Aprendizaje Reforzado
Clase 7

Julián Martínez

¿Qué hacer cuándo el espacio de estados es muy grande/ continuo?

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

- Backgammon: 10^{20} states
- Go: 10^{170} states
- Helicopter: continuous state space
- Robots: informal state space (physical universe)

How can we scale up our methods for prediction and control?

Tomado de los slides del curso de UCL - RL, 2018

Aproximación de función de valor

Idea principal: Cambiar el aproximar la función de valor ***para cada estado*** (approach *tabular*), por ***aproximarla globalmente***.

$$v_\pi(s) \simeq \hat{v}(s, w)$$

El menú para $\hat{v}(s, w)$

Casi toda herramienta que sirve para ***Aprendizaje Supervisado***:

- Árboles de decisión
- Regresión lineal
- Redes neuronales
- Fourier

Función de costo/Método del gradiente

$$J(\omega) := E_{\mu}\left[\left(v_{\pi}(S) - \hat{v}(S;\omega)\right)^2\right]$$

Típicamente μ es la invariante para $p^{\pi}_{s,s'}$

$$\Delta\omega = -\frac{1}{2}\alpha \nabla_{\omega} J(\omega)$$

$$= \alpha E_{\mu}\left[\left(v_{\pi}(S) - \hat{v}(S;\omega)\right) . \nabla_{\omega}\hat{v}(S;\omega)\right]$$

Stochastic Gradient - Incremental

$$\Delta w = \alpha\left(v_\pi(S) - \hat{v}(S, w)\right) \cdot \nabla_w \hat{v}(S; w)$$

$$w^{k+1} = w^k - \alpha\left(v_\pi(S_k) - \hat{v}(S_k; w^k)\right) \cdot \nabla_w \hat{v}(S_k; w^k)$$

$$\Delta w := \alpha\left(G_t - \hat{v}(S_t; w)\right) \cdot \nabla_w \hat{v}(S_t; w)$$

$$\Delta w = \alpha\left(\left(R_{t+1} + \gamma \hat{v}(S_{t+1}; w)\right) - \hat{v}(S_t, w)\right) \cdot \nabla_w \hat{v}(S_t; w)$$

Gradient Monte Carlo Algorithm for Estimating $\hat{v} \approx v_\pi$

Input: the policy π to be evaluated
Input: a differentiable function $\hat{v} : \mathcal{S} \times \mathbb{R}^d \to \mathbb{R}$
Algorithm parameter: step size $\alpha > 0$
Initialize value-function weights $\mathbf{w} \in \mathbb{R}^d$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)

Loop forever (for each episode):
    Generate an episode $S_0, A_0, R_1, S_1, A_1, \ldots, R_T, S_T$ using π
    Loop for each step of episode, $t = 0, 1, \ldots, T-1$:
        $\mathbf{w} \leftarrow \mathbf{w} + \alpha \big[G_t - \hat{v}(S_t,\mathbf{w}) \big] \nabla \hat{v}(S_t,\mathbf{w})$

Semi-gradient TD(0) for estimating $\hat{v} \approx v_\pi$

Input: the policy π to be evaluated
Input: a differentiable function $\hat{v} : \mathcal{S}^+ \times \mathbb{R}^d \to \mathbb{R}$ such that $\hat{v}(\text{terminal},\cdot) = 0$
Algorithm parameter: step size $\alpha > 0$
Initialize value-function weights $\mathbf{w} \in \mathbb{R}^d$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)

Loop for each episode:
    Initialize S
    Loop for each step of episode:
        Choose $A \sim \pi(\cdot|S)$
        Take action A, observe R, S'
        $\mathbf{w} \leftarrow \mathbf{w} + \alpha \big[R + \gamma \hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w}) \big] \nabla \hat{v}(S,\mathbf{w})$
        $S \leftarrow S'$
    until S is terminal

