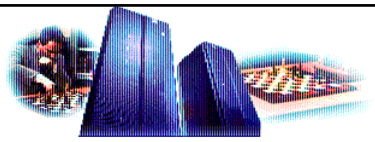
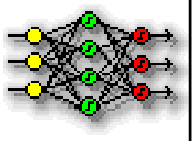




Sistemi Intelligenti
Reinforcement Learning: Evaluative Feedback

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.

Il nostro framework

Task, obbiettivo dell'agente.

Definire la politica ottima per l'agente:

problema di controllo (ingegnere)

problema di ragionamento, planning (AI, informatico)

Gli elementi del RL

Goal. Obbiettivo che deve raggiungere l'agente. Si può aggiungere che l'agente deve raggiungere l'obbiettivo con una policy ottima.

Policy. Descrive l'azione scelta dall'agente: mapping tra stato (output dell'ambiente) e azioni dell'agente. Funzione di controllo. Le policy possono avere una componente stocastica. Viene utilizzata una modalità adeguata pe rappresentare il 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, solitamente ***qualitativo***.

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.

Meccanismo di apprendimento nel RL

Ciclo dell'agente (le tre fasi sono sequenziali):

1) Implemento una policy
2) Aggiorno la Value function
3) Aggiorno la policy.

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"

Situazione iniziale costante.

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?

Slot machine stocastica

Il reward della slot machine è completamente definito dalla densità di probabilità associata alla macchina.

Si suppone la densità di probabilità costante nel tempo.

Per semplicità si suppone che la densità di probabilità sia descrivibile da una funzione analitica, ad esempio una Gaussiana. In questo caso la densità di probabilità è definiti dai parametri della Gaussiana: media e standard deviation.

Che cosa rappresenta la densità di probabilità?

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 = $a_1 \ldots a_n$.

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

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

Voglio scegliere a_k che massimizza:Q(a).

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

$$Q_t(a_k) = \frac{r_1 + r_2 + \ldots + r_{N(a_k)}}{N(a_k)}$$

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 t = 0. Nessuna stima disponibile.

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

k : $Q_t(a_k) > Q_t(a_j)$ $\forall j \neq k$.

$$a^* : Q_t(a^*) = \max_a \{Q_t(a)\}$$

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-gready action selection. Come funziona?

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

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

Uniforme sulle altre a

Esempio: 10-armed testbed

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

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 μ_k, diversa per le diverse azioni, ma costante per tutto il task, e varianza 1. $\mu_k = Q^*(a_k)$

Misuro per ogni istante di tempo, t:

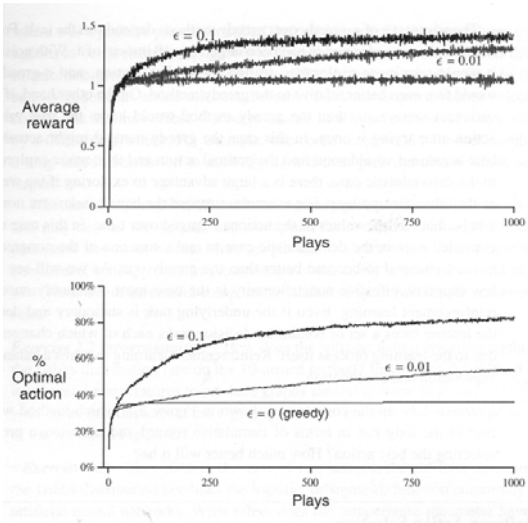
Il reward dell'azione (in questo caso viene dato un reward $\neq$ 0 per ogni azione)

Calcolo la ricompensa totale (Value Function).

Valuta la performance dopo le 1000 giocate di ogni task.

Quanto vale μ_k? Per ogni task vario μ_k estraendolo da una distribuzione Gaussiana con media = 0 e varianza = 1.

Risultati



Media su 2000 task, ciascuno di 1000 giocate (azioni, plays)

Si potrebbe implementare una polirica ε-greedy variabile: ε↓ #Plays↑

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

$$Q_t(a_k) = \frac{r_1 + r_2 + ... + r_{N(a_k)}}{N(a_k)}$$

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

Supponiamo di Exploit l'azione a_k. Calcoliamo i reward al tempo t (primi t reward) e li chiamiamo $Q_t(a_k)$. $Q_t(a_k)$ coinciderà con la media delle prime $N(a_k)$ ricompense:

$$Q_t(a_k) = \frac{r_1 + r_2 + ... + r_{N(a_k)}}{N(a_k)}$$

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

$$Q_{t+1}(a_k) = \frac{r_1 + r_2 + ... + r_N + r_{N+1}}{N+1}$$


Riassunto

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


Determinazione ricorsiva di Q_k

$$Q_t(a_k) = \frac{r_1 + r_2 + ... + r_N}{N} \qquad Q_{t+1}(a_k) = \frac{r_1 + r_2 + ... + r_N + r_{N+1}}{N+1}$$

$$Q_{t+1} = \frac{r_1 + r_2 + ... + r_N}{N+1} + \frac{r_{N+1}}{N+1} = \frac{Q_N N}{N+1} + \frac{r_{N+1}}{N+1} =$$

$$\frac{Q_t(N+1-1)}{N+1} + \frac{r_{N+1}}{N+1} = \frac{Q_t(N+1) - Q_t}{N+1} + \frac{r_{N+1}}{N+1} =>$$

$$Q_{t+1} = Q_t - \frac{Q_t}{N+1} + \frac{r_{N+1}}{N+1}$$

Dipende da t + 1

Non dipende da t + 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, non è necessariamente coincidente con il tempo t!

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

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

StepSize = α = $1/N_{k+1}$

Pseudo-codice per il calcolo di Q_k.

```
####### 1) Definizione delle variabili:
N_scelte = m; eps_greedy = 0.1;          // epsilon dipende dal grado di greedy che voglio dare all'agente
          ## Variabili dell'agente
A = {1, 2,    m};                        // Azioni possibili
Q = {Q1, Q2 ….. Qm} = 0;                 // Value function per ogni azione
N_azioni = {1, 2, …… m}                  // Numero di volte in cui è scelta l'azione j (e collezionato il reward associato).
          ## Variabili dell'ambiente. Date nella simulazione, misurate nell'ambiente nella realtà
// Inizializzo i parametri della distribuzione (stazionaria) dei reward per ogni azione
meanReward = [mean_1, mean_2, ….. mean_m]; stdReward = [mean_1, mean_2, ….. mean_m];

###### 2) Ciclo di funzionamento
while (true)
{         eps = randu([0 1]);                    // Per politica epsilon-greedy
    // Exploitation
          [a_attuale Q_attuale] = SearchMax(Q);  // Cerca l'azione ottima secondo Q
    // Exploration: se eps < epsilon_min, allora exploration
          if (eps < epsilon_min)
          // Devo trovare un'azione diversa da a_attuale -> a_ref
          {         trovato = false; a_ref = a_attuale;
                    while (trovato == false)
                    {         a_attuale = randu(A);
                              if (a_attuale != a_ref)
                              {  trovato = true; Q_attuale = Q(a_attuale);
                              }
                    }
          }
    // Eseguo l'azione a_attuale e misuro il reward ottenuto dalla slot machine
          r_attuale = randg[meanReward(a_attuale), stdReward(a_attuale)];
    // Update i dati per l'azione a_attuale: il numero di azioni ed il value
          N_azioni(a_attuale)++;
          Q(a_attuale) = Q(a_attuale) + 1/[N_azioni(attuale)] * (r_attuale – Q(a_attuale));
}
```

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.

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

Peso decrescente ai nuovi campioni

Cosa succede se il task è non stazionario?

Caso non stazionario

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

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

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

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

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

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

Osservazioni

$$Q_{k+1} = (1-\alpha)^k Q_0 + \sum_{i=1}^{k} \alpha (1-\alpha)^{k-i} r_i = (1-\alpha)^k Q_0 + \sum_{i=0}^{k} w_i r_i$$

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

Il peso di ciascun campione decresce esponenzialmente:

$$w_i = \alpha (1-\alpha)^{k-i} \qquad \alpha < 1$$

Exponential, recency-weighted average.

Somma dei pesi dei reward è unitaria

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

Riscrivo considerando solamente i coefficienti.

$$1 = 1\cdot\alpha + (1-\alpha)\alpha + (1-\alpha)^2\alpha + (1-\alpha)^k =$$

$$\alpha\left(\sum_{i=0}^{k}(1-\alpha)^i - (1-\alpha)^k\right) + (1-\alpha)^k =$$

$$\alpha\left[\frac{(1-\alpha)^{k+1}-1}{(1-\alpha)-1} - (1-\alpha)^k\right] + (1-\alpha)^k =$$

$$\alpha\left[\frac{(1-\alpha)^{k+1}-1}{(-\alpha)}\right] - \alpha(1-\alpha)^k + (1-\alpha)^k =$$

$$(1-\alpha)^k\left[-(1-\alpha)-\alpha+1\right]+1 \quad cvd$$

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.

Come posso gestire una situazione in cui la slot machine cambia improvvisamente la sua densità di probabilità di reward?

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.