Chapter 10 On-policy Control with Approximation

Created	@December 21, 2023 3:56 PM
Tags	

10.1 Episodic Semi-gradient Control

Semi-gradient prediction的延伸至 action-value十分的直接，也就是直接近似 $\hat{q} \approx q_\pi$，以函數參數化的型態 $\mathbf{w}$ 去做表示，而先前隨機訓練的範例 $S_t \mapsto U_t$，而現在我們則是以 $S_t, A_t \mapsto U_t$的實例作表達。這個update target U_t可以是對 $q_\pi(S_t, A_t)$ 的任何預測，包含一般的backed up value，像是整個MC return G_t或是任何的 n-step Sarsa的 return (7.4)，一般的gradient-descent 對 action value的更新為

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha \left[U_t - \hat{q}(S_t, A_t, \mathbf{w}_t)\right] \nabla \hat{q}(S_t, A_t, \mathbf{w}_t) \tag{10.1}$$

而 one-step Sarsa的更新則為

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha \left[R_{t+1} - \gamma \hat{q}(S_{t+1}, A_{t+1}, \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t)\right] \nabla \hat{q}(S_t, A_t, \mathbf{w}_t) \tag{10.2}$$

我們稱之為 episodic Semi-gradient one-step Sarsa，在這個固定的policy下，這個policy和 $\mathrm{TD}(0)$ 依樣會收束到同一個點的錯誤區間間 (9.14)。

組成一個control method我們需要將action-value prediction與policy improvement和action selection做連結。比較適合連續的 action，或者是有非常大量的離散action現階段沒有一個確切的答案。不過如果這個action是離散但沒有太大，就可以用上幾章的方法。也就是說，對於 S_{t+1} 下一個可能的 action a，我們可以計算出 $\hat{q}(S_{t+1}, a, \mathbf{w}_t)$然後找到greedy action

$A^*_{t+1} = \arg\max_a \hat{q}(S_{t+1}, a, \mathbf{w})$

policy improvement之後可以靠把estimate policy 改成 soft approximation的 greedy policy 如 ϵ-greedy policy，Action也選自相同的policy。

Episodic Semi-gradient Sarsa for Estimating $\hat{q} \approx q_*$

Input: a differentiable action-value function parameterization $\hat{q} : \mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \to \mathbb{R}$

Algorithm parameters: step size $\alpha > 0$, small $\epsilon > 0$

Initialize value-function weights $\mathbf{w} \in \mathbb{R}^d$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)

Loop for each episode:

$S, A \leftarrow$ initial state and action of episode (e.g., ϵ-greedy)

Loop for each step of episode:

Take action $A,$ observe R, S'

If S' is terminal:

$\mathbf{w} \leftarrow \mathbf{w} + \alpha[R - \hat{q}(S, A, \mathbf{w})]\nabla\hat{q}(S, A, \mathbf{w})$

Go to next episode

Choose A' as a function of $\hat{q}(S, \cdot, \mathbf{w})$(e.g., ϵ-greedy)

$\mathbf{w} \leftarrow \mathbf{w} + \alpha[R + \gamma\hat{q}(S', A', \mathbf{w}) - \hat{q}(S, A, \mathbf{w})]\nabla\hat{q}(S, A, \mathbf{w})$

$S \leftarrow S'$

$A \leftarrow A'$

10-2 Semi-gradient n-step Sarsa

我們可以用n-step return當作更新對象來更新目標，這就是 n-step episodic semi-gradient Sarsa的概念。

$$G_{t:t+n} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n Q_{t+n-1}(S_{t+n}, A_{t+n}, \mathbf{w}_{t+n-1}) \tag{10.4}$$
$$t + n < T$$
$$\text{with } G_{t:t+n} = G_t \text{ if } t + n \geq T$$

n-step update 公式是

$$\mathbf{w}_{t+n} = \mathbf{w}_{t+n-1} + \alpha[G_{t:t+n} - \hat{q}(S_t, A_t, \mathbf{w}_{t+n-1})]\nabla\hat{q}(S_t, A_t\mathbf{w}_{t+n-1}), \qquad (10.4)$$
$$0 \leq t < T$$

Input: a differentiable action-value function parameterization $\hat{q} : \mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \to \mathbb{R}$

Input: a policy π if estimating q_π

Algorithm parameters step size $\alpha > 0,$ small $\epsilon > 0$, a positive integer n

Initialize value-function weights $\mathbf{w} \in \mathbb{R}^d$ arbitrarily(e.g.,$\mathbf{w} = \mathbf{0}$)