Caso Simple e Importante - Aproximación lineal

$$\phi(s) = \begin{bmatrix} \phi_1(s) \\ \vdots \\ \phi_n(s) \end{bmatrix} \qquad \hat{v}(s;w) = \phi(s)^T w = \sum_{j=1}^{n} \phi_j(s) w_j$$

$$\delta_t := R_{t+1} + \gamma \hat{v}(S_{t+1};w) - \hat{v}(S_t;w)$$

$$\nabla_w \hat{v}(S_t;w) = \phi(S_t) = \phi_t$$

$$\boxed{w^{t+1} = w^t + \alpha\, \delta_t(w^t)\, \phi_t}$$

Sobre la convergencia - MC

$$\min_{w} E_{\mu}\left[\left(G_t - \hat{v}_w(S_t)\right)^2\right]$$

$$\min_{\beta} E\left[\left(Y - \beta^T X\right)^2\right]$$

$$\beta^* = E\left[X X^T\right]^{-1} \cdot E\left[Y . X\right]$$

Sobre la convergencia - TD

$$w^{t+1} = w^t + \alpha\left(R_{t+1} + \gamma w^t \phi_{t+1} - w^t \phi_t\right) \cdot \phi_t$$

$$E\left[W^{t+1} \mid W^t = w\right] = w + \alpha E_w\left[R_{t+1} \phi_t\right] - \alpha w \cdot E_w\left[\left(\phi_t - \gamma \phi_{t+1}\right)^T \phi_t\right]$$

$$\Rightarrow \quad E_\mu\left[R_{t+1} \cdot \phi(S_t)\right] = w \, E_\mu\left[\left(\phi(S_t) - \gamma \phi(S_{t+1})\right)^T \phi(S_t)\right]$$

Sobre convergencia - TD

$$\min_{w} E\left[\left(R_{t+1} + \gamma v_w(S_{t+1}) - v_w(S_t)\right)^2\right]$$

$$w^{t+1} = w^t + \alpha\left(R_{t+1} + \gamma w^t \phi_{t+1} - w^t \phi_t\right) \cdot \left(\phi_t - \gamma \phi_{t+1}\right)$$

$$\neq$$

$$w^{t+1} = w^t + \alpha\left(R_{t+1} + \gamma w^t \phi_{t+1} - w^t \phi_t\right) \cdot \phi_t$$

Como usar los datos más veces (hecharle agua al tang)

$$\mathcal{D} = \left\{ \left(s_1, \hat{v}_1^{\pi}\right), \ldots, \left(s_T, \hat{v}_T^{\pi}\right) \right\}$$

Experience Replay (BATCH)

$$\hat{v}_i^{\pi} \simeq v_{\pi}(s_i)$$

$$\underset{w}{\arg\min} \sum_{i=1}^{T} \left(\hat{v}_i^{\pi} - \hat{v}_w(s_i) \right)^2$$

$$\underset{w}{\arg\min}\ \mathbb{E}_{P_T}\left[\left(\hat{v}^{\pi}(S) - \hat{v}_w(S) \right)^2 \right]$$

Ejemplo concreto de problemas con la notación

Stochastic Gradient Descent with Experience Replay

Given experience consisting of $\langle state, value \rangle$ pairs

$$\mathcal{D} = \{\langle S_1, \hat{v}_1^\pi \rangle, \langle S_2, \hat{v}_2^\pi \rangle, ..., \langle S_T, \hat{v}_T^\pi \rangle\}$$

Repeat:

1. Sample state, value from experience

$$\langle s, \hat{v}^\pi \rangle \sim \mathcal{D}$$

2. Apply stochastic gradient descent update

$$\Delta\theta = \alpha(\hat{v}^\pi - v_\theta(s))\nabla_\theta v_\theta(s)$$

Converges to least squares solution

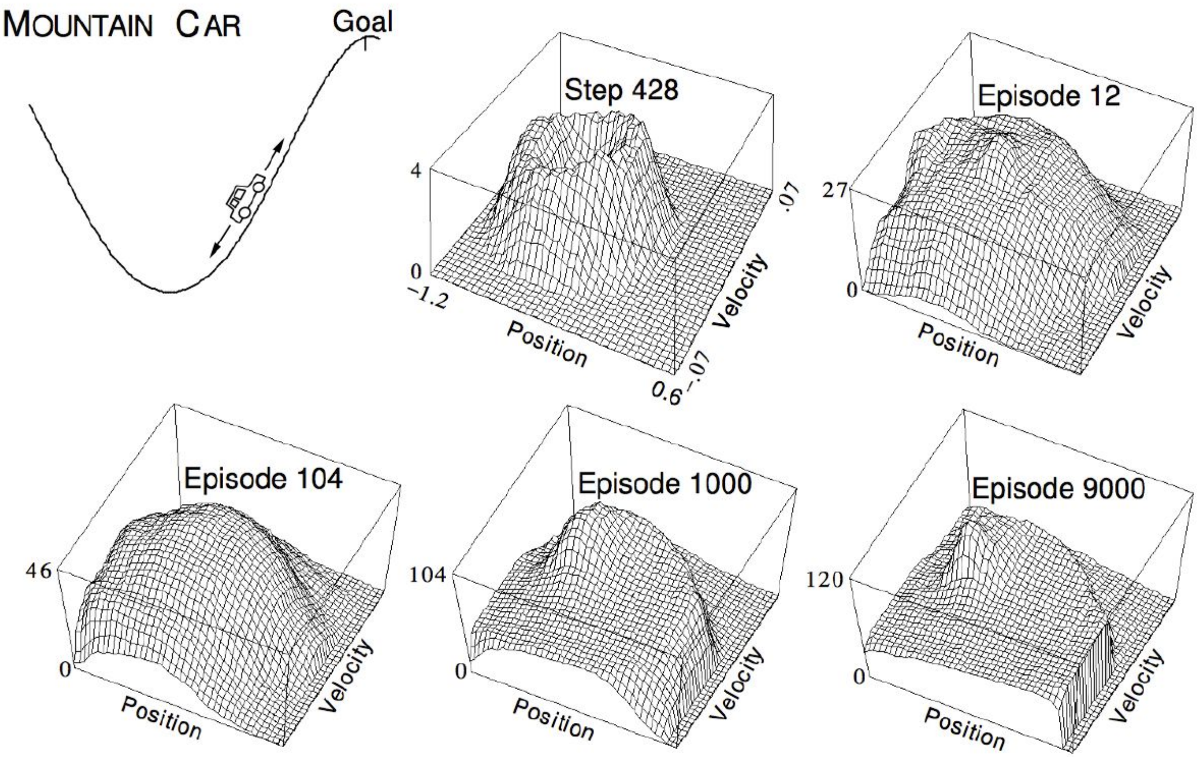
$$\theta^\pi = \underset{\theta}{\text{argmin}}\ \text{LS}(\theta) = \underset{\theta}{\text{argmin}}\ \mathbb{E}_{\mathcal{D}}\left[(\hat{v}_i^\pi - v_\theta(S_i))^2\right]$$

Linear Least Squares Prediction

$$w_B^* = \left(\sum_{t=1}^{T} \phi(s_t) \cdot \phi(s_t)^T \right)^{-1} \sum_{t=1}^{T} \phi(s_t) \hat{v}_t^\pi$$

Ver algoritmo Shermann-Morrison para resolver esto con menor complejidad.

Sobre la "Granularidad del aproximante"



El lado oscuro...

- Sobre la convergencia de temporal difference https://papers.nips.cc/paper/3809-convergent-temporal-difference-learning-with-arbitrary-smooth-function-approximation.pdf
- Detalles y referencias sobre gradient temporal difference y convergencia https://towardsdatascience.com/reinforcement-learning-an-introduction-to-gradient-temporal-difference-learning-algorithms-4a72ce5ab31e


TENGO HASTA AHÍ...
GENERADORMEMES.COM