, r_1, \dots, s_{k-1}, a_{k-1}, r_{k-1}, s_k)$

Q-function Backups vs V-function Backups

$$[\mathcal{T}Q](s_0, a_0) = \mathbb{E}_{s_1}\left[r_0 + \gamma \max_{a_1} Q(s_1, a_1)\right]$$

$$[\mathcal{T}^\pi Q](s_0, a_0) = \mathbb{E}_{s_1}\left[r_0 + \gamma \mathbb{E}_{a_1 \sim \pi}\left[Q(s_1, a_1)\right]\right]$$

vs

$$[\mathcal{T}V](s_0) = \max_{a_1}[E_{s_1}[r_0 + \gamma V^\pi(s_0)]]$$

$$[\mathcal{T}^\pi V](s_0) = E_{a_0 \sim \pi}[E_{s_1}[r_0 + \gamma V^\pi(s_0)]]$$

max and $\mathbb{E}$ swapped: can get unbiased estimate for $Q(s_0, a_0)$ but not $V(s_0)$ using (s_0, a_0, r_0, s_1).

Why Q Rather than V?

- Can compute greedy action $\max_a Q(s, a)$ without knowing P
- Can compute unbiased estimator of backup value $[\mathcal{T}Q](s, a)$ without knowing P using single transition (s, a, r, s')
- Can compute unbiased estimator of backup value $[\mathcal{T}Q](s, a)$ using *off-policy* data

Sampling-Based Algorithms

- Start with Q-value iteration or Q-policy iteration
- Replace backup by estimator

$$[\mathcal{T}Q](s_t, a_t) \to \widehat{\mathcal{T}Q}_t = r_t + \max_{a_{t+1}} Q(s_{t+1}, a_{t+1})$$

$$[\mathcal{T}^\pi Q](s_t, a_t) \to \widehat{\mathcal{T}^\pi Q}_t = r_t + \mathbb{E}_{a_{t+1} \sim \pi}\left[Q(s_{t+1}, a_{t+1})\right]$$

- Can also replace $[(\mathcal{T}^\pi)^k Q](s_t, a_t)$ (from MPI) by sample-based estimate

Sampling-Based Q-Value Iteration

Algorithm 4 Sampling-Based Q-Value Iteration

Initialize $Q^{(0)}$
for $n = 0, 1, 2, \ldots$ until termination condition **do**
  Interact with the environment for K timesteps (including multiple episodes)
  for $(s, a) \in \mathcal{S} \times \mathcal{A}$ **do**
    $Q^{(n+1)}(s, a) = \text{mean}\left\{\widehat{\mathcal{T}Q}_t, \forall t \text{ such that } (s_t, a_t) = (s, a)\right\}$
      where $\widehat{\mathcal{T}Q}_t = r_t + \gamma \max_{a_{t+1}} Q^{(n)}(s_{t+1}, a_{t+1})$
  end for
end for

$$Q^{(n+1)} = \mathcal{T}Q^{(n)} + noise$$

Least Squares Version of Backup

- Recall $Q^{(n+1)}(s,a) = \text{mean}\left\{\widehat{\mathcal{T}Q}_t, \forall t \text{ such that } (s_t, a_t) = (s,a)\right\}$
- $\text{mean}\{\hat{x}_i\} = \arg\min_x \sum_i \|x_i - x\|^2$
- $Q^{(n+1)}(s,a) = \arg\min_Q \sum_{t \text{ where } (s_t,a_t)=(s,a)} \left\|\widehat{\mathcal{T}Q}_t - Q\right\|^2$
- $Q^{(n+1)}(s,a) = \arg\min_Q \sum_{t=1}^{K} \left\|\widehat{\mathcal{T}Q}_t - Q(s_t,a_t)\right\|^2$

Sampling-Based Value Iteration

Algorithm 5 Sampling-Based Q-Value Iteration (v2)

Initialize $Q^{(0)}$

for $n = 0, 1, 2, \ldots$ until termination condition **do**

Interact with the environment for K timesteps (including multiple episodes)