All store and access operations (S_t, A_t, R_t) can take their index mod $n + 1$

Loop for each episode:

Initialize and store $S_0 \neq$ terminal

Selected and store an action $A_0 \sim \pi(\cdot, S_0)$ or ϵ-greedy wrt $\hat{q}(S_{t+1}, \cdot, \mathbf{w})$

$T \leftarrow \infty$

Loop for $t = 0, 1, 2, \ldots$

If $t < T$, then:

Take action A_t

Observe and store the next reward as R_{t+1} and the next state as S_{t+1}

If S_{t+1} terminal:

$T \leftarrow t + 1$

else:

Select and store $A_{t+1} \sim \pi(\cdot|S_{t+1})$ or ϵ-greedy wrt $\hat{q}(S_{t+1}, \mathbf{w})$

$\tau \leftarrow t - n + 1$

If $\tau \geq 0$:

$G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} \gamma^{i-\tau-1} R_i$

If $\tau + n < T$, then $G \leftarrow G + \gamma^n \hat{q}(S_{\tau+n}, A_{\tau+n}, \mathbf{w})$

$\mathbf{w} \leftarrow \mathbf{w} + \alpha[G - \hat{q}(S_\tau, A_\tau, \mathbf{w})]\nabla \hat{q}(S_\tau, A_\tau, \mathbf{w})$

Until $\tau = T - 1$

跟以前一樣，中等大小的 n 可以使其有最好的效能。

10.3 Average reward: A New Problem Setting for Continuing Tasks

現在我們遇到了第三個設定，除了episodic還有discounting setting一去用公式導出goal在MDP問題之中，就像discounted setting，**average reward**這個設定是用於continuing problem的。continuing problem是沒有terminal state跟start state的。不過這裡沒有discounting，所以Agent對於較近的reward跟較遠的reward是同樣重視的。這個average reward常常被認為比較常用於動態規劃而不常用於做強化學習。我們下章會談到，這個**discounting setting 會對 function approximation 造成問題**，所以這個average reward是用來替代他的。

在average-reward，一個policy的"品質(quality)"被定義為獲得reward的平均，或是直接稱作average reward，遵守policy下，我們的average reward $r(\pi)$

$$r(\pi) = \lim_{h \to \infty} \frac{1}{h} \sum_{t=1}^{h} \mathbb{E}[R_t | S_0, A_{0:t-1} \sim \pi] \tag{10.6}$$

$$\begin{aligned} &= \lim_{t \to \infty} \mathbb{E}[R_t | S_0, A_{0:t-1} \sim \pi] \\ &= \sum_s \mu_\pi(s) \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) r \end{aligned} \tag{10.7}$$

($\mu(s)$ 是機率分布)

這個期望值是在 S_0 為初始的情況下然後依照著後續連續的Action $A_0, A_1, \ldots, A_{t-1}$根據 π。這第二跟第三的等式成立在 $\mu(s)$ 分布穩定的情況下， $\mu(s) = \lim_{t \to \infty} \Pr[S_t = s | A_{0:t-1} \sim \pi]$，並存在跟獨立於 S_0，換句話說，如果這個MDP是 ergodic 遍歷性，在這個具遍歷性的MDP中，一些初始state或是前幾代的判斷對於agent只會有短暫的效用，長時間還是會以policy跟MDP轉移的機率，要讓 (10.6) 獲得收束，遍歷效力(Ergodicity) 是足夠保證極限的存在卻非必要條件。

在undiscounted continuing case中，我們可以在不同類型的最佳化之間做細微的區分。只是，對大多數的實際目的來說，其實你只在每個time step根據它們的average reward $r(\pi)$ 簡單的排序policies就可以了。這個量(quantity)本質上就是在 π 下的平均reward 也是 (10.7)，或是被稱為 reward rate，特別是，我們認為所有能夠達到 $r(\pi)$ 的所有policy都為最佳的。

我們需要注意到恆定狀態(steady state)的分佈 μ_π是一種特別的分布，如果你根據 π 選擇action，那你還是會在相同的分布中，也就是說

$$\sum_s \mu_\pi(s) \sum_a \pi(a|s) p(s'|s, a) = \mu_\pi(s') \tag{10.8}$$

在 average-reward 的設定，return被定義為reward跟 average reward的差異：

$$G_t = R_{t+1} - r(\pi) + R_{t+2} - r(\pi) + R_{t+3} - r_(\pi) \ldots \tag{10.9}$$

