Control with function approximation

Corrado Possieri

Machine and Reinforcement Learning in Control Applications

Introduction

- We now want to build and approximate

Addestriamo i pesi per approssimare q cappello verso la funzione qualità

$$\hat{q}(s, a, \mathbf{w}) \simeq q_*(s, a).$$

LO stato continuo, numero di azioni discrete e limitate poiché potrei ottenere un problema non conveso e uiqndi diffice di risolvere Se le iterazioni sono finite e piccone posso decidere di minimzare q rispetto alle azioni.

- Natural extension of prediction in the episodic case.
- Attention needed in the continuing case.
- We follow the general pattern of on-policy GPI.

Episodic Semi-gradient Control

Vogliamo utilizzare la legge di aggiornamento del gradiente per approsismare la funzione qualità. Dobbiamo quindi definire nel dettaglio quale è il target.

- The target update U_t can be any approximation of q_π
  - MC update

Dominio:
$$S_t, A_t \mapsto G_t;$$

  - TD(0) update (SARSA)

$$S_t, A_t \mapsto R_{t+1} + \gamma Q(S_{t+1}, A_{t+1});$$

effettuiamo bootstrapping

  - n-step TD update

$$S_t, A_t \mapsto G_{t:t+n}.$$

nootstrapping all'ennesimo paso

- The general update form is

stima corrente della funzione qualità

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha(U_t - \hat{q}(S_t, A_t, \mathbf{w}_t))\nabla\hat{q}(S_t, A_t, \mathbf{w}_t).$$

misura
(ritorno/misura di bootstrapping)

- Couple action-value prediction methods with techniques for policy improvement and action selection.

$$\hat{q}(S_t, A_t, w_t)$$

$$w_t, S_t \longrightarrow A_t ?$$

$$A_t = \max_a \hat{q}(S_t, a, w_t)$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha(U_t - \hat{q}(S_t, A_t, \mathbf{w}_t))\nabla\hat{q}(S_t, A_t, \mathbf{w}_t).$$

mistura

- MC : G_t
- TD(0) : $R_{t+1} + \gamma \hat{q}(S_{t+1}, A_{t+1}, w_t)$

nSte TD

Semi-gradient SARSA algorithm

Semi gradient poiché considero solo la mia stima corrente

Solo per task episodici

Semi-gradient SARSA algorithm

Input: $\alpha > 0$, $\varepsilon > 0$, approximation function $\hat{q}$
Output: approximate of q_* and π_*

Initialization

$\mathbf{w} \leftarrow$arbitrarily

Loop

$S, A \leftarrow$initial state and action of episode (*e.g.*, ε-greedy)
for each step of the episode **do**
  take action A and observe R, S'
  if S' is terminal **then**
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha(R - \hat{q}(S, A, \mathbf{w}))\nabla\hat{q}(S, A, \mathbf{w})$
    reinitialize the episode
  choose A' as a function of $\hat{q}(S', \cdot, \mathbf{w})$ (*e.g.*, ε-greedy)
  $\mathbf{w} \leftarrow \mathbf{w} + \alpha(R + \gamma\hat{q}(S', A', \mathbf{w}) - \hat{q}(S, A, \mathbf{w}))\nabla\hat{q}(S, A, \mathbf{w})$
  $S \leftarrow S'$
  $A \leftarrow A'$

n-step semi-gradient SARSA algorithm

n-step semi-gradient SARSA algorithm

Input: $\alpha > 0$, a positive integer n, approximation function $\hat{q}$ (lineare e non lineare)
Output: approximate of q_* and π_*

Initialization

$\mathbf{w} \leftarrow$ arbitrary

Loop

initialize $S_0 \neq$ terminal
store $A_0 \leftarrow \varepsilon$-greedy$(\hat{q}(S_0, \cdot, \mathbf{w}))$ stima corrente basata sui pesi a disposizione
$T \leftarrow \infty$ istante terminale che non so qual è
for $t = 0, 1, 2, \ldots$ **do**
  take action A_t
  observe and store R_{t+1} and S_{t+1}
  if S_{t+1} is terminal **then**
    $T \leftarrow t + 1$
  else
    store $A_{t+1} \leftarrow \varepsilon$-greedy$(\hat{q}(S_{t+1}, \cdot, \mathbf{w}))$
  $\tau = t - n + 1$
  if $\tau \geq 0$ **then**
    $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} \gamma^{i-\tau-1} R_i$ Non faccio bootstrapping
    if $\tau + n < T$ **then**
      $G \leftarrow G + \gamma^n \hat{q}(S_{\tau+n}, A_{\tau+n}, \mathbf{w})$ Faccio ootstrapping
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha(G - \hat{q}(S_\tau, A_\tau, \mathbf{w}))\nabla\hat{q}(S_\tau, A_\tau, \mathbf{w})$
  if $\tau = T - 1$ **then**
    proceed to next episode Non cancello la stima corrente di w

Average return

si applica a problemi di tipo continuativo , non minimizzo una somma scontata del ritorno ma il ritorno medio. Peso il futuro quanto peso il presente e quindi non introdurre il fattore di sconto
Introdu

