Appunti di Reinforcement Learning in Control Applications

Matteo Scarcella 0285629 scarcella@outlook.it

Marzo 2024

Indice

1 Multi-Arm Bandits

Il Multi-Arm Bandit è un problema in cui un decisore deve prendere iterativamente una singola azione fra un insieme possibili scelte. Per formalizzarlo, si definiscono le seguenti quantità:

- $\mathcal{A} = \{1, 2, 3, ..., N\}$ è un insieme di possibili azioni
- $A_t \in \mathcal{A}$ è l'azione presa all'istante t
- R_t è il reward corrispondente all'azione A_t

Il reward viene scelto sulla base di una certa distribuzione di probabilità che dipende dall'azione. Ogni azione ha quindi un valore $q_*(a)$ che in generale non è deterministico, ma è definito da una propria distribuzione di probabilità. In generale:

$$q(a) = \mathbb{E}[R_t | a = A_t] \tag{1}$$

Cioè $q(a)$ è il valore atteso della distribuzione di probabilità associata all'azione $a = A_t$. Il valore q non è noto a priori, pertanto occorrerà calcolarne una stima. Si definisce quindi $Q_t(a)$ la stima di $q(a)$. Un modo per determinare tale stima è il sample-average.

1.1 ε-greedy

Il metodo del Sample-Average consiste nell'inizializzare in valore di $Q_t(a)$ a 0, e man mano assegnare ad esso la media dei reward per l'azione a, cioè la somma di tutti i reward presi precedentemente per l'azione a fino a quello corrente, diviso $N_t(a)$, che è il numero di volte che è stata presa la decisione a al tempo t. Più in dettaglio si ha:

$$Q_t(a) = \begin{cases} 0 & \text{se } N_t(a) = 0 \\ \frac{R_1 + R_2 + ... + R_{N_t(a)}}{N_t(a)} & \text{se } N_t(a) > 0 \end{cases} \tag{2}$$

Si può dimostrare che questa scelta garantisce la convergenza di $Q_t(a)$ a $q(a)$ per $t \to \infty$.

La scelta dell'azione da prendere all'istante t è fondamentale per garantire che l'algoritmo riesca a convergere ad una condizione in cui la preferenza corrisponde all'azione con reward migliore. Una scelta naturale potrebbe essere quella di prendere

$$A_t = \arg\max_a Q_t(a) \quad (\forall a \in \mathcal{A}) \tag{3}$$

Cioè l'azione presa al tempo t è l'azione che fra tutte ha la stima di $q(a)$ più grande. Tuttavia, l'algoritmo all'inizio non ha alcuna informazione sui valori delle azioni, e le stime sono tutte quante pari a 0. Pertanto, una qualunque scelta iniziale potrebbe far cresce la stima $Q_t(a)$ per una particolare azione che non è la migliore, e da quel momento in poi, in base alla (3), l'algoritmo sceglierà sempre la stessa azione, bloccandosi su di essa.

Piuttosto, un metodo più efficace è il metodo "ε-greedy", nel quale vengono combinate sia l'esplorazione sia l'esperienza. In particolare, data ε una quantità arbitrariamente piccola compresa fra 0 e 1, allora:

$$A_t = \begin{cases} \arg\max_a Q_t(a) \quad (\forall a \in \mathcal{A}) & p = 1 - \varepsilon \\ a \in \mathcal{A} & p = \epsilon \end{cases} \tag{4}$$

In questo modo, per $t \to \infty$, si ottiene

- $Q_t(a) \to q(a)$
- $N_t(a) \to \infty$

Per ogni azione a.

Occorre notare che la seconda condizione è fondamentale per evitare condizioni di stallo nell'algoritmo: infatti, in questa circostanza l'algoritmo prende periodicamente delle scelte casuali dall'insieme $\mathcal{A}$ con probabilità $1-\varepsilon$, garantendo così che tutte le scelte vengono effettivamente prese almeno una volta e in particolare il numero di volte che ciascuna azione viene scelte tende sempre ad ∞, perchè anche se la stima di $Q_t(a)$ converge ad un valore di $q(a)$ di molto maggiore rispetto a tutti gli altri, comunque l'algoritmo manterrà la sua componente d'esplorazione attiva. Certamente la tendenza ad ∞ sarà più "lenta" rispetto all'azione con stima migliore.

L'implementazione di tale algoritmo può essere ricavata in modo ricorsivo. Si consideri la stime del valore

dell'azione a al tempo $k+1$, essa è data dalla media pesata di tutti i reward ricevuti negli istanti precedenti, ovvero

$$Q_{k+1} = \frac{1}{k}\sum_{i=1}^{k} R_i \tag{5}$$

Dalla somma può essere estratto il termine k-esimo, ottenendo così

$$Q_{k+1} = \frac{1}{k}(R_k + \sum_{i=1}^{k-1} R_i) \tag{6}$$

Ma la somma rimanente altro non è che $(k-1)Q_k$, pertanto

$$Q_{k+1} = \frac{1}{k}(R_k + (k-1)Q_k) \tag{7}$$

Cioè, riordinando i termini

$$Q_{k+1} = Q_k + \frac{1}{k}(R_k - Q_k) \tag{8}$$

Di seguito una semplice implementazione pratica dell'algoritmo

ε-greedy

for $a = 1$ **to** n **do**
  $Q(a) \leftarrow 0$
  $N(a) \leftarrow 0$
end for

while 1 **do**
  $A \leftarrow \begin{cases} \arg\max_a Q_t(a) & (\forall a \in \mathcal{A}) \quad p = 1-\varepsilon \\ a \in \mathcal{A} & p = \epsilon \end{cases}$
  $R \leftarrow \text{bandit}(A)$
  $N(A) \leftarrow N(A) + 1$
  $Q(A) \leftarrow Q(A) + \frac{1}{N(A)}(R - Q(A))$
end while

1.2 Upper Confidence Bound

Il metodo ε-greedy prevede di scegliere casualmente delle azioni con stime non ottime per effettuare esplorazione ed evitare quindi che l'algoritmo stalli su scelte non ottime. Anzichè scegliere casualmente "rischiando" di prendere delle scelte che potrebbero solo rallentare l'algoritmo, si può effettuare una scelta secondo la legge

$$A_t = \arg\max_a Q_t(a) + c\frac{\ln t}{N_t(a)} \tag{9}$$

Il $\ln t$ è un termine che cresce ad ogni iterazione in modo sub-lineare mentre il termine $N_t(a)$ viene incrementato ogni volta che una azione a viene scelta. In pratica si può affermare che il termine $\frac{\ln t}{N_t(a)}$ offre una stima di quanto l'azione a sia "buona" da scegliere. Infatti, se l'azione a è presa molto di frequente, il termine $N_t(a)$ crescerà quasi linearmente e il rapporto $\frac{\ln t}{N_t(a)}$ sarà minore di 1. Viceversa, se una azione è stata presa poco di frequente, il termine $N_t(a)$ sarà più piccolo rispetto a $\ln t$ nel tempo, rendendo $\frac{\ln t}{N_t(a)}$ più grande, dando quindi la possibilità all'azione di essere scelta. Il coefficiente $c > 0$ può essere visto come un indice di esplorazione.

Nel caso di distribuzioni di probabilità non stazionarie la legge d'aggiornamento può essere modificata nella seguente

$$Q_{k+1} = Q_k + \alpha(R_k - Q_k) \tag{10}$$

Con α costante. Quindi, riordinando

$$Q_{k+1} = \alpha R_k + (1-\alpha)Q_k \tag{11}$$

A questo punto, espandendo ricorsivamente il termine Q_k, si può verificare che l'espressione può essere riscritta in modo più compatto

$$Q_{k+1} = (1-\alpha)^k R_k + \sum_{i=1}^{k} \alpha(1-\alpha)^{k-i} Q_1 \tag{12}$$

Si è quindi ottenuta un'espressione che dipende sia da R che da Q. La particolarità rispetto al caso precedente è che per valori grandi di k, il primo termine decresce. Analogamente, il secondo termine è una somma di valori via via crescenti, dove l'ultimo termine è sempre il più grande. Dunque si può affermare che il fatto di aver scelto α costante consente di ottenere un'espressione che pesa meno i valori di Q e di R più vecchi, mentre pesa di più i valori recenti.

In aggiunta, è possibile adottare la Optimistic Inizialization, che inizializza $Q_t(a)$ a valori molto grandi, facendoli via via decrescere e convergere, teoricamente, ai valori $q(a)$. Di seguito, un'implementazione di UCB + Optimistic Initialization

UCB + Optimistic Initialization

for $a = 1$ **to** n **do**
  $Q(a) \leftarrow \infty$
  $N(a) \leftarrow 0$
end for

while 1 **do**
  $t \leftarrow t + 1$
  $A \leftarrow \arg\max_a Q_t(a) + c\frac{\ln t}{N_t(a)}$
  $R \leftarrow bandit(A)$
  $N(a) \leftarrow N(a) + 1$
  $Q(a) \leftarrow Q(a) + \frac{1}{\alpha}(R - Q(a))$
end while

1.3 Preference Updates

Un'ulteriore strategia potrebbe essere quella di dare un indice di preferenza per una data azione. Sia quindi $H_t(a)$ la preferenza relativa all'azione a al tempo t. Dopo aver effettuato la scelta A_t e aver ricevuto il reward R_t, si aggiornano le preferenze nel modo seguente

$$H_{t+1}(a) = \begin{cases} H_t(a) + \alpha(R_t - \overline{R}_t)(1 - \pi_t(a)) & a = A_t \\ H_t(a) - \alpha(R_t - \overline{R}_t)\pi_t(a) & a \neq A_t \end{cases} \tag{13}$$

Dove $\overline{R}_t$ è il reward medio mentre $\pi_t(a)$ è una distribuzione di tipo softmax definita come segue

$$\pi_t(a) = \frac{\exp H_t(a)}{\sum_{b=1}^{n} \exp H_t(b)} \tag{14}$$

2 Markov Decision Process

Un processo decisionale di Markov (MDP) è un modello che può essere applicato al reinforcement learning. Il modello si assume completamente osservabile. Di fatto, i banditi sono un MDP a singolo stato. Di seguito, verrà modellizzato un MDP.

