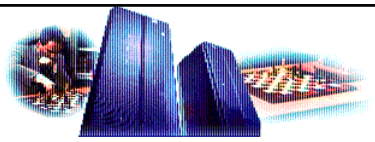
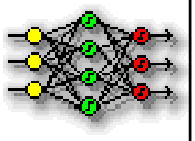




Sistemi Intelligenti
Reinforcement Learning: Evaluative Feedback

Alberto Borghese

Università degli Studi di Milano
Laboratorio di Sistemi Intelligenti Applicati (AIS-Lab)
Dipartimento di Scienze dell'Informazione
borghese@dsi.unimi.it

Riassunto

- Evaluative feedback ed esplorazione.
- La Value function.
- La Q function.
- Problemi non stazionari.

Gli attori del RL

Policy. Descrive l'azione scelta dall'agente: mapping tra stato (input dall'ambiente) e azioni. Funzione di controllo. Le policy possono avere una componente stocastica. Viene utilizzato un modello adeguato del comportamento dell'agente (e.g. tabella, funzione continua parametrica…).

Reward function. Ricompensa **immediata**. Associata all'azione intrapresa in un certo stato. Può essere data al raggiungimento di un goal (esempio: successo / fallimento). E' uno scalare. Rinforzo primario.

Value function. "Cost-to-go". Ricompensa a **lungo termine**. Somma dei reward: costi associati alle azioni scelte istante per istante + costo associato allo stato finale. Orizzonte temporale ampio. Rinforzo secondario.

Environment. Fornisce la reward function, fornisce l'input in base al quale l'agente aggiorna lo stato. Reagisce agli output dell'agente.

Caratteristiche del RL

Non ho informazione su quale sia l'azione migliore.

Ho una valutazione QUALITATIVA sulla bontà dell'azione (evaluative feedback). Non so se ce ne sono di migliori o di peggiori.

Iniziamo con un setting non associativo…..

Alcuni benchmarks e problemi test sul RL:
http://www.cs.rutgers.edu/~mlittman/topics/nips05-mdp/

Il problema del "n-Armed bandit"

Scelta tra n azioni.

La richiesta di scegliere viene ripetuta più volte nel tempo.

La ricompensa è stocastica (e.g. slot machine).

Obbiettivo: viene massimizzata la ricompensa a lungo termine.

Soluzione possibile: selezionare l'azione che fornisce la massima ricompensa a lungo termine.

Come?

Come massimizzare la ricompensa

Consento all'agente di avere memoria.

Memorizzo il valore associato alle diverse azioni.

Posso ad un certo punto scegliere SEMPRE l'azione che mi ha dato la RICOMPENSA MAGGIORE.

GREEDY ACTION (Greedy = Goloso).

EXPLOITING KNOWLEDGE.

Perché dovremmo scegliere un'azione che non appare la migliore (NON GREEDY)?

Exploration

Perchè esploriamo soluzioni diverse.

La ricompensa non è deterministica. Potremmo ottenere di più con altre azioni.

Quello che conta non è la ricompensa istantanea ma la somma delle ricompense ottenute.

Occorre quindi mantenere un istinto ad esplorare azioni diverse.

Il bilanciamento di “exploration” e di “exploitation” è un compito complesso.

Riassunto

- Evaluative feedback ed esplorazione.
- La Value function.
- La Q function.
- Problemi non stazionari.

La Value Function e la scelta delle azioni

Posso selezionare n-azioni: a = a_1….a_n.

Ciascuna di queste azioni ha un suo valore: Q*(a_k) = long-time reward. Q(a) è la funzione valore.

Ciascuna di questa azioni ha anche una stima del suo valore a lungo termine (VALUE): Q(a_k). Supponiamo questa stima funzione del tempo: $Q_t(a_k)$.

Voglio scegliere a_k che massimizza:Q(a).

In caso di exploitation di a_k, posso stimare il "value" all'istante t, come:

$$Q_t(a_k) = \frac{r_1 + r_2 + ... + r_t}{N_t}$$

Dove r_j è il reward per avere scelto a_k all'istante j.

Caratteristiche della Value Function

Value function calcolata come media:

$Q_t(a_k)$ -> $Q^*(a_k)$ per k -> ∞
$Q_t(a_k)$ = 0 k = 0. Nessuna stima disponibile.

Prima possibilità di selezione dell'azione che dà all'istante t, la massima VALUE FUNCTION stimata:

k : $Q_t(a_k) > Q_t(a_j)$ $\forall j \neq k$. $\quad a^* : Q_t(a^*) = \max_a \{Q_t(a)\}$

Così viene EXPLOITED la conoscenza accumulata, è una politica GREEDY.

Non vengono esplorate soluzioni alternative.

Come si può formalizzare un'alternativa?

Exploitation and Exploration

Suppongo che con probabilità, ε, viene scelta un'azione diversa.