- The average reward setting applies to continuing problems.
- No discounting (delayed rewards count as immediate reward).
- The quality of π is defined as the average rate of reward

Sommatoria di reward non scontati

tutte le azioni seguendo policy

Fattore di qualità di una policy

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

ritorno scontato ad h passi di tempo

- If the MDP is *ergodic*, *i.e.*, the steady state distribution

$$\mu_\pi(s) = \lim_{t\to\infty} \mathbb{P}[S_t = s | A_{0:t} \sim \pi]$$

exists and is independent of S_0, then

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

Differential return and differential value functions

- In the average reward case, we consider the *differential return*

$$G_t = R_{t+1} - r(\pi) + R_{t+2} - r(\pi) + R_{t+3} - r(\pi) + \dots.$$

- *Differential value functions* are defined upon G_t

$$v_\pi(s) = \mathbb{E}[G_t|S_t = s],$$
$$q_\pi(s,a) = \mathbb{E}[G_t|S_t = s, A_t = a].$$

- The corresponding Bellman equations are

Soluzione unica a meno di una costante, Questo non è un problema poiché l'ordinamento delle azioni ottime ottenute da q rimane invariato. Queste tecniche non vanno bene per predizione, ma per controllo si.

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a) \left(r - r(\pi) + v_\pi(s')\right),$$
$$q_\pi(s,a) = \sum_{s',r} p(s',r|s,a) \left(r - r(\pi) + \sum_{a'} \pi(a'|s') q_\pi(s',a')\right),$$
$$v_*(s) = \max_a \sum_{s',r} p(s',r|s,a) \left(r - \max_\pi r(\pi) + v_*(s')\right),$$
$$q_*(s,a) = \sum_{s',r} p(s',r|s,a) \left(r - \max_\pi r(\pi) + \max_{a'} q_*(s',a')\right).$$

Ottengo ritorno finito anche senza scontare

Differential errors

- The TD differential errors are

Target

Stima reward medio

Predizione

$$\delta_t = R_{t+1} - \bar{R}_t + \hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_t, \mathbf{w}_t),$$
$$\delta_t = R_{t+1} - \bar{R}_t + \hat{q}(S_{t+1}, A_{t+1}, \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t),$$
$$\delta_t = R_{t+1} - \bar{R}_{t+n-1} + R_{t+2} - \bar{R}_{t+n-1} + \cdots + R_{t+n} - \bar{R}_{t+n-1} + \hat{q}(S_{t+n}, A_{t+n}, \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t),$$

  where $\bar{R}_t$ is an estimate of $r(\pi)$.

- With these definitions, we can implement most algorithms
  - *e.g.*, the SARSA update is

Differenze

Fattore di Sconto 1

Reward medio

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

    - converges to a differential values plus an arbitrary offset;
    - the Bellman equations and the TD errors are unaffected if all the values are shifted by the same amount.

Differential semi-gradient SARSA

Task Continuativi

Differential semi-gradient SARSA algorithm

Input: $\alpha > 0$, $\beta > 0$, $\varepsilon > 0$, approximation function $\hat{q}$
Output: approximate of q_* and π_*

Cpmvergenza locale

aloaha aggiornamento pesi
beta aggiornamento reward medio

Initialization

$\mathbf{w} \leftarrow$arbitrarily
$\bar{R} \leftarrow$arbitrarily

Loop

$S, A \leftarrow$initial state and action of episode (*e.g.*, ε-greedy)
for each step of the episode **do**
take action A and observe R, S'
choose A' as a function of $\hat{q}(S', \cdot, \mathbf{w})$ (*e.g.*, ε-greedy)
$\delta \leftarrow (R - \bar{R} + \hat{q}(S', A', \mathbf{w}) - \hat{q}(S, A, \mathbf{w}))$ errore alle TD
$\bar{R} \leftarrow \bar{R} + \beta\delta$
$\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\nabla\hat{q}(S, A, \mathbf{w})$
$S \leftarrow S'$
$A \leftarrow A'$

Differential n-step semi-gradient SARSA algorithm

Differential n-step semi-gradient SARSA algorithm

Input: $\alpha > 0$, $\beta > 0$, a positive integer n, approximation function $\hat{q}$
Output: approximate of q_* and π_*

Initialization

$\mathbf{w} \leftarrow$ arbitrary, $\bar{R} \leftarrow$ arbitrary

Loop

initialize $S_0 \neq$ terminal
store $A_0 \leftarrow \varepsilon$-greedy$(\hat{q}(S_0, \cdot, \mathbf{w}))$
$T \leftarrow \infty$
for $t = 0, 1, 2, \ldots$ **do**
  take action A_t, observe and store R_{t+1} and S_{t+1}
  if S_{t+1} is terminal **then**
    $T \leftarrow t + 1$
  else
    store $A_{t+1} \leftarrow \varepsilon$-greedy$(\hat{q}(S_{t+1}, \cdot, \mathbf{w}))$
  $\tau = t - n + 1$
  if $\tau \geq 0$ **then**
    $\delta \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} (R_i - \bar{R}) - \hat{q}(S_\tau, A_\tau, \mathbf{w})$
    if $\tau + n < T$ **then**
      $\delta \leftarrow \delta + \hat{q}(S_{\tau+n}, A_{\tau+n}, \mathbf{w})$
    $\bar{R} \leftarrow \bar{R} + \beta\delta$
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\nabla\hat{q}(S_\tau, A_\tau, \mathbf{w})$
  if $\tau = T - 1$ **then**
    proceed to next episode

Deprecating discounted setting

Nei task continuativi, minimizzare il reward con sconto e avarege return sono uguale.