這被稱之為 differential return，然後對應的value function被稱為 differential value function。不同的value function是根據new return所定義，就像傳統的return適用discounted return，我們會利用相同的(notation)符號， $v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$還有 $q_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$(類似 v_* 及 q_*)對於differential value。Differential value function也有其Bellman Equations，只有一小部分的變動，我們純粹就是移除所有 γ 然後將所有 reward 改成 reward 跟 $r(\pi)$的差異。

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\Big[r - r(\pi) + v_\pi(s')\Big]$$

$$q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)\Big[r - r(\pi) + \sum_{a'} \pi(a'|s') q_\pi(s',a')\Big]$$

$$v_*(s) = \max_a \sum_{s',r} p(s',r|s,a)\Big[r - \max_\pi r(\pi) + v_*(s')\Big]$$

$$q_*(s,a) = \sum_{s',r} p(s',r|s,a)\Big[r - \max_a r(\pi) + \max_a q_*(s',a')\Big]$$

TD error也有differential版本的

$$\delta_t = R_{t+1} - \bar{R}_t + \hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_t, \mathbf{w}_t) \tag{10.10}$$

還有

$$\delta_t = R_{t+1} - \bar{R}_t + \hat{q}(S_{t+1}, A_{t+1}, \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t) \tag{10.11}$$

$\bar{R}_t$ 是在時間 t 的預測平均 reward $r(\pi)$ ，有這些替代的定義，我們大部分的理論結果都可以原封不動的套用這個 average reward 的設定。

舉個例子，average reward版本的semi-gradient Sarsa (10.2)就可以被定義為

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha \delta_t \nabla \hat{q}(S_t, A_t, \mathbf{w}_t) \tag{10.12}$$

用 (10.11) 提供的 δ_t

其中一個限制是這個演算法不會收束到differential values而會收束到differential values再加上任意的偏差。不過我們可以發現上述 Bellman equation 以及 TD error不受影響如果值都有一樣的偏差，所以實際上這個offset實際上並不會有問題。

Differential semi-gradient Sarsa for estimating $\hat{q} \approx q_*$

Input: a differentiable action-value function parameterization $\hat{q} : \mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \leftarrow \mathbb{R}$

Algorithm parameters: step size $\alpha, \beta > 0$

Initialize value-function weight $\mathbf{w} \in \mathbb{R}^d$ arbitrarily $(\text{e.g.,} \mathbf{w} = \mathbf{0})$

Initialize value-function weight $\bar{R} \in \mathbb{R}$ arbitrarily $(\text{e.g.,} \bar{R} = 0)$

Initialize state S, and action A

Loop for each step :

  Take action A, observe R, S'

  Choose A' as a function of $\hat{q}(S', \cdot, \mathbf{w})$ (e.g.,ϵ-greedy)

  $\delta \leftarrow R - \bar{R} + \hat{q}(S', A', \mathbf{w}) - \hat{q}(S, A, \mathbf{w})$

  $\bar{R} \leftarrow \bar{R} + \beta\delta$

  $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\nabla\hat{q}(S, A, \mathbf{w})$

  $S \leftarrow S'$

  $A \leftarrow A'$

10.4 Deprecating the Discounted Setting

在continuing discounted問題形式在板面型(tabular case)的問題非常有效，然而在預測的問題會讓人困惑是否要使用它。

至於為什麼，我們先考慮到無限的序列，沒有起頭也沒有結尾的序列。State可能只能被feature vector表示，很難去分辨說這些state之間有甚麼實質上的區別。在一些特殊例子中，所有的 feature vector 可能都是一樣的，因此，我們手上有的就只有reward sequence(與actions)，而且效能就只能從這邊來著手。能怎麼做呢？第一個方法就是將長時間的reward做平均，也就是我們以前所說將average reward方法。但是如果是discounting的方法呢？每個time step我們都可以計算它的discounted return，有時候小有時候大，所以我們一樣要取很長的區間去平均他們，在continuing的設置中並沒有開始也沒有結束，也沒有特殊的time step，所以我們沒有什麼能做的了。不過如果你這樣做，那這個discounted value就是average return 的一部分，事實上 $r(\pi)/(1-\gamma)$是policy π 的平均discounted return，所以它本質上就是average reward。特別是在average discounted return設置中，所有policies的排序將完全地與在average-reward設置中一樣，兩種設置上的排序是一致的。這個 γ 甚至可以為0，結果仍然不會改變。