2.1 Modeling

Una prima proprietà da evidenziare è la proprietà di Markov

$$\mathbb{P}[S_{t+1}|S_t] = \mathbb{P}[S_{t+1}|S_1, S_2, S_3, ..., S_t] \tag{15}$$

Ovvero, l'informazione per determinare lo stato futuro è interamente contenuta nello stato presente, che racchiude anche le informazioni di tutti gli stati precedenti. In altre parole, il futuro è indipendente dal passato e dipende solo dal presente.

Un processo di Markov è composto da un certo numero di stati. Siano s, s' due stati di un processo di Markov, la transizione da uno stato all'altro è definita nel seguente modo

$$P_{s,s'} \triangleq \mathbb{P}[S_{t+1} = s'|S_t = s] \tag{16}$$

Quindi, se il numero di stati è finito, è possibile definire una matrice delle transizioni in cui l'elemento $a_{i,j}$ è la probabilità di transizione dallo stato i allo stato j

$$P \triangleq \begin{bmatrix} P_{1,1} & P_{1,2} & \dots & P_{1,n} \\ P_{2,1} & P_{2,2} & \dots & P_{2,n} \\ \dots & \dots & \dots & \dots \\ P_{n,1} & P_{n,2} & \dots & P_{n,n} \end{bmatrix} \tag{17}$$

Una importante proprietà di questa matrice è

$$\sum_{j=1}^{n} P_{i,j} = 1 \quad i = 1, 2, \dots, n \tag{18}$$

Cioè ogni riga ha somma 1, il che equivale a dire che l'insieme delle possibili azioni di ogni stato coincide con le transizioni stesse.

Infine, sia

$$\pi(t) \triangleq \big[\mathbb{P}[S_t = 1], \mathbb{P}[S_t = 2], \dots, \mathbb{P}[S_t = 2]\big] \tag{19}$$

allora si ha

$$\pi(t+1) = \pi(t)P \tag{20}$$

Dove P è la matrice delle transizioni. É importante notare che $\pi(t)$ è un vettore riga che premoltiplica la matrice P e che quindi restituisce un nuovo vettore riga.

Si può quindi definire una catena di Markov come una coppia (S, P) dove S è un insieme di stati (finiti) e P è la matrice delle transizioni.

2.2 Markov Decision Process

Un modello di Reinforcement Learning basato su Markov Decision Process si compone di un agente che prende decisioni, e un ambiente che è tutto ciò che sta intorno all'agente. Una tipica fotografia di un MDP all'istante t si basa su 4 passi:

- l'agente osserva l'ambiente nello stato S_t
- l'agente prende la decisione relativa all'azione A_t
- l'agente riceve il reward R_{t+1}
- l'agente passa nello stato S_{t+1}

Dunque è possibile definire una dinamica di un MDP nel modo seguente

$$p(s', r|s, a) \triangleq \mathbb{P}[S_{t+1} = s', R_{t+1} = r|S_t = s, A_t = a] \tag{21}$$

con

$$\sum_{s \in \mathcal{S}} \sum_{r \in \mathcal{R}} p(s', r|s, a) = 1 \quad \forall a \in \mathcal{A}, \forall s \in \mathcal{S} \tag{22}$$

Questo vincolo va inteso in questo modo: facendo variare s e a, $p(s', r|s, a)$ diventa una matrice contenente nell'elemento $a_{i,j}$ la probabilità di transitare nello stato s' con il reward r a partire dallo stato i scegliendo l'azione j. Allora, dati uno stato s e un'azione a presa su quello stato, il reward e lo stato successivo sono unici, quindi la probabilità è 1 sulla cella $a_{i,j}$ che soddisfa tale condizione mentre è nulla altrove. In altre parole, il vincolo rappresenta l'univocità della transizione a partire da stato e azione fissati. (? non credo sia così, chiedere)

A partire dalla (21) è possibile definire altre funzioni di probabilità utili nello sviluppo del modello

- Probabilità di transizione dallo stato s allo stato s' data l'azione a

$$p(s'|s, a) \triangleq \sum_{r \in \mathcal{R}} p(s', r|s, a) \tag{23}$$

- Valore atteso del reward per la coppia stato-azione (s, a)

$$r(s, a) \triangleq \sum_{r \in \mathcal{R}} r \cdot \sum_{s' \in \mathcal{S}} p(s', r|s, a) \tag{24}$$

- Valore atteso del reward per la tripla stato - azione - stato successivo (s, a, s')

$$r(s, a, s') \triangleq \sum_{r \in \mathcal{R}} r \cdot \frac{p(s', r|s, a)}{p(s'|s, a)} \tag{25}$$