- Averaging the rewards over a long interval leads to the average reward setting.
  - the average of discounted returns equals $\frac{r(\pi)}{1-\gamma}$.
- The value of γ has no effect with function approximation.
- We lost the policy improvement theorem using function approximation.
- Discounting algorithms with function approximation do not optimize discounted value over the on-policy distribution,

SARSA(λ)

- Eligibility traces can be used also for control.
- The off-line λ-return algorithm uses $\hat{q}$ rather than $\hat{v}$

task episodici

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha(G_t^\lambda - \hat{q}(S_t, A_t, \mathbf{w}_t))\nabla\hat{q}(S_t, A_t, \mathbf{w}_t).$$

- The backward view of this algorithm is

$$\begin{aligned}
\delta_t &= R_{t+1} + \gamma\hat{q}(S_{t+1}, A_{t+1}, \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t), \\
\mathbf{z}_t &= \gamma\lambda\mathbf{z}_{t-1} + \nabla\hat{q}(S_t, A_t, \mathbf{w}_t), \\
\mathbf{w}_{t+1} &= \mathbf{w}_t + \alpha\delta_t\mathbf{z}_t,
\end{aligned}$$

with $\mathbf{z}_{-1} = \mathbf{0}$.

SARSA(λ) with binary features and linear approximation

esempio tile coding

SARSA(λ) with binary features and linear approximation

Input: $\alpha > 0$, $\varepsilon > 0$, function $\mathcal{F}(s, a)$ returning active features
Output: approximate of q_* and π_*

Initialization

$\mathbf{w} \leftarrow$arbitrarily

Loop Tracce di eleggibilità a 0

initialize S
$A \leftarrow \varepsilon$-greedy($\hat{q}(S, \cdot, \mathbf{w})$)
for each step of the episode **do**
  take action A and observe R, S'
  $\delta \leftarrow R$
  for $i \in \mathcal{F}(S, A)$ **do**
    $\delta \leftarrow \delta - w_i$ w_i valori dei pesi delle feature attive
    $z_i \leftarrow z_i + 1$ or $z_i \leftarrow 1$ dutch o replacing
  if S' is terminal **then**
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\mathbf{z}$
    proceed to next episode
  $A' \leftarrow \varepsilon$-greedy($\hat{q}(S', \cdot, \mathbf{w})$)
  for $i \in \mathcal{F}(S', A')$ **do**
    $\delta \leftarrow \delta + \gamma w_i$
  $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\mathbf{z}$
  $\mathbf{z} \leftarrow \gamma\lambda\mathbf{z}$
  $S \leftarrow S'$
  $A \leftarrow A'$

True online SARSA(λ)

True online SARSA(λ) algorithm

Input: $\alpha > 0$, $\lambda > 0$, feature function $\mathbf{x}$ such that $\mathbf{x}(\text{terminal}, \cdot) = 0$
Output: q_*, π_*

Initialization

$\mathbf{w}$ $\leftarrow$arbitrarily

Loop

initialize S; $A \leftarrow \varepsilon$-greedy$(\hat{q}(S, \cdot, \mathbf{w}))$
$\mathbf{x} \leftarrow \mathbf{x}(S, A)$
$\mathbf{z} \leftarrow 0$
$Q_{\text{old}} \leftarrow 0$
for each step of the episode **do**
  take action A and observe R, S'
  $A' \leftarrow \varepsilon$-greedy$(\hat{q}(S', \cdot, \mathbf{w}))$
  $\mathbf{x}' \leftarrow \mathbf{x}(S', A')$
  $Q \leftarrow \mathbf{w}^\top \mathbf{x}$
  $Q' \leftarrow \mathbf{w}^\top \mathbf{x}'$
  $\delta \leftarrow R + \gamma Q' - Q$
  $\mathbf{z} \leftarrow \gamma\lambda\mathbf{z} + (1 - \alpha\gamma\lambda\mathbf{z}^\top\mathbf{x})\mathbf{x}$
  $\mathbf{w} \leftarrow \mathbf{w} + \alpha(\delta + Q - Q_{\text{old}})\mathbf{z} - \alpha(Q - Q_{\text{old}})\mathbf{x}$
  $Q_{\text{old}} \leftarrow Q'$
  $\mathbf{x} \leftarrow \mathbf{x}'$
  $A \leftarrow A'$
  if if S' is terminal **then**
    reinitialize the episode