證明在下面區域

The Futility of Discounting in Continuing Problems

Perhaps discounting can be saved by choosing an objective that sums discounted values over the distribution with which states occur under the policy:

$$
\begin{aligned}
J(\pi) &= \sum_s \mu_\pi(s) v_\pi^\gamma(s) && \text{(where } v_\pi^\gamma \text{ is the discounted value function)} \\
&= \sum_s \mu_\pi(s) \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r \,|\, s, a) \left[r + \gamma v_\pi^\gamma(s') \right] && \text{(Bellman Eq.)} \\
&= r(\pi) + \sum_s \mu_\pi(s) \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r \,|\, s, a) \gamma v_\pi^\gamma(s') && \text{(from (10.7))} \\
&= r(\pi) + \gamma \sum_{s'} v_\pi^\gamma(s') \sum_s \mu_\pi(s) \sum_a \pi(a|s) p(s' \,|\, s, a) && \text{(from (3.4))} \\
&= r(\pi) + \gamma \sum_{s'} v_\pi^\gamma(s') \mu_\pi(s') && \text{(from (10.8))} \\
&= r(\pi) + \gamma J(\pi) \\
&= r(\pi) + \gamma r(\pi) + \gamma^2 J(\pi) \\
&= r(\pi) + \gamma r(\pi) + \gamma^2 r(\pi) + \gamma^3 r(\pi) + \cdots \\
&= \frac{1}{1-\gamma} r(\pi).
\end{aligned}
$$

The proposed discounted objective orders policies identically to the undiscounted (average reward) objective. The discount rate γ does not influence the ordering!

每個time step都是完全的相同。通過discounting，每個reward都會在某些return中的每一個位置就出現一次。

這個範例跟上面證明更通俗的論點說明著，如果我們在on-policy distribution上最佳化discounted value，其影響會跟最佳化undiscounted average reward是一樣的；γ的實際值(actual value)是沒有影響的。不過你當然還是可以繼續在解決方法中使用discounting。不同的是，discount就會從problem parameter(問題參數)變成solution method parameter(解決方法參數)!不幸的是，使用discounting algorithms的function approximation並不會最佳化on-policy distribution上的discounted value，也就是說，它就不能保證能夠最佳化average reward。

其最根本的原因追朔到4.2 policy improvement theorem，它在function approximation中失去效用，『如果我們改變policy來改進一個state的discounted value，那我們就可以保證在任何有用的意義上改善總體策略(overall policy)』已然不成立，而那正是支撐強化學習最重要的一個理論，而我們卻在function approximation中丟棄了！

事實上，缺乏策略改進理論(policy improvement theorem)的同時也是total-episodic與average-reward設置上的一個理論缺陷(lacuna)。一旦我們引入function approximation，那我們就無法再保證任何設置上的一個改進。在地13章我們會額外新的強化學習演算法基於

參數化policy，還有一個理論上的保證：policy-gradient theorem，它扮演了類似於 policy improvement theorm類似的用途，但是對於學習action values的方法，我們似乎沒有一個區域性改善的保證(possibly the approach taken by Perkins and Precup (2003) may provide a part of the answer)。我們確實知道， ϵ-greedification有時候可能會得到一些比較不是那麼好的策略(inferior policy)，因為它可能會在好的策略旁來回振動，而不是收斂(Gordon, 1996a)。這是一個存在多個開放性理論問題的領域。

10.5 略

Differential semi-gradient n-step Sarsa for estimating $\hat{q} \approx q_\pi$ or q_*

Input: a differentiable function $\hat{q} : \mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \to \mathbb{R}$, a policy π
Initialize value-function weights $\mathbf{w} \in \mathbb{R}^d$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)
Initialize average-reward estimate $\bar{R} \in \mathbb{R}$ arbitrarily (e.g., $\bar{R} = 0$)
Algorithm parameters: step sizes $\alpha, \beta > 0$, small $\varepsilon > 0$, a positive integer n
All store and access operations (S_t, A_t, and R_t) can take their index mod $n + 1$

Initialize and store S_0 and A_0
Loop for each step, $t = 0, 1, 2, \ldots$:
  Take action A_t
  Observe and store the next reward as R_{t+1} and the next state as S_{t+1}
  Select and store an action $A_{t+1} \sim \pi(\cdot|S_{t+1})$, or ε-greedy wrt $\hat{q}(S_{t+1}, \cdot, \mathbf{w})$
  $\tau \leftarrow t - n + 1$ (τ is the time whose estimate is being updated)
  If $\tau \geq 0$:
    $\delta \leftarrow \sum_{i=\tau+1}^{\tau+n}(R_i - \bar{R}) + \hat{q}(S_{\tau+n}, A_{\tau+n}, \mathbf{w}) - \hat{q}(S_\tau, A_\tau, \mathbf{w})$
    $\bar{R} \leftarrow \bar{R} + \beta\delta$
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\nabla\hat{q}(S_\tau, A_\tau, \mathbf{w})$