Questa azione viene scelta con probabilità uniforme tra le n possibili azioni a disposizione (**ε–Greedy method**).

$Q_t(a_k)$ -> $Q^*(a_k)$ t -> ∞

Near-greary action selection. Come funziona?

$$a^* : Q_t(a^*) = \max_a \{Q_t(a)\} \qquad P = 1 - \varepsilon$$

$$a \neq a^* \qquad P = \varepsilon$$

Esempio: 10-armed testbed

n-armed bandit problem: n = 10: $a = a_1, a_2 \ldots a_{10}$.

Per ogni task, eseguo 1000 volte la scelta dell'azione:

$t = t_1, t_2 \ldots t_{1000}$.
$a = a(t_1), a(t_2) \ldots a(t_{1000})$
$r = r(a(t_1)), r(a(t_2)) \ldots r(a(t_{1000}))$

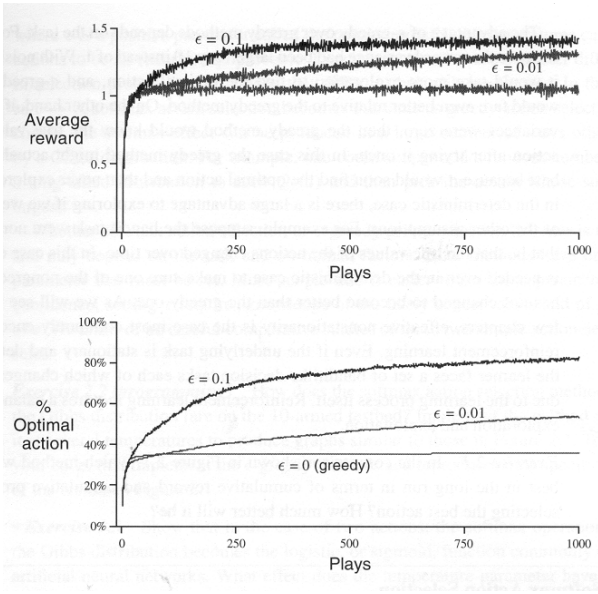
$r(a_k)$ viene selezionato in modo random da una distribuzione Gaussiana con media $Q^*(a_k)$, diversa per le diverse azioni, ma costante per tutto il task, e varianza 1.

Misuro per ogni istante di tempo, t:
- Se l'azione scelta è la migliore o meno.
- La ricompensa ottenuta (Value Function).

Valuta la performance dopo le 1000 giocate di ogni task.

Per ogni task vario $Q^*(a_k)$, estraendolo da una distribuzione Gaussiana con media = 0 e varianza = 1.

Risultati



Media su 2000 task

Domande

Supponiamo che la distribuzione da cui si sceglie il valore medio del reward abbia varianza nulla. Quale metodo funziona meglio: Greedy o ε-Greedy?

Supponiamo che la distribuzione da cui si scegli il valore medio del reward abbia varianza maggiore (e.g. = 10). Cosa succede? Quale metodo si comporterebbe meglio?

In quali altre condizioni sarebbe utile avere esplorazione?

Problemi di memoria

$$Q_t(a_k) = \frac{r_1 + r_2 + ... + r_k}{N_k}$$

Occorre scegliere un algoritmo che calcoli $Q_t(.)$ con un piccolo carico computazionale e di memoria.

Supponiamo di Exploit l'azione a_j. Calcoliamo i reward al tempo t (primi k reward) e li chiamiamo Q_k. Q_k coinciderà con la media delle prime k ricompense:

$$Q_t = \frac{r_1 + r_2 + ... + r_k}{N_k}$$

Scegliendo ancora a_j, otteniamo il seguente valore di Q al tempo t+1:

$$Q_{t+1} = \frac{r_1 + r_2 + ... + r_k + r_{k+1}}{N_{k+1}}$$

Riassunto

- Evaluative feedback ed esplorazione.
- La Value function.
- La Q function.
- Problemi non stazionari.

Determinazione ricorsiva di Q_k

$$Q_k = \frac{r_1 + r_2 + ... + r_k}{N_k} \qquad Q_{k+1} = \frac{r_1 + r_2 + ... + r_k + r_{k+1}}{N_{k+1}}$$

$$Q_{k+1} = \frac{r_1 + r_2 + ... + r_k}{N_{k+1}} + \frac{r_{k+1}}{N_{k+1}} = \frac{Q_k N_k}{N_{k+1}} + \frac{r_{k+1}}{N_{k+1}} =$$

$$\frac{Q_k(N_k + 1 - 1)}{N_{k+1}} + \frac{r_{k+1}}{N_{k+1}} = \frac{Q_k(N_k + 1) - Q_k}{N_{k+1}} + \frac{r_{k+1}}{N_{k+1}} =>$$

$$\boxed{Q_{k+1} = Q_k - \frac{Q_k}{N_{k+1}} + \frac{r_{k+1}}{N_{k+1}}}$$

$Q_1 = r_1(a_j) \quad \forall Q_0$

Osservazioni su Q_k

$$Q_{k+1} = Q_k - \frac{Q_k}{N_{k+1}} + \frac{r_{k+1}}{N_{k+1}} = \boxed{Q_k + \alpha[r_{k+1} - Q_k]} \qquad \alpha = 1/N_{k+1}$$

Occupazione di memoria minima: Solo Q_k e k.
NB k è il numero di volte in cui è stata scelta a_j.

