Control and VFA

Prof. Alfio Ferrara

Reinforcement Learning

Improve policy using VFA

The goal of control methods is to evaluate state-action values $\hat{Q}^{\pi}(s,a)$. In the approximate setting, we still have a vector of weights $\mathbf{w}$, such that

$$\hat{Q}^{\pi}(s,a,\mathbf{w}) \approx Q^{\pi} \tag{1}$$

This can be use to perform ϵ-greedy policy improvement through **off-policy learning**.

Theoretical setting with the ground truth

Let's start assuming to know the true state-action values $Q^{\pi}(s,a)$. We can compute the expected error in the prediction by MSE, such as:

$$L(\mathbf{w}) = \mathbb{E}_{\pi}\left[\left(Q^{\pi}(s,a) - \hat{Q}^{\pi}(s,a,\mathbf{w})\right)^2\right] \tag{2}$$

Then we use SGD to find the local minimum of the error function

$$-\frac{1}{2}\nabla_{\mathbf{w}}L(\mathbf{w}) = \mathbb{E}\left[\left(Q^{\pi}(s,a) - \hat{Q}^{\pi}(s,a,\mathbf{w})\right)\nabla\hat{Q}^{\pi}(s,a,\mathbf{w})\right] \tag{3}$$

From with the derive the update value $\Delta(\mathbf{w})$ as

$$\Delta(\mathbf{w}) = -\frac{1}{2}\alpha\nabla_{\mathbf{w}}L(\mathbf{w}) \tag{4}$$

Feature selection

Now, since we are dealing with the state-action values, we use features to represent the state-action pairs instead of the state alone. Thus, we introduce a vector representation of state-action pairs in the following form

$$\mathbf{x}(s,a) = \begin{bmatrix} x_1(s,a) \\ x_2(s,a) \\ \dots \\ x_n(s,a) \end{bmatrix} \tag{5}$$

This allows us to represent $\hat{Q}(s,a,\mathbf{w})$ as a linear combination of features and weights

$$\hat{Q}(s,a,\mathbf{w}) = \mathbf{x}(s,a)^T\mathbf{w} = \sum_{i=1}^{n} x_i(s,a)w_i \tag{6}$$

Control without ground truth

In reality, we do not know the true $Q^\pi(s,a)$, so we need to substitute it with a target value.

Monte Carlo

In Monte Carlo methods, the target will be the return value $G_t(s,a)$, so that the update equation will be

$$\Delta(\mathbf{w}) = \alpha \left(G_t(s,a) - \hat{Q}(s_t,a_t,\mathbf{w}) \right) \nabla \mathbf{w} \hat{Q}(s_t,a_t,\mathbf{w}) \tag{7}$$

which becomes

$$\Delta(\mathbf{w}) = \alpha \left(G_t(s,a) - \mathbf{x}(s_t,a_t)^T\mathbf{w} \right) \nabla \mathbf{w} \mathbf{x}(s_t,a_t)^T\mathbf{w} \tag{8}$$

SARSA

Recall that SARSA update rule in the tabular setting is

$$Q(s_t,a_t) = Q(s_t,a_t) + \alpha \left[r_{t+1} + \gamma \sum_a P(a \mid s_{t+1}) Q(s_{t+1},a) - Q(s_t,a_t) \right] \tag{9}$$

Thus, now we have

$$\Delta(\mathbf{w}) = \alpha \left[r_{t+1} + \gamma \hat{Q}(s_{t+1},a_{t+1},\mathbf{w}) - \hat{Q}(s_t,a_t,\mathbf{w}) \right] \nabla_{\mathbf{w}} \hat{Q}(s_t,s_t,\mathbf{w}) \tag{10}$$

which becomes

$$\Delta(\mathbf{w}) = \alpha \left[r_{t+1} + \gamma \mathbf{x}(s_{t+1},a_{t+1})^T\mathbf{w} - \mathbf{x}(s_t,a_t)^T\mathbf{w} \right] \nabla_{\mathbf{w}} \mathbf{x}(s_t,a_t)^T\mathbf{w} \tag{11}$$

Q-learning

With Q-learning, we use the target $r + \gamma \max_{a_{t+1}} \hat{Q}(s_{t+1}, a_{t+1})$ so that

$$\Delta(\mathbf{w}) = \alpha \left[r_{t+1} + \gamma \max_{a_{t+1}} \hat{Q}(s_{t+1}, a_{t+1}, \mathbf{w}) - \hat{Q}(s_t, a_t, \mathbf{w}) \right] \nabla_{\mathbf{w}} \hat{Q}(s_t, a_t, \mathbf{w}) \tag{12}$$

which becomes

$$\Delta(\mathbf{w}) = \alpha \left[r_{t+1} + \gamma \max_{a_{t+1}} \mathbf{x}(s_{t+1}, a_{t+1})^T \mathbf{w} - \mathbf{x}(s_t, a_t)^T \mathbf{w} \right] \nabla_{\mathbf{w}} \mathbf{x}(s_t, a_t)^T \mathbf{w} \tag{13}$$