$Q^{(n+1)}(s,a) = \arg\min_Q \sum_{t=1}^{K} \left\| \widehat{\mathcal{T}Q}_t - Q(s_t, a_t) \right\|^2$

end for

$$Q^{(n+1)} = \mathcal{T}Q^{(n)} + noise$$

Partial Backups

- Full backup: $Q \leftarrow \widehat{\mathcal{T}Q_t}$
- Partial backup: $Q \leftarrow \epsilon\widehat{\mathcal{T}Q_t} + (1-\epsilon)Q$
- Equivalent to gradient step on squared error

$$
\begin{aligned}
Q \rightarrow Q - \epsilon \nabla_Q \left\| Q - \widehat{\mathcal{T}Q_t} \right\|^2 / 2 \\
= Q - \epsilon(Q - \widehat{\mathcal{T}Q_t}) \\
= (1-\epsilon)Q + \epsilon\widehat{\mathcal{T}Q_t}
\end{aligned}
$$

- For sufficiently small ϵ, expected error $\left\| Q - \widehat{\mathcal{T}Q} \right\|^2$ decreases

Sampling-Based Q-Value Iteration

Algorithm 6 Sampling-Based Q-Value Iteration (v3)

Initialize $Q^{(0)}$

for $n = 0, 1, 2, \ldots$ until termination condition **do**

Interact with the environment for K timesteps (including multiple episodes)

$Q^{(n+1)} = Q^{(n)} + \epsilon \nabla_Q \sum_{t=1}^{K} \left\| \widehat{\mathcal{T}Q}_t - Q(s_t, a_t) \right\|^2 / 2$

end for

$$Q^{(n+1)} = Q^{(n)} + \epsilon \left(\nabla_Q \left\| \mathcal{T} Q^{(n)} - Q \right\|^2 / 2 \Big|_{Q=Q^{(n)}} + noise \right)$$
$$= \arg\min_Q \left\| (1-\epsilon) Q^{(n)} + \epsilon (\mathcal{T} Q^{(n)} + noise) \right\|^2$$

- $K = 1 \Rightarrow$ Watkins' Q-learning[2]
- Large K: batch Q-value iteration

[2] C. J. Watkins and P. Dayan. "Q-learning". *Machine learning* (1992).

Convergence

- Consider partial backup update:

$$Q^{(n+1)} = Q^{(n)} + \epsilon\Big(\nabla_Q \big\|\mathcal{T}Q^{(n)} - Q\big\|^2/2\big|_{Q=Q^{(n)}} + noise\Big)$$

- Gradient descent on $L(Q) = \|\mathcal{T}Q - Q\|^2/2$, converges?
- No, because objective is changing, $\mathcal{T}Q^{(n)}$ is a moving target.
- General stochastic approximation result: do "partial update" for contraction + appropriate stepsizes $\Rightarrow$ converge to contraction fixed point[3]
- Given appropriate schedule, e.g. $\epsilon = 1/n$, $\lim_{n\to\infty} Q^{(n)} = Q^*$

[3] T. Jaakkola, M. I. Jordan, and S. P. Singh. "On the convergence of stochastic iterative dynamic programming algorithms". *Neural computation* (1994).

Function Approximation / Neural-Fitted Algorithms

- Parameterize Q-function with a neural network Q_θ
- To approximate $Q \leftarrow \widehat{\mathcal{T}Q}$, do

$$\underset{\theta}{\text{minimize}} \sum_t \left\| Q_\theta(s_t, a_t) - \widehat{\mathcal{T}Q}(s_t, a_t) \right\|^2$$

Algorithm 7 Neural-Fitted Q-Iteration (NFQ)[4]

- Initialize $\theta^{(0)}$.

for $n = 1, 2, \ldots$ **do**

Sample trajectory using policy $\pi^{(n)}$.

$\theta^{(n)} = \text{minimize}_\theta \sum_t \left(\widehat{\mathcal{T}Q}_t - Q_\theta(s_t, a_t) \right)^2$

end for

[4] M. Riedmiller. "Neural fitted Q iteration–first experiences with a data efficient neural reinforcement learning method". *Machine Learning: ECML 2005*. Springer, 2005.