Questa forma è la base del RL. La sua forma generale è:

NewEstimate = OldEstimate + StepSize [Target – OldEstimate]
*NewEstimate = OldEstimate + StepSize * Error.*

StepSize = $\alpha = 1/N_{k+1}$

Pseudo-codice per il calcolo di Q_k.

Riassunto

- Evaluative feedback ed esplorazione.
- La Value function.
- La Q function.
- Problemi non stazionari.

Caso stazionario

$$Q_{k+1} = \frac{r_1 + r_2 + ... + r_k + r_{k+1}}{N_{k+1}}$$

Il peso di ciascun campione è pari a $1/N_{k+1}$.

$$Q_{k+1} = Q_k - \frac{Q_k}{N_{k+1}} + \frac{r_{k+1}}{N_{k+1}}$$

Ogni nuovo campione viene pesato con $1/N_{k+1}$;

Il peso segue un'iperbole.

$$Q_{k+1} = Q_k + \alpha[r_{k+1} - Q_k]$$

Cosa succede se il task è non stazionario?

Caso non stazionario

$$Q_{k+1} = Q_k + \alpha[r_{k+1} - Q_k]$$

Suppongo $\alpha = cost$ ➔ $\alpha_k = \alpha$ $\quad 0 \le \alpha \le 1$

$$\begin{aligned} Q_k &= Q_{k-1} + \alpha[r_k - Q_{k-1}] = \\ &= \alpha r_k + (1-\alpha)Q_{k-1} = \\ &= \alpha r_k + (1-\alpha)[\alpha r_{k-1} + (1-\alpha)Q_{k-2}] = \\ &\quad \alpha r_k + (1-\alpha)\alpha r_{k-1} + (1-\alpha)^2 Q_{k-2}] = \end{aligned}$$

$$= \alpha r_k + (1-\alpha)\alpha r_{k-1} + (1-\alpha)^2 \alpha r_{k-2} + ... + (1-\alpha)^{k-1}\alpha r_1 + (1-\alpha)^k Q_0 =>$$

$$Q_{k+1} = (1-\alpha)^k Q_0 + \sum_{i=1}^{k} \alpha(1-\alpha)^{k-i} r_i$$

$0 \le 1-\alpha \le 1$
$0 \le \alpha \le 1$

Osservazioni

$$Q_{k+1} = (1-\alpha)^k Q_0 + \sum_{i=1}^{k} \alpha(1-\alpha)^{k-i} r_i$$

Dimostrare che la somma dei pesi è = 1.

I reward non sono pesati tutti allo stesso modo: weighted average.

I pesi diminuiscono esponenzialmente: $\frac{1}{b^q}$ q -> + oo

Exponential, recency-weighted average.

Osservazioni statistiche

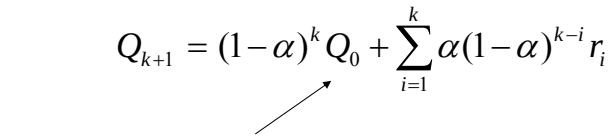
E' noto che devono essere soddisfatte le seguenti condizioni perché una serie converga con probabilità 1:

$$\sum_{k=0}^{\infty} a_k = \infty \qquad \sum_{k=0}^{\infty} a^2{}_k < \infty$$

$a_k = 1/N_k$ soddisfa?

a_k = cost soddisfa?

Condizioni iniziali



$$Q_{k+1} = (1-\alpha)^k Q_0 + \sum_{i=1}^{k} \alpha(1-\alpha)^{k-i} r_i$$

Metodi ad $\alpha = 1/N_k$, Q_0 non viene utilizzato se non al primo passo, viene poi sostituito da Q_1.

Metodi ad α costante, Q_0, conta sempre meno, ma la polarizzazione è permanente ($Q_0 = 0$).

Q_0 può essere utilizzato per fornire della conoscenza a-priori o per favorire l'esplorazione.

Riassunto

- Evaluative feedback ed esplorazione.
- La Value function.
- La Q function.
- Problemi non stazionari.

Direzione di miglioramento: selezione di un certo numero di azioni in un certo range di VALUE function, selezione dell'azione con la maggiore incertezza statistica.