23.11.2022

- clasificări
- noțiuni/concepte
- important pt notare.

Inteligență Artificială

Cursul 9

Procese de decizie Markov	Învățare bazată pe reinforcement
→ avem un set de stări S și un set de acțiuni A	→ se bazează pe procese de decizie Markov.
→ modelul tranzițiilor $P(S'/S,a)$ este cunoscut	→ modelul de tranziții nu este cunoscut
→ fct. de recompense $R(S)$ este cunoscută	→ fct. $R(S)$ nu este cunoscută
→ calculează politica optimă	→ învață politica optimă

Reinforcement Learning
- Pasivă
  - Bazată pe modele
  - Fără model
    - Value learning
      - Direct Evaluation
      - TD learning
    - Q-learning
- Activă

Învățarea pasivă → un agent execută o politică fixă și o învață

Scopul este să învățăm cât de bună este politica π.

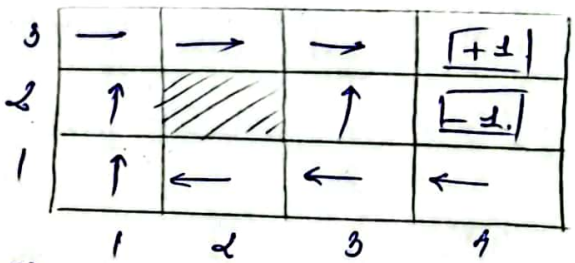
Pasul 1) Învățăm utilitatea $U^{\pi}(s)$ pt fiecare stare.

Pasul 2) Se aplică politica și se învață din experiență

Pasul 3) Evaluăm politica

$(1,1)_{-.04} \rightsquigarrow (1,2)_{-.04} \rightsquigarrow (1,3)_{-.04} \rightsquigarrow (1,2) \rightsquigarrow$

$\rightsquigarrow (1,3)_{-.04} \rightsquigarrow (2,3)_{-.04} \rightsquigarrow (3,3)_{-.04} \rightsquigarrow (4,3)+1$



$(1,1)_{-.04} \rightsquigarrow (1,2)_{-.04} \rightsquigarrow (1,3)_{-.04} \rightsquigarrow (2,3)_{-.04} \rightsquigarrow (3,3)_{-.04} \rightsquigarrow$

$\rightsquigarrow (3,2)_{-.04} \rightsquigarrow (3,3)_{-.04} \rightsquigarrow (4,3)+1$

$(1,1)_{-.04} \rightsquigarrow (2,1)_{-.04} \rightsquigarrow (3,1)_{-.04} \rightsquigarrow (3,2)_{-.04} \rightsquigarrow (4,3)_{-1}$.

Învățarea bazată pe model ↔ învățăm modelul probabilist (MBP)
aplicăm tehnicile de la procesele de decizie Markov. (rezolvă MBP)

Cea mai simplă metodă de a învăța este să învățăm modelul probabilist și recompensele.

MBA / Programarea Dinamică Adaptivă → pt. rezolvarea procesului de decizie Markov.

Vom estima $P(s'|s,\pi(s))$ și $R(s)$ din încercări.

$$U^{\pi}(s) = R(s) + \gamma \sum_{s'} P(s'|s,\pi(s)) U^{\pi}(s') \quad \text{(ecuație liniară)}$$

Se va folosi un tabel de probabilități pt. a calcula procesul de învățare al modelului.

Acă avem jocuri de dimensiuni mari, nu este recomandată folosirea MBA (jocul de table)

Învățarea fără model → nu învățăm modelul probabilist, ci direct politica.

Pasul 1) Estimarea directă a utilității (poate fi considerată o met. de învățare supervizată)

Utilitatea = suma tuturor recompenselor obținute de către agent (reward-to-go)

Dezavantajul: nu se ține cont de ec. Bellman, deci nu se ține cont de utilitățile stărilor succesoare ⇒ convergența este mai mare (se învață mai greu)

Învățarea diferențelor temporale

→ se folosesc ec. Bellman pt. calcul utilități. și o actualizează doar stările direct afectate.

Ecuația diferențelor temporale: calculează diferența dintre utilitatea stării următoare și utilitatea stării curente.

$$U^{\pi}(s) \leftarrow U^{\pi}(s) + \alpha\left(R(s) + \gamma U^{\pi}(s') - U^{\pi}(s)\right)$$

Actualizarea utilităților se face online, adică după fiecare tranziție, nu după fiecare episod.

Valoarea medie $U^{\pi}(s)$ va converge către valoarea corectă.

Dacă se realizează suficiente încercări, tranzițiile care apar dar

Dacă α este o funcție care scade pe măsură ce nr. de vizitări a unei stări crește, atunci $U^{\pi}(s)$ converge către valoarea corectă: $\alpha(n) = \frac{1}{n}$ sau $\alpha(n) = \frac{1}{1+n} \in (0,1]$

La examen: aplicarea învățării diferențelor temporale (slide 18) pt. calcularea utilităților

explicăm formula $U^{\pi}(s) = U^{\pi}(s) + \alpha\left(R(s) + \gamma U^{\pi}(s') - U^{\pi}(s)\right) =$

$$= (1-\alpha)\, U^{\pi}(s) + \alpha\left(R(s) + \gamma U^{\pi}(s')\right)$$

$B \xrightarrow{\text{east}} C$, -2.

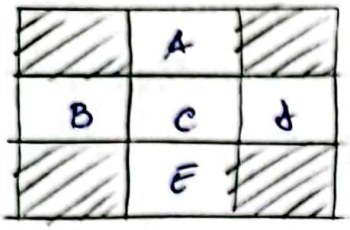
$$U^{\pi}(B) = \underbrace{U^{\pi}(B)}_{0} + \underbrace{\alpha}_{0,5}\left(-2 + 1 \cdot \underbrace{U^{\pi}(C)}_{0} - \underbrace{U^{\pi}(B)}_{0}\right) = -1.$$

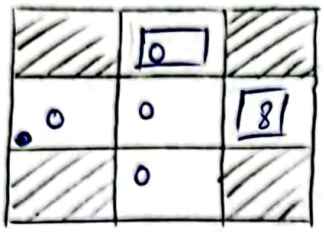
$C \xrightarrow{\text{east}} A$, -2.

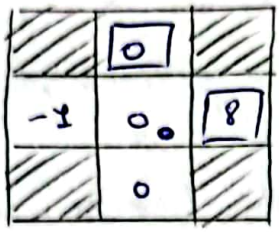
$$U^{\pi}(C) = \underbrace{U^{\pi}(C)}_{0} + \underbrace{\alpha}_{0,5}\left(-2 + 1\, \underbrace{U^{\pi}(A)}_{8} - \underbrace{U^{\pi}(C)}_{0}\right) = \frac{8-2}{2} = 3.$$

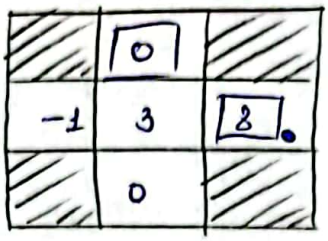
Stările

Tranzițiile observate









$\gamma = 1, \alpha = \frac{1}{2}$

B, east, C, -2

C, east, A, -2

Adaptive Dynamic Programming (PDA)	Temporal Difference Learning
→ este bazat pe model.	→ nu are nevoie de model.
→ actualizează doar un succesor, nu pe toți	→ converge mai greu.
	→ este o aproximare a PDA

Învățarea activă → agentul actualizează politica pe măsură ce învață.

Agentul poate face explorare sau exploatare. La început, acesta trebuie să exploreze deoarece este un mediu nou, iar apoi să exploateze ce a învățat. Între cele 2 trebuie să existe un echilibru deoarece altfel se va ajunge într-un optim local.

ε - greedy:

Fie $\varepsilon \in [0,1]$. Acțiunea următoare selectată va fi { o acțiune aleatoare, cu probab ε / acțiunea optimă, cu probab $1-\varepsilon$

ε nu va scădea niciodată sub un anumit prag.

Alg. Q-Learning → cel mai folosit în RL., care ține cont de stare și de acțiune.

$$Q(s,a) = R(s) + \gamma \sum_{s'} P(s'|s,a) \max_{a'} Q(s',a')$$

↑ Fiecare eșantion (s, a, s', r) actualizează valoarea Q.

Ec. de actualizare TD Q-Learning: $Q(s,a) = Q(s,a) + \alpha\left(R(s) + \gamma \max_{a'} Q(s',a') - Q(s,a)\right)$, $\alpha \in (0,1)$

α = coeficientul de învățare.

s	a	s'	r
A	y_0	B	2
C	y_{top}	A	0
B	y_{top}	A	-2
B	y_0	C	-6
C	y_0	A	2
A	y_0	A	-2

$A \xrightarrow{y_0} B, 2$

$Q(A, y_0) = \underbrace{Q(A, y_0)}_{0} + \underbrace{2}_{0,5} \cdot (2 + \max_{a'} \underbrace{Q(B,a')}_{0} - Q(a, y_0)) = 1.$

$C \xrightarrow{y_{top}} A, 0$

$Q(C, y_{top}) = \underbrace{Q(C, y_{top})}_{0} + \underbrace{2}_{0,5} \cdot (0 + \underbrace{\max_{a'} Q(A,a')}_{1} - \underbrace{Q(C, y_{top})}_{0}) = 0,5$

SARSA - o variantă a alg. Q-Learning, dar nu mai avem acel maxim.

$$Q(s,a) = Q(s,a) + \alpha(R(s) + \gamma Q(s',a') - Q(s,a))$$

Deep Reinforcement Learning → folosește o rețea neuronală care aproximează valorile lui Q.

Eclația de actualizare: $Q(s,a) = Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$

Funcția de pierdere: calculează MSE (Mean Squared Error) a valorii prezise pentru Q.

Target value: $target(s') = r + \gamma \max_{a'} Q(s',a')$ (este estimat → convergență slabă)

Minimize $loss(s,a,s') = (Q(s,a) - target(s'))^2$

O altă metodă este să se folosească metoda gradientului.

Pt lab: de implementat Q